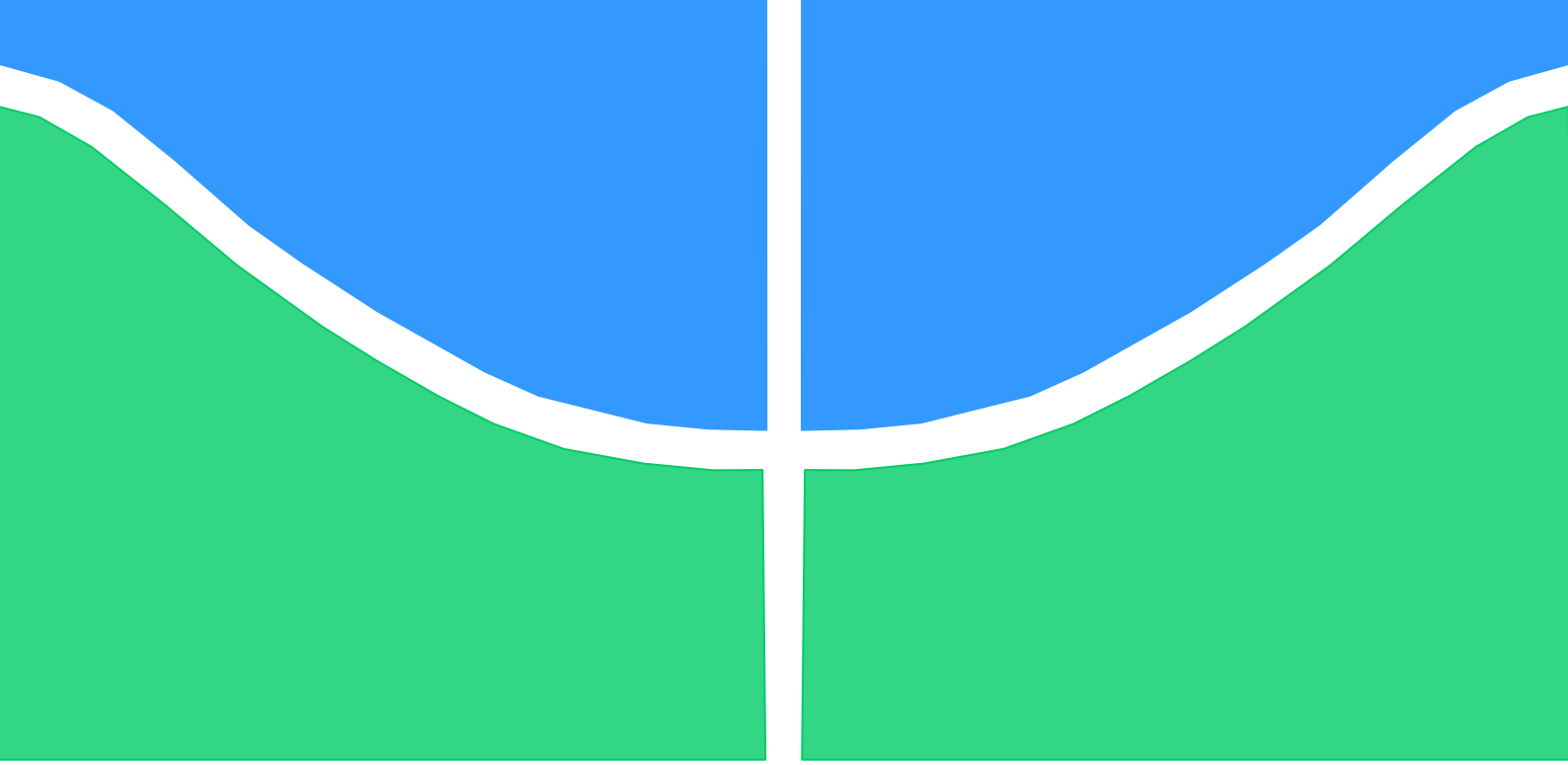


TRABALHO DE CONCLUSÃO DE CURSO

CONSTRUÇÃO DE APARATO EXPERIMENTAL PARA CONTROLE DE SISTEMAS TÉRMICOS COM ATRASO DE TRANSPORTE

Por,

Davi Menezes Visintainer

Brasília, Junho de 2017

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia
Curso de Graduação em Engenharia Elétrica

TRABALHO DE CONCLUSÃO DE CURSO

CONSTRUÇÃO DE APARATO EXPERIMENTAL PARA CONTROLE DE SISTEMAS TÉRMICOS COM ATRASO DE TRANSPORTE

POR,

Davi Menezes Visintainer

Relatório submetido como requisito parcial para obtenção
do grau de Engenheiro Eletricista.

Banca Examinadora

Prof. Dr. Eng. Eduardo Stockler Tognetti, UnB/ ENE (Orientador)	
Prof. Dr. Eng. Adolfo Bauchspiess, UnB/ ENE (Coorientador)	
Prof. Dr. Eng. Lélio Ribeiro Soares Junior, UnB/ ENE	

Brasília, Junho de 2017

FICHA CATALOGRÁFICA

DAVI, MENZES VISINTAINER
CONSTRUÇÃO DE APARATO EXPERIMENTAL PARA CONTROLE DE SISTEMAS TÉRMICOS COM ATRASO DE TRANSPORTE,

[Distrito Federal] 2017.

xvii, 123p., 297 mm (FT/UnB, Engenheiro, Eletricista, 2017). Trabalho de Conclusão de Curso – Universidade de Brasília. Faculdade de Tecnologia.

I. Elétrica/FT/UnB

REFERÊNCIA BIBLIOGRÁFICA

VISINTAINER, D. M., (2017). Construção De Aparato Experimental Para Controle De Sistemas Térmicos Com Atraso De Transporte. Trabalho de Graduação em Engenharia Elétrica, Faculdade de Tecnologia, Universidade de Brasília, Brasília, DF.

CESSÃO DE DIREITOS

AUTOR: Davi Menezes Visintainer.

CONSTRUÇÃO DE APARATO EXPERIMENTAL PARA CONTROLE DE SISTEMAS TÉRMICOS COM ATRASO DE TRANSPORTE: Texto escrito do projeto da construção de um aparato experimental para ensaios de técnicas de controle com atraso de transporte.

GRAU: Engenheiro Eletricista ANO: 2017

Davi Menezes Visintainer
SQN 113 Bloco D ap 404 – Asa Norte.
70.763-040 Brasília – DF – Brasil.

AGRADECIMENTOS

“Quando passares pelas águas estarei contigo, e quando pelos rios, eles não te submergirão; quando passares pelo fogo, não te queimarás, nem a chama arderá em ti. Porque eu sou o Senhor teu Deus, o Santo de Israel, o teu Salvador; ”
Isaías 43:2,3b

Dedico e agradeço à Deus pelas oportunidades que tive, pelos professores que me ajudaram e ensinaram durante o curso de engenharia elétrica, pelos colegas que também me ajudaram durante esse curso. Agradeço à Deus pela família que tenho e me apoiou em todos os momentos desse curso, que me motivaram e opiaram.

Davi M. Visintainer.

RESUMO

Em processos de controle industriais, o fenômeno de transporte de calor está presente e saber identifica-lo para obter um modelo matemático é importante para o desenvolvimento de um projeto de controle dinâmico. Assim, este trabalho aborda o contexto da criação, desenvolvimento, identificação, modelagem e controle de um aparto experimental para controle de sistemas térmicos com atraso no transporte. Foi construída uma planta de controle com um tubo metálico de dois metros de comprimento onde o ar aquecimento por uma resistência elétrica passa em seu interior. A eletrônica envolvida na construção do aparato envolve o uso de TRIAC, para o controle da resistência elétrica, de uma ponte H, para a amplificação do sinal PWM para um motor DC e do microcontrolador Atmega 328 do Arduino Uno para fazer a *interface* entre a planta de controle e um computador. No computador, os sinais de acionamento eram gerados pelo *software* MATLAB em conjunto com as leituras dos sensores. No mesmo *software* foram feitas as análises gráficas dos resultados obtidos, a identificação do modelo matemático no domínio da frequência e a criação do controlador PID. Para isso, usamos *toolbox* que o MATLAB disponibiliza especificamente para projetos de controle dinâmico. Foi projetado um controlador do tipo PID (Proporcional – Integrador – Derivador) em série com o modelo do sistema. Foi feita uma realimentação com ganho unitário negativo para o fechamento da malha de controle.

Palavras-chave: sistema térmico, atraso de transporte, aparato experimental, TRIAC, controle dinâmico, identificação, modelagem.

ABSTRACT

In industrial control processes, the phenomenon of heat transport is present and knowing how to identify it to obtain a mathematical model is important for the development of a dynamic control project. Thus, this work addresses the context of the creation, development, identification, modeling and control of an experimental unit for the control of thermal systems with delay in transportation. A control plant was built with a metal tube with two meters long where air heating by an electric resistance passes inside. The electronics involved in the construction of the apparatus involves the use of TRIAC, for the control of electrical resistance, an H-bridge, for the amplification of the PWM signal for a DC motor and the Atmega 328 microcontroller of the Arduino Uno to interface the plant and a computer. On the computer, the drive signals were generated by the MATLAB software in conjunction with the sensor readings. In the same software, the graphical analysis of the results obtained, the identification of the mathematical model in the frequency domain and the creation of the PID controller were made. For this, we use a toolbox that MATLAB makes available specifically for dynamic control projects. A PID-type controller has been designed in series with the system model. A negative feedback gain was obtained for the control loop closure.

Keywords: thermal system, transport delay, experimental apparatus, TRIAC, dynamic control, identification, modeling.

SUMÁRIO

CAPÍTULO 1 – INTRODUÇÃO 1
1.1 MOTIVAÇÃO 1
1.2 OBJETIVOS 2
1.2.1 OBJETIVOS ESPECÍFICOS 3
1.3 COMPOSIÇÃO E ESTRUTURA DO TRABALHO. 3
CAPÍTULO 2 – DESCRIAÇÃO DO APARATO EXPERIMENTAL 5
2.1 APRESENTAÇÃO DO PROJETO 5
2.2 CIRCUITOS ELETRÔNICOS E INSTRUMENTAÇÃO 7
2.2.1 REGULADOR DE TENSÃO E PONTE H 7
2.2.2 DETECTOR DE ZEROS 8
2.2.3 CIRCUITO DE DISPARO 10
2.2.4 ATUADORES 14
2.2.5 SENSOR DE TEMPERATURA 17
CAPÍTULO 3 – FUNDAMENTOS DO PROCESSO DE TRANSMISSÃO DE CALOR 24
3.1 SISTEMAS TÉRMICOS 24
3.2 SISTEMA FLUÍDICO 26
3.3 PROPOSTA DE IDENTIFICAÇÃO 28
CAPÍTULO 4 – IDENTIFICAÇÃO E MODELAGEM 30
4.1 ENSAIO COM SINAL PSEUDOALEATÓRIO – PRBS 31
4.2 MODELO MATEMÁTICO 34
4.3 VALIDAÇÃO DO MODELO 38
CAPÍTULO 5 – PROJETO DE CONTROLE DINÂMICO 43
5.1 CONTROLADOR PID 44
5.2 CÁLCULO DO SINAL DE ERRO DO SISTEMA 47
5.3 RESULTADO DO CONTROLADOR 47
CAPÍTULO 6 – CONCLUSÃO 53
6.1 TRABALHOS FUTUROS 54
REFERÊNCIAS BIBLIOGRÁFICAS 55
APÊNDICES 56
Apêndice A 56
Apêndice B 60
Apêndice C 64
Apêndice D 67
Apêndice E 69

LISTA DE FIGURAS

Figura 2.1 – Diagrama do aparato experimental....6
Figura 2.2 – Bancada com o aparato experimental....6
Figura 2.3 – (a) Regulador de tensão LM317. (b) *Drive* ponte H L298N. [3]....7
Figura 2.4 – Ponte H utilizada pelo drive L298N [*Datasheet* – fabricante]....8
Figura 2.5 – *Zero crossing*....8
Figura 2.6 – (a) Fotoacoplador PIC817. (b) Conexão dos circuitos e formato do sinal de saída [*Datasheet* – fabricante]....9
Figura 2.7 – Inversão de semiciclo negativo pela ponte retificadora [3]....10
Figura 2.8 – Circuito de disparo do TRIAC....11
Figura 2.9 – (a) Especificações do TRIAC BTA16. (b) Especificações do MOC3021 [*Datasheet* – fabricante]....11
Figura 2.10 – Ondas referentes ao controle de ângulo de disparo [3]....12
Figura 2.11 – (a) Relação entre tensão RMS e o ângulo de disparo na carga. (b) Aproximação linear para a relação entre o ângulo de disparo e a tensão aplicada na carga....14
Figura 2.12 – (a) Resistência do secador de cabelo. (b) Detalhe das conexões das três resistências internas do secador de cabelo....15
Figura 2.13 – Mecanismo eletromecânico de proteção....15
Figura 2.14 – (a) Motor DC. (b) Detalhe das hélices acopladas no eixo do motor DC....16
Figura 2.15 – Suporte para conexão da resistência e motor DC no tubo metálico....16
Figura 2.16 – (a) Chaves seletoras em detalhes. (b) Chaves seletoras....17
Figura 2.17 – Sensor DS18B20 [4]....18
Figura 2.18 – Estrutura interna do sensor DS18B20....19
Figura 2.19 – (a) Sensor de entrada. (b) Sensor de saída. (c) Tubo metálico com os sensores instalados....20
Figura 2.10 – Primeiro teste de funcionamento da plana de controle....21
Figura 3.1 – Representação esquemática de uma resistência térmica [5]....24
Figura 3.2 – Tubulação com escoamento em regime permanente [5]....26
Figura 3.3 – Representação de uma resistência fluídica [5]....26
Figura 4.1 – Sinal pseudoaleatório....29
Figura 4.2 – Resultado do ensaio com o sinal Pseudoaleatório....31
Figura 4.3 – Ensaio com o sinal Pseudoaleatório com a remoção dos Outliers....32
Figura 4.4 – System Identification....34
Figura 4.5 – Resultado da temperatura de entrada esperada em função da variação da tensão de carga aplicada na resistência....35

Figura 4.6 – Resultado da temperatura de saída esperada em função da variação da tensão de carga aplicada na resistência 36
Figura 4.7 – Diagrama de blocos do modelo em malha aberta.....36
Figura 4.8 – Comparativo entre as curvas da tensão aplicada na resistência, das temperaturas de entrada e ambiente e da temperatura de entrada esperada do sistema 38
Figura 4.9 – Comparativo entre as curvas da tensão aplicada na resistência, das temperaturas de saída e ambiente e da temperatura de saída esperada do sistema 39
Figura 4.10 – Teste com a amplitude reduzida.....40
Figura 4.11 – Comparativo com amplitude reduzida entre o valor da temperatura enviada pelo sensor de entrada e a temperatura de entrada esperada através da função de transferência. 41
Figura 4.12 – Comparativo com amplitude reduzida entre o valor da temperatura enviada pelo sensor de saída e a temperatura de saída esperada através da função de transferência. 41
Figura 5.1 – Diagrama de blocos para o projeto em malha fechada.....43
Figura 5.2 – Diagrama de blocos de malha fechada com PID(s) 44
Figura 5.3 – Sisotool 45
Figura 5.4 – (a) Plano *s* apresentado pelo *Sisotool*.(b) Diagrama de Bode.....46
Figura 5.5 – Teste em malha fechada com 1 resistência.....48
Figura 5.6 – Análise do sinal enviado ao Arduino 49
Figura 5.7 – Teste de malha fechada 50
Figura 5.8 – (a) Sinal enviado ao Arduino com a planta funcionando com duas resistências. (b) Detalhe do sinal enviado ao Arduino nos instantes em que o Erro oscila em torno do zero 51

CAPÍTULO 1 – INTRODUÇÃO

1.1 MOTIVAÇÃO

Sistemas de transferência térmica são aqueles que envolvem a transferência de calor de um meio para outro em um dado intervalo de tempo até que a temperatura dos meios se igualem. Na indústria há diversas situações onde a transferência de calor é a operação base para diversos processos e a julgar que desde a primeira revolução industrial, as máquinas térmicas promoveram grandes impactos no avanço da ciência, os processos térmicos formam a principal base da indústria.

Com o avanço da indústria e a busca pela eficiência na produção, o controle dos processos térmicos se tornou necessário. Dessa forma, o primeiro trabalho significativo, na área de controle dinâmico, foi o trabalho realizado por James Watt, no século XVIII, com a construção de um controlador centrífugo para o controle de velocidade em uma máquina a vapor.

Com o avanço da ciência e da tecnologia, atualmente é possível desenvolver trabalhos de controle para sistemas térmicos mais precisos, todavia, novos problemas surgem, conforme os recursos disponíveis para controle dinâmico mudam. Segundo Katsuhiko Ogata [1], os sistemas térmicos podem ser representados por meio de resistências e capacitâncias, mas adverte que esse tipo de representação não possui precisão em parâmetros concentrados, uma vez que o calor é dissipado no ambiente e a melhor forma para isso seria utilizar parâmetros distribuídos. Porém, para se mensurar a temperatura, utilizamos sensores térmicos que medem a temperatura pontualmente, ou seja, onde os sensores são instalados. Assim, para uma análise distribuída de um sistema é necessário utilizar diversos sensores ao longo do sistema de forma que o gradiente de temperatura entre os sensores seja o menor possível.

Desenvolver projetos onde a análise seja simplificada à parâmetros concentrados, proporciona a redução de custos de equipamentos, com o uso de menos sensores. Analisar computacionalmente diversos sensores em tempo real exige que a eletrônica envolvida seja mais sofisticada, aumentando o custo desse tipo de projeto. Dessa forma, é desejável desenvolver projetos de sistemas térmicos que possam fornecer resultados consistentes de forma mais simplificada e com custos reduzidos.

Em alguns processos, como fornalhas e turbinas por exemplo, o processo de aquecimento pode acontecer em um ambiente diferente do qual se deseja aquecer, ou seja, nesses processos há o aquecimento de um fluído, que por sua vez tem a função de transportar o calor até outro ambiente. Há presente na literatura de controle dinâmico a apresentação desse tipo de sistema como um sistema que possui um tempo morto, contudo essa representação pode causar instabilidade até mesmo para sistemas de primeira ordem, pois o lugar geométrico das

raízes entra no semiplano direito do plano s caso o ganho de entrada seja muito alto. Esse tempo morto também é conhecido como atraso no transporte de calor, pois ele acontece em sistemas onde o aquecimento acontece longe do objeto que será aquecido.

Como os sistemas térmicos, que possuem um tempo morto, decorrente do transporte de calor necessitam deslocar algum fluído aquecido ao longo de um meio, esses sistemas estão sujeitos a perturbações como a perda de calor durante o processo. Entretanto, é desejável controlar esses sistemas de tal modo que se tenha no ambiente desejado uma temperatura específica, logo é preciso analisar as técnicas de controle dinâmico para esses sistemas, pois, dependendo do sistema, essas perturbações não são desejadas e a precisão é uma necessidade.

Logo, é preciso que haja o estudo de sistemas térmicos para compreendê-los e procurar aprimoramentos em técnicas de controle dinâmico. Diante do avanço tecnológico, que permitiu a maior compreensão de como funcionava as máquinas térmicas, até o presente momento, onde se tem ciência do atraso no transporte do calor, é preciso que a análise desse tipo de sistema seja contínua.

1.2 OBJETIVOS

O objetivo desse trabalho é desenvolver um aparato experimental que possa apresentar em seu processo um sistema térmico controlável e que apresente um atraso no transporte. Esse desenvolvimento visa estudar a construção física do sistema, para que possamos ter um sistema com a didática compatível com o conhecimento apresentado na graduação de Engenharia Elétrica, tanto na parte eletrônica, quanto na parte computacional.

Assim, como objetivo, procuramos empregar componentes eletrônicos cujas instruções de uso sejam de fácil acesso e/ou gratuitas. Além disso, procuramos empregar programas computacionais que sejam comuns no ambiente de ensino de engenharia, apesar da gratuidade não existir, pois torna a compreensão, do que foi feito, mais fácil.

Além da construção do aparato experimental, temos como objetivo modelá-lo através de uma função de transferência no domínio da frequência. Juntamente a esse objetivo, desejamos criar um modelo de controle em malha fechada para termos um controle dinâmico da temperatura de saída do aparato experimental.

Assim, ao final do projeto, desejamos ter um aparato experimental para o controle de sistemas térmicos e que apresente atraso no transporte de calor. Junto à isso, desejamos ter um projeto para controlar a temperatura de saída do aparato por meio da manipulação da potência elétrica.

1.2.1 OBJETIVOS ESPECÍFICOS

1.2.1.1 CONSTRUIR APARATO EXPERIMENTAL

- Definir equipamentos e materiais a serem usados no aparato experimental;
- Construir aparato experimental;
- Testar a qualidade e funcionalidade dos materiais e equipamentos empregados e fazer os ajustes necessários;
- Coletar dados para planejar ensaios de identificação;

1.2.1.2 IDENTIFICAR E MODELAR O SISTEMA

- Fazer ensaios de identificação do sistema com sinais pseudoaleatórios;
- Identificar a função de transferência do sistema entre a tensão na carga resistiva (entrada) e o sensor de saída da planta de controle;
- Validar testes de identificação;

1.2.1.3 PROJETAR CONTROLADOR

- Projetar um controlador em malha fechada para controlar a temperatura de saída do tubo metálico;
- Implementar e avaliar controlador em malha fechada.

1.3 COMPOSIÇÃO E ESTRUTURA DO TRABALHO.

O trabalho está dividido da seguinte forma:

1. Introdução: Neste capítulo é introduzido o presente trabalho juntamente com as motivações e objetivos.

2. Descrição do Aparato Experimental: Neste capítulo é apresentado a construção do aparato experimental, que foi utilizado no trabalho. Há a descrição dos circuitos empregados, da escolha do sensor e por fim são apresentados os primeiros testes do projeto para avaliar o funcionamento e como iremos utilizar o aparato experimental.

3. Processo de Transmissão de Calor: Neste capítulo é feita a apresentação dos sistemas físicos que estão envolvidos no projeto, a fim de contextualizar o trabalho em parte física. Este capítulo ajudou também a fornecer as bases teóricas para a modelagem matemática do sistema e assim projetar o sistema de controle.

4. Identificação e Modelagem: Neste capítulo apresentamos os testes com o aparato experimental a fim de criarmos um modelo matemático, no domínio da frequência. Neste capítulo utilizamos recursos computacionais para avaliar o trabalho e os resultados do processo.

5. Projeto de Controle Dinâmico: A partir da identificação e modelagem feitas no capítulo anterior, este capítulo apresenta o projeto de controle em malha fechada utilizando um controlado PID.

CAPÍTULO 2 – DESCRIAÇÃO DO APARATO EXPERIMENTAL

2.1 APRESENTAÇÃO DO PROJETO

Este trabalho visa a construção de um aparato tecnológico para o estudo do controle de temperatura em um sistema sujeito a atrasos no transporte da massa térmica, que chamamos de planta de controle de processo térmico. Em termos técnicos, Katsuhiko Ogata [1] define Planta como “uma parte de equipamento ou apenas um conjunto de componentes de um equipamento que funcione de maneira integrada, com o objetivo de realizar determinada operação”.

A planta de controle é composta pela parte física, ou *hardware*, e a parte digital, ou *software*. A parte de *hardware* é composta pelos atuadores, sensores e toda a eletrônica necessária para soprar o ar e aquece-lo. A parte de *software* é composta pelos códigos, tanto para programar o microcontrolador, quanto para gerar sinais que serão aplicados no sistema e ler os sinais dos sensores presentes na planta de controle.

No Apêndice A temos em anexo os códigos utilizados para programar o microcontrolador Atmega e para processar os dados fornecidos pelo mesmo. Temos dois códigos distintos, pois inicialmente a planta de controle foi construída para trabalhar em malhar aberta, todavia para obtermos um modelo matemático melhor e aplicarmos técnicas de controle, optamos em utilizar o software MATLAB (MathWorks, R2015a) [2].

A comunicação entre o *hardware* e o *software* acontece através de uma porta serial do computador com um cabo USB. Nessa comunicação o Arduino Uno monitora constantemente a porta serial para obter a informação de quantos porcentos da tensão deve ser aplicada no resistor ou no motor e ele também escreve na porta serial, dentro de um intervalo de tempo, os valores lidos nos sensores de temperatura.

Na Figura 2.1, está representado o diagrama com as conexões e disposições dos equipamentos utilizados na construção do aparato experimental, bem como o que cada equipamento envia e recebe. Os componentes mostrados na Fig. 2.1 serão apresentados detalhadamente na seção 2.2.

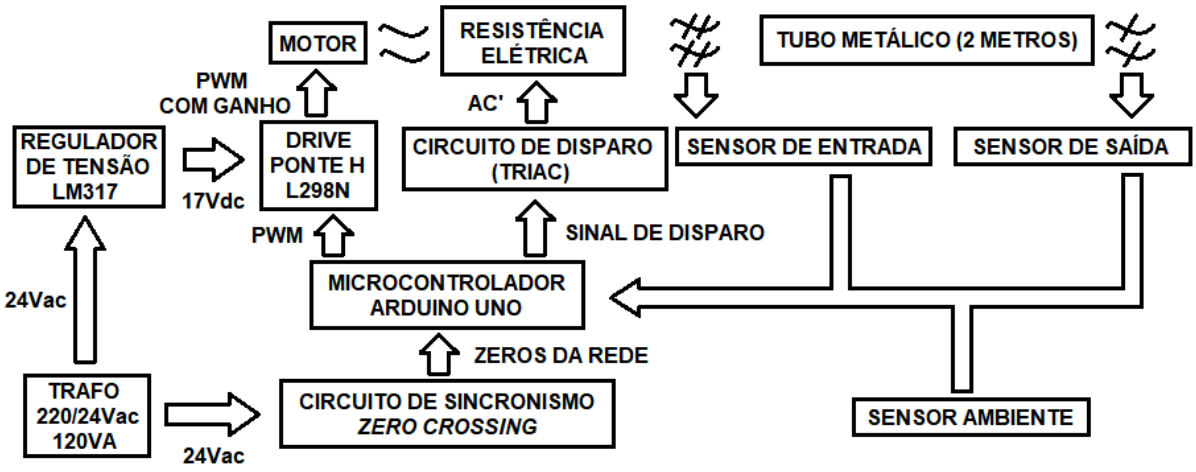


Figura 2.1. Diagrama do aparato experimental.

A montagem dos componentes apresentados no diagrama da Fig. 2.1 foi feita conforme a Fig. 2.2. Na imagem apresentada, todos os componentes foram colocados sobre a bancada, logo os sensores não estão inseridos no tubo metálico, para que assim fosse possível observar melhor os componentes utilizados.

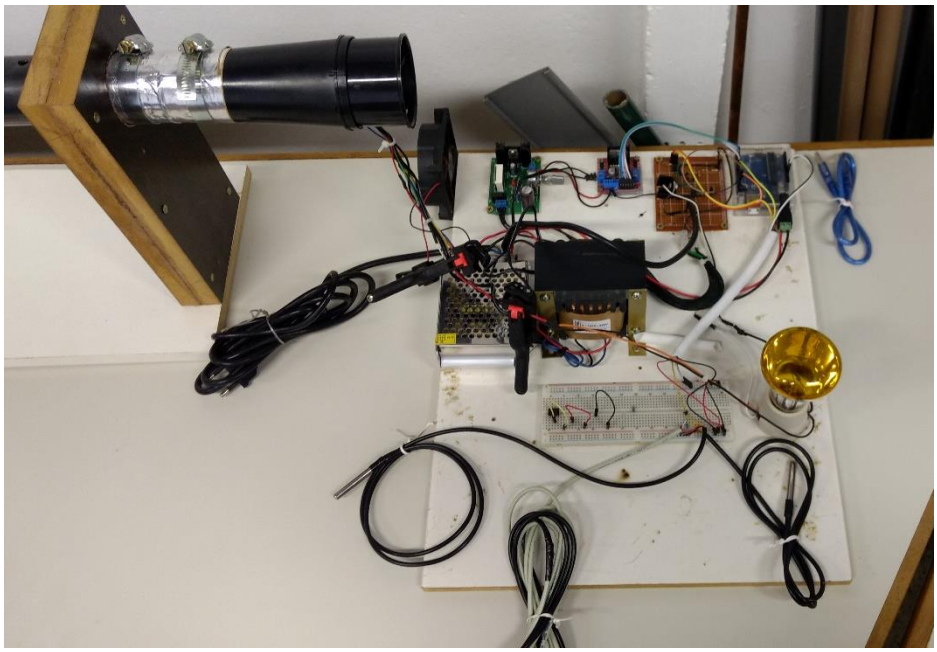


Figura 2.2. Bancada com o aparato experimental.

Além dos equipamentos mostrados no diagrama, na Fig. 2.2 há uma lâmpada resistiva apenas para facilitar a observação do aparato funcionando, também há uma fonte DC com um ventilador para manter os componentes resfriados e garantir a alimentação constante do Arduino.

2.2 CIRCUITOS ELETRÔNICOS E INSTRUMENTAÇÃO

2.2.1 REGULADOR DE TENSÃO E PONTE H

A opção que escolhemos para controlar a velocidade do motor DC é utilizar um sinal PWM, que o Arduino pode oferecer interruptamente enquanto executa outros comandos. Todavia, o valor da tensão do PWM que o Arduino oferece é de 5Vdc e o motor que utilizamos opera em uma faixa de 17Vdc, logo é necessário amplificar o sinal que temos do Arduino para a faixa de tensão que o motor trabalha.

Para amplificar o sinal PWM utilizamos duas placas de circuito impresso que são baseados nos seguintes componentes: o regulador de tensão LM317, apresentado na Fig. 2.3 (a), e o *drive* ponte H L298N, apresentado na Fig. 2.3 (b). Ambos foram obtidos em placas pré-fabricadas com os chips citados prontas para uso a fim de garantir a qualidade construtiva da bancada e evitar falhas como a interrupção dos circuitos por erros em soldagens.

(a) (b)

Figura 2.3. (a) Regulador de tensão LM317. (b) *Drive* ponte H L298N. [3].

Operando em conjunto, o regulador de tensão funciona como uma fonte de bancada, ajustada para fornecer a tensão necessária para o motor funcionar e o drive recebe o sinal PWM do Arduino e amplifica para a tensão fornecida pelo regulador. As pontes H do *drive* podem operar dois motores DC simultaneamente podendo girá-los no sentido horário ou anti-horário, assim foi tomado o cuidado no código de programação do Arduino para que o motor sempre gire no sentido que sopre o ar no tubo. Os circuitos que implementam as pontes H do *drive* usado estão apresentados na Fig. 2.4.

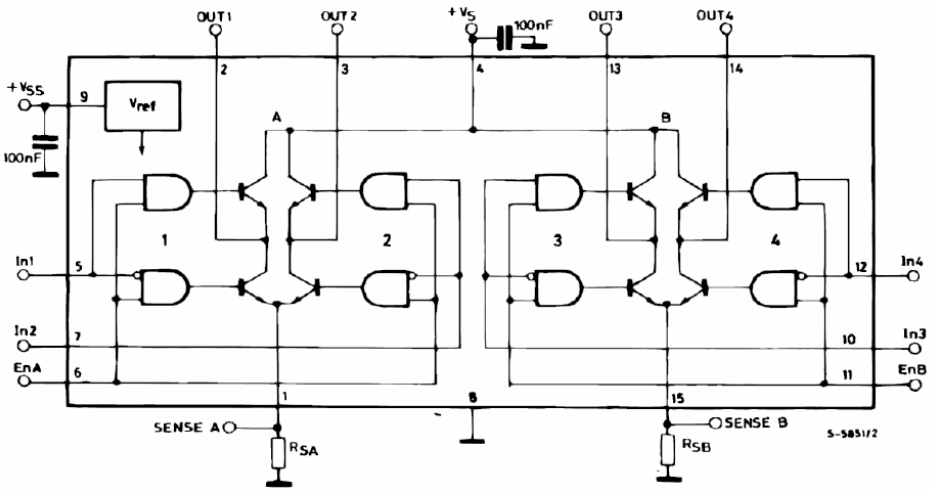


Figura 2.4. Ponte H utilizada pelo drive L298N [*Datasheet* – fabricante].

2.2.2 DETECTOR DE ZEROS

O circuito detector de zeros ou *zero crossing* serve para detectar quando a tensão AC da rede elétrica muda a polaridade, assim, garantirmos que o pulso de disparo do TRIAC seja feito de forma sincronizada à tensão de alimentação. A Fig. 2.5 mostra este módulo de sincronismo, onde temos o circuito de *zero crossing* que é formado por uma ponte completa de diodos, um resistor de 2,7 kΩ, fotoacoplador PIC817 e um resistor de 10kΩ. A tensão senoidal aplicada na ponte de diodos possui 24 volts a 60Hz obtida através de um transformador 220/24 Vac, 120VA para que possamos trabalhar na faixa de tensão do fotoacoplador. Baseado na tensão sobre a ponte de diodos, determinamos o valor da resistência R1 para 2,7 KΩ a fim de garantirmos uma corrente máxima de 9mA na entrada do fotoacoplador.

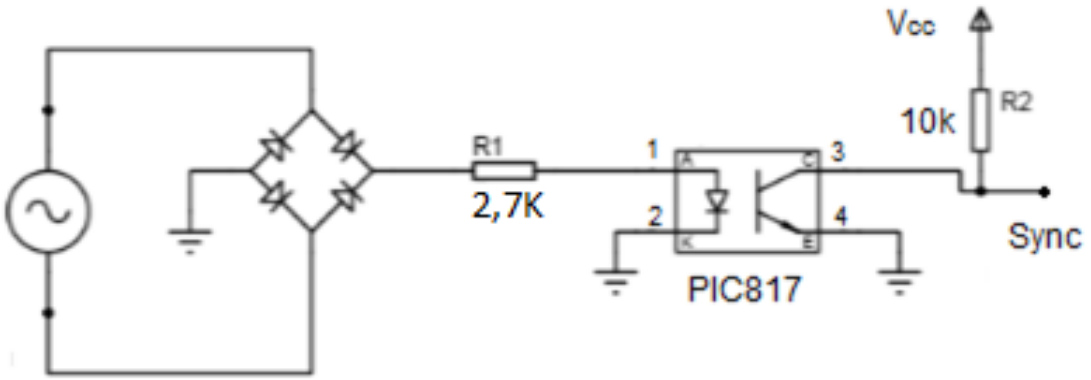


Figura 2.5. *Zero crossing*.

Este sincronismo garante que tenhamos sempre a mesma forma de onda aplicada na carga e assim garante sempre a mesma tensão equivalente sobre a carga, caso não houvesse esse módulo, não saberíamos em que momento que se deveria disparar o TRIAC e assim não poderíamos garantir o valor constante da tensão entregue a carga.

O componente PIC817, também pode ser encontrado como PC817, é chamado de fotoacoplador e a sua função é impedir que exista um contato elétrico entre dois circuitos. No

caso do nosso circuito de detecção de zeros, o fotoacoplador impede o contato do circuito em 24Vac com a porta do microcontrolador para que não ocorra danos físicos a ele. Na Figura 2.6 (a), é possível observar a estrutura do fotoacoplador, onde não há contato físico entre o circuito conectado nos terminais *Anode* e *Cathode* e com o circuito conectado nos terminais *Emitter* e *Collector*. A única informação que passa de um circuito para o outro são os instantes em que a tensão em 24Vac cruza os zeros.

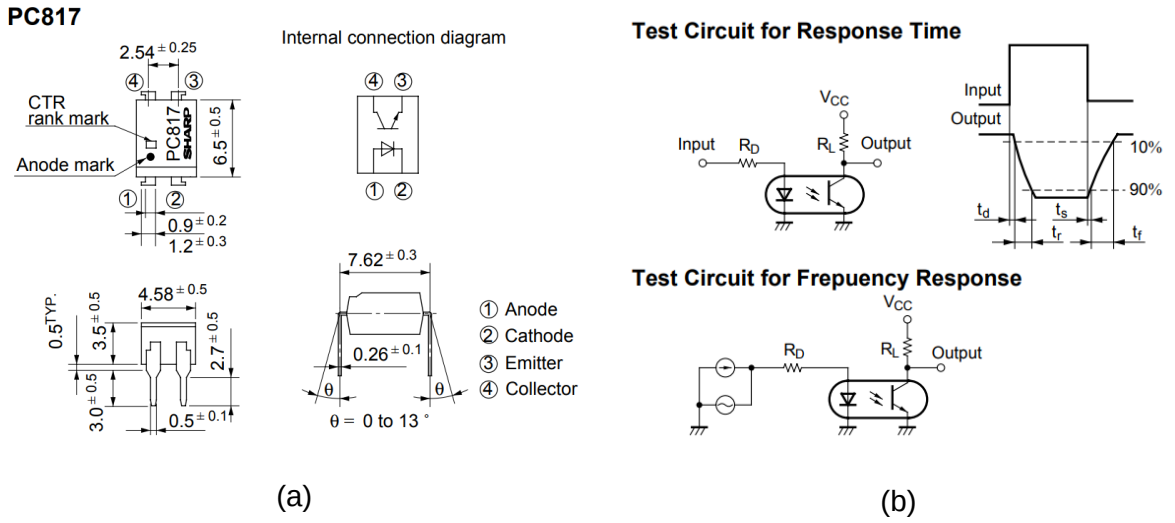


Figura 2.6. (a) Fotoacoplador PIC817. (b) Conexão dos circuitos e formato do sinal de saída [*Datasheet* – fabricante].

Conforme a Fig. 2.6 (b), podemos observar a orientação do fabricante para a utilização deste componente. O que se deve destacar disso é que o sinal resultante não será uma onda triangular, mas deverá apresentar um formato trapezoidal. Todavia, como a informação que desejamos é os instantes em que este sinal acontece, o formato da onda de saída do componente não será relevante, mas sim a periodicidade com que ele ocorre.

Ao construir a ponte retificadora de onda completa, que está apresentada na Fig. 2.5, optamos em utilizar 4 diodos do tipo 1n4001. É possível adquirir um componente só que faz exatamente o que a ponte de retificação de onda completa, que construímos, faz. Porém a opção de construir a ponte utilizando 4 diodos foi feita para que outras pessoas que analisem o circuito possam entender a finalidade da ponte retificadora.

Na grande maioria das aplicações da ponte retificadora de onda completa, o objetivo de utiliza-la é construir fontes de tensão DC, que retificam a tensão AC para DC. Todavia o objetivo da nossa ponte retificadora de onda completa não é esse, mas sim passarmos os semiciclos negativos do sinal AC para um sinal positivo. Em outras palavras, desejamos transformar a tensão AC em valores positivos, conforme é mostrado na Fig. 2.7, todavia, não desejamos tratar esse sinal com capacitores e outros componentes, como é feito em fontes DC.

Desejamos fazer isso para que a ponte de onda completa junto com um fotoacoplador forneçam ao Arduino pulsos toda vez que acontece uma inversão da tensão fornecida e como

estamos trabalhando com uma alimentação de 220V fase neutro, a 60Hz, teremos um sinal cujo módulo está em 120Hz. O esperado para a ponte retificadora de onda completa é mostrado na Fig. 2.7, onde os semiciclos negativos do sinal original fornecido pela rede ($v_s = V_m \sin \omega t$) são invertidos e passam a se comportarem como semiciclos positivos e assim podemos reforçar o resultado esperado de termos 120Hz como frequência de entrada no nosso microcontrolador.

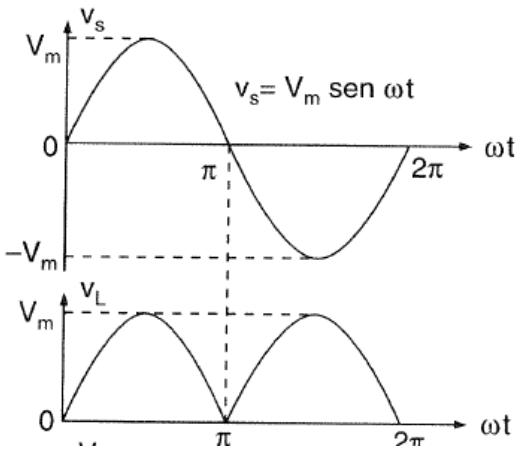


Figura 2.7. Inversão de semiciclo negativo pela ponte retificadora [4].

Outro detalhe importante é que existem oscilações na tensão que é fornecida pela rede de alimentação através de inconstâncias na sua frequência e por mais confiável que possa ser o fornecedor, haverá pequenas variações na frequência entregue. Isso acontece devido ao fato que a todo momento cargas são conectadas e desconectadas da rede causando variações imprevistas ao fornecedor e como a geração de energia é feita através de máquinas síncronas, essas alternâncias na rede causam pequenas oscilações da frequência síncrona do sistema.

Assim, o módulo *zero crossing* é importante para que o efeito das perturbações não atrapalhe o controle do ângulo de disparo do TRIAC, pois se mantivéssemos os pulos constante depois de constatado a primeira passagem pelo zero da tensão da rede, o TRIAC não permaneceria em sincronismo e a tensão na nossa resistência não seria constante.

2.2.3 CIRCUITO DE DISPARO

O outro circuito do aparato experimental é mostrado na Fig. 2.8, onde temos o circuito de disparo do TRIAC com os componentes responsáveis para que ele possa ser disparado corretamente. O TRIAC que usamos nesse circuito é o BTA16 e ele foi instalado junto com um dissipador de calor.

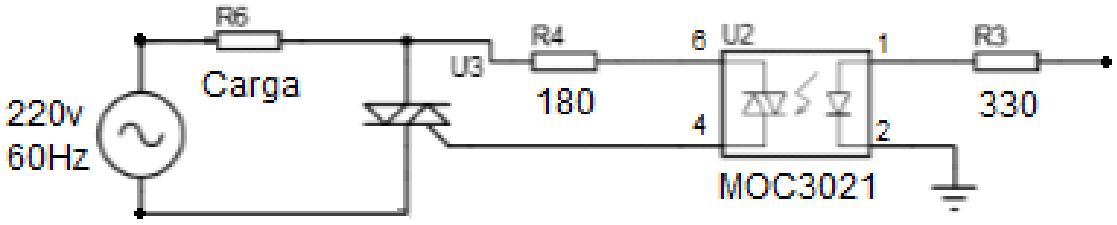
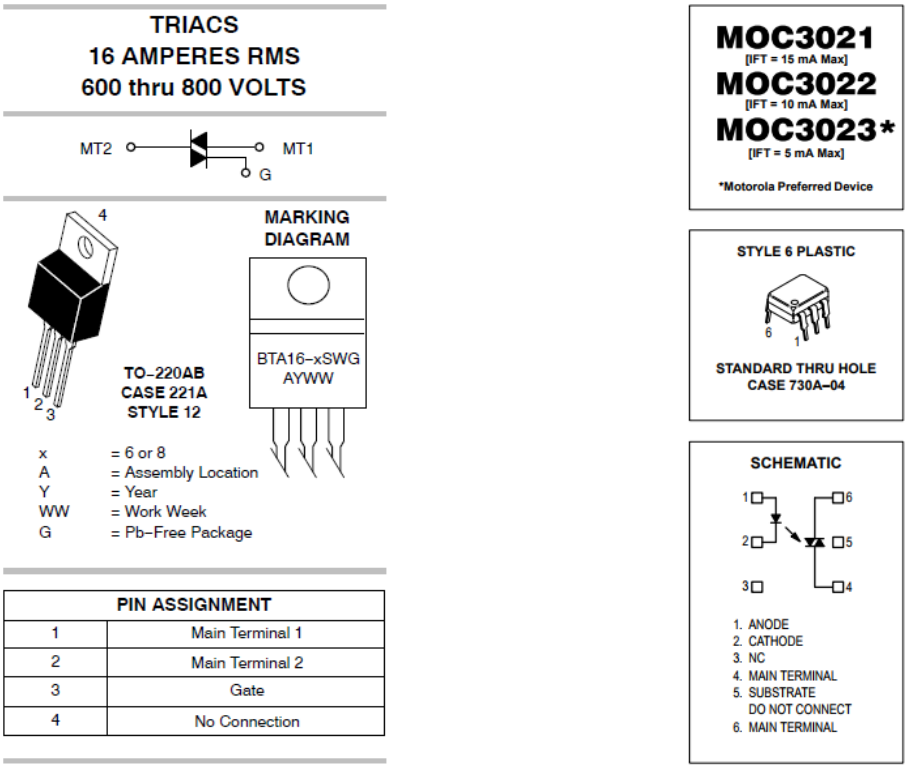


Figura 2.8. Circuito de disparo do TRIAC.

Além do BTA16, o circuito da Fig. 2.8 possui um acoplador óptico baseado em um TRIAC também, designado por MOC3021, porém sem um *gate* para acionamento. As especificações técnicas desses componentes estão apresentadas na Fig. 2.8, sendo que o TRIAC BTA16, para o acionamento das cargas, está apresentado na Fig. 2.8 (a), e o MOC3021, está apresentado na Fig. 2.8 (b).



(a) (b)

Figura 2.9. (a) Especificações do TRIAC BTA16. (b) Especificações do MOC3021 [*Datasheet* – fabricante].

Existem circuitos impressos vendidos que já oferecem a função do circuito apresentado na Fig. 2.8, porém a nossa resistência elétrica demanda uma corrente na faixa de 9A. Dessa forma, os componentes prontos não vendidos não atendem a nossa demanda.

A partir disso, escolhemos os componentes apresentados na Fig. 2.9 e é possível observar nas especificações que a corrente máxima suportada pelo BTA16 e pelo MOC3021 é de 16A e 15A, respectivamente.

A construção deste circuito é orientada no datasheet do próprio MOC3021 para controle de cargas puramente resistivas. Em cargas indutivas é orientado o uso de um capacitor entre um dos terminais do TRIAC e o *gate* devido aos reativos que circulam pelo circuito, porém as cargas que estão ligadas ao BTA16 são resistivas. A princípio, a única carga que seria controlada por esse circuito seria a resistência elétrica para dissipar calor, contudo, colocamos uma lâmpada resistiva em paralelo para podermos visualizar quando o circuito estivesse atuando com uma noção da intensidade de potência dissipada no resistor.

O circuito de disparo é controlado pelo Arduino através de uma porta de comunicação analógica, que envia um pulso para o MOC3021 no momento em que o TRIAC for fazer o controle do ângulo da senoide. Os pulsos são determinados pelo microcontrolador através dos pulsos, por ele recebidos do circuito detector de zeros, considerando a porcentagem da senoide que se deseja aplicar nas nossas cargas.

Podemos calcular qual é a tensão equivalente aplicada no nosso resistor sabendo qual foi o ângulo de disparo do TRIAC e conhecendo a tensão e frequências fornecidas pela rede. O tipo de circuito de controle de tensão que estamos aplicando com o TRIAC é conhecido como Conversor Monofásico Semicontrolado [4]. Na Fig. 2.10 podemos observar graficamente como o TRIAC atua sobre uma tensão AC ($v_s = V_m \sin \omega t$) em que dois pulsos i_{T1} e i_{T2} são aplicados nele resultando numa tensão V_0 sobre as cargas que estão sendo controladas.

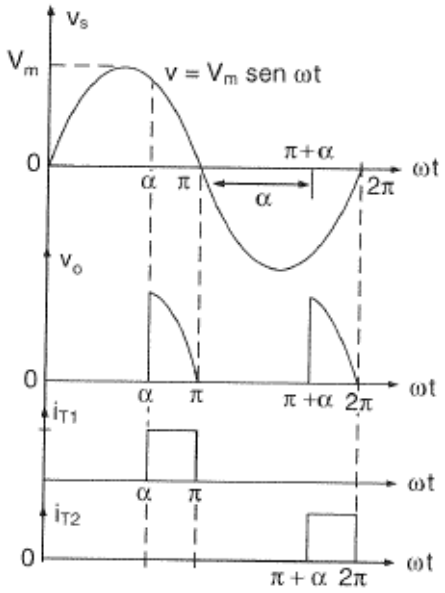


Figura 2.10. Ondas referentes ao controle de ângulo de disparo [4].

Podemos calcular qual o valor médio da tensão V_0, mostrado na Fig. 2.10. Primeiramente consideramos a função que define a tensão de entrada como:

$$v = V_m \sin \omega t \qquad (1)$$

Durante o semiciclo positivo dessa tensão de entrada, o TRIAC estará diretamente polarizado e será disparado, para conduzir corrente quando:

$$\omega t = \alpha \quad (2)$$

O TRIAC irá conduzir do instante em que a tensão da rede cruzar o zero, ou seja:

$$\alpha \leq \omega t \leq \pi \quad (3)$$

Durante o semiciclo negativo, o mesmo processo acontecerá, lembrando que o TRIAC é um dispositivo que pode conduzir tanto no semiciclo negativo quanto no positivo. No semiciclo negativo a condução ocorrerá no instante:

$$\omega t = (\pi + \alpha) \quad (4)$$

Analisando a forma de onda resultante entregue à carga, podemos calcular o valor médio da tensão V_0 através da integração da forma de onda resultante. Assim, a tensão média que é fornecida é dada pela expressão:

$$V_{médio} = \frac{1}{2\pi}\int_{\alpha}^{\pi} V_m \sin \omega t \, d(\omega t) + \frac{1}{2\pi}\int_{\alpha+\pi}^{2\pi} V_m \sin \omega t \, d(\omega t) \quad (5)$$

Podemos reescrever da seguinte forma:

$$V_{médio} = \frac{2}{2\pi}\int_{\alpha}^{\pi} V_m \sin \omega t \, d(\omega t) \quad (6)$$

$$V_{médio} = \frac{V_m}{\pi}[-\cos \omega t]_{\alpha}^{\pi} \quad (7)$$

$$V_{médio} = \frac{V_m}{\pi}[1 + \cos \alpha] \quad (8)$$

Contudo, é mais interessante que tenhamos o valor da tensão eficaz que o TRIAC pode fornecer à nossa carga. Assim, o cálculo da expressão do valor eficaz é:

$$V_{RMS} = \sqrt{\frac{1}{2\pi}\int_{\alpha}^{\pi} {V_m}^2 (\sin \omega t)^2 \, d(\omega t) + \frac{1}{2\pi}\int_{\alpha+\pi}^{2\pi} {V_m}^2 (\sin \omega t)^2 \, d(\omega t)} \quad (9)$$

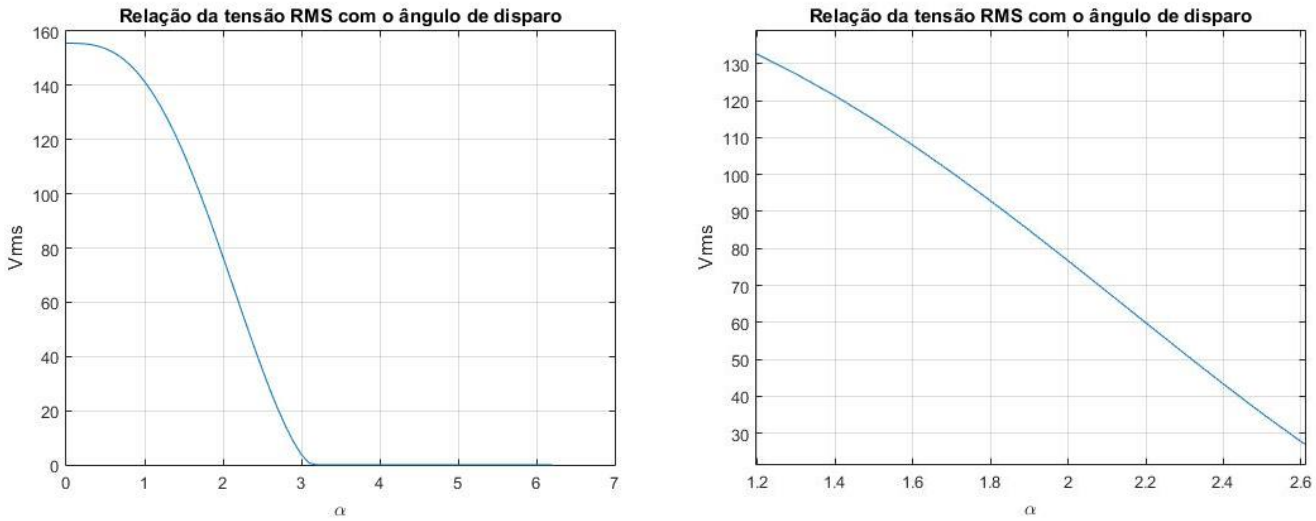
$$V_{RMS} = \sqrt{\frac{{V_m}^2}{2\pi}\int_{\alpha}^{\pi} (1 + \cos 2\omega t) \, d(\omega t)} \quad (10)$$

$$V_{RMS} = \frac{V_m}{\sqrt{2\pi}}\sqrt{\pi - \alpha + \frac{\sin 2\alpha}{2}} \quad (11)$$

Dessa forma, podemos confirmar que estamos controlando a potência elétrica que está sendo dissipada no sistema através do efeito Joule. É fundamental termos a ciência de quais parâmetros estamos alterando do sistema para controlar a temperatura final, pois podemos avaliar a eficiência energética que temos e garantir a robustez do sistema.

Como mostra a Eq. 11, a tensão eficaz que estamos controlando depende da amplitude da tensão da rede, que manteremos constante, e o valor do ângulo de disparo α imposto ao TRIAC. A partir disso, podemos mensurar no sistema real como a temperatura se comportará através das tensões aplicadas no resistor.

Devemos destacar que a tensão RMS aplicada na carga e o ângulo de disparo não possuem uma relação linear. Isso pode ser observado na Fig. 2.11 (a), em que quanto maior for o ângulo de disparo, menor será a tensão aplicada na carga. Diante a este fato, é importante trabalhar uma faixa de valores do ângulo de disparo onde a relação entre esse parâmetro e a tensão RMS possam se aproximar da linearidade, conforme mostrado na Fig. 2.11 (b). Ou seja, não devemos trabalhar com valores de tensão RMS que correspondam próximos à 100% da tensão nominal (220V) nem com valores próximos à 0% da tensão nominal.



(a) (b)

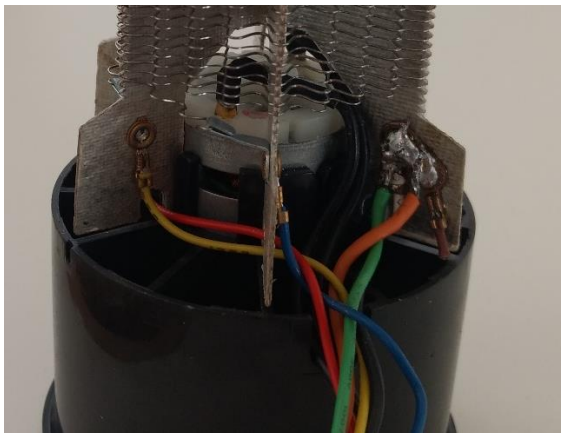
Figura 2.11. (a) Relação entre tensão RMS e o ângulo de disparo na carga. (b) Aproximação linear para a relação entre o ângulo de disparo e a tensão aplicada na carga.

2.2.4 ATUADORES

Na construção do aparato experimental, foi utilizado um secador de cabelo cujo o fabricante informa que possui 2000 watts de potência. Por se tratar de um equipamento com essa grandeza de potência, a corrente de operação do secador de cabelo deve ser na faixa de 9A, quando acionado na operação máxima. O secador foi desmontado para desmembrar os circuitos da resistência elétrica do circuito do motor.

A resistência elétrica do secador, mostrada na Fig. 2.12 (a), é composta por 3 resistências internas, conforme é possível observar na Fig. 2.12 (b). Cada resistência pode ser ligada em paralelo uma com a outra de forma a reduzir a resistência equivalente do circuito e assim a resistência equivalente pode variar entre 127Ω, 63Ω e 32Ω. A ideia dessa operação é aumentar a corrente que vai alimentar o conjunto de resistência e assim aumentar a potência dissipada.

(a)



(b)

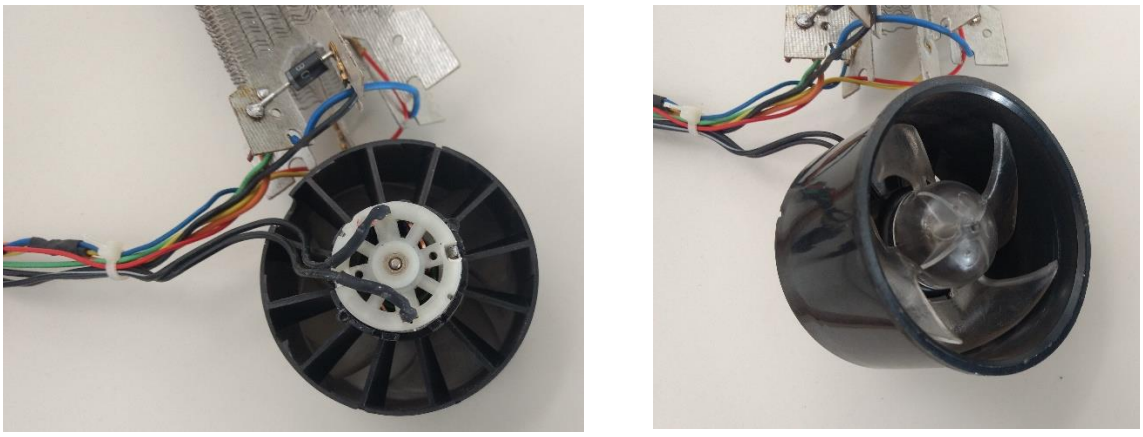
Figura 2.12. (a) Resistência do secador de cabelo. (b) Detalhe das conexões das três resistências internas do secador de cabelo.

A estrutura da resistência do secador de cabelo vem com um mecanismo eletromecânico de fabrico para proteger a resistência elétrica do superaquecimento, conforme mostrado na Fig. 2.13. Este mecanismo não foi removido para mantermos a segurança do aparato experimento, uma vez que ao desmontarmos o secador de cabelo e desmembrarmos os circuitos é possível que haja aquecimento sem que exista o fluxo de ar na resistência, podendo provocar o superaquecimento da mesma.

Figura 2.13. Mecanismo eletromecânico de proteção.

O motor DC, que estava ligado no mesmo circuito do secador de cabelo e está apresentado na Fig. 2.14 (a), era alimentado somente por uma ponte de diodos, sem nenhum outro circuito para melhorar a retificação do sinal. Para estimar qual tensão DC aplicar no motor, foi media a tensão AC sobre a ponte de diodos e calculada a tensão RMS que a ponte aplicava no motor. Considerando os fatos apresentados, foi definida uma tensão menor no motor para a sua proteção.

A estrutura física do motor DC não foi alterada, mantendo assim o suporte original do motor, como apresentado na Fig. 2.14 (a), e a hélice, responsável em soprar o ar na resistência elétrica, conforme apresentada na Fig. 2.14 (b).



(a) (b)

Figura 2.14. (a) Motor DC. (b) Detalhe das hélices acopladas no eixo do motor DC.

Assim, apesar dos circuitos do motor e da resistência elétrica terem sido separados, a disposição física dos componentes não foi alterada. Logo, motor e resistência elétrica, são acoplados, conforme apresentado na Fig. 2.12 (a), e posteriormente encaixados no suporte mostrado na Fig. 2.15. Este suporte aproveitou a estrutura interna do próprio secador de cabelo e o fixou com abraçadeiras metálicas e material de isolação térmica, conforme apresentado.

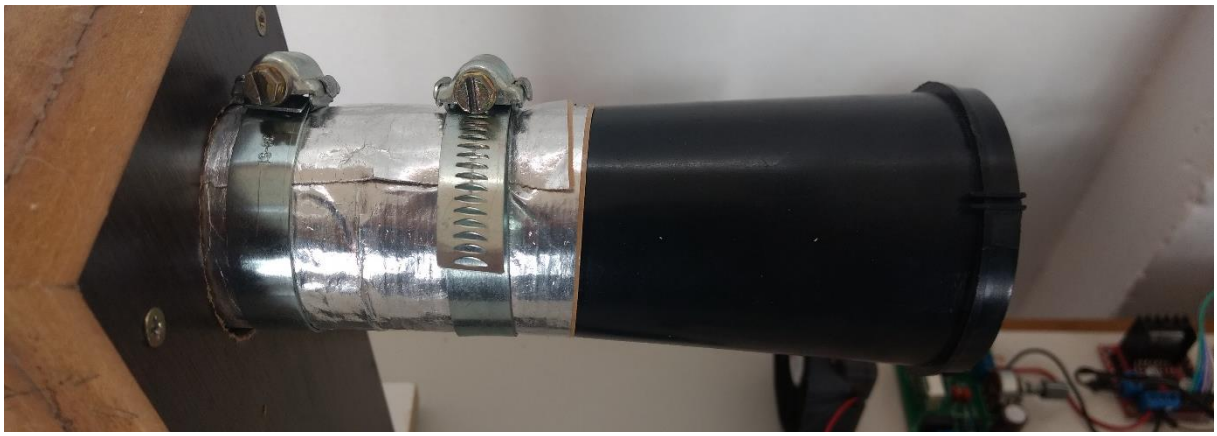


Figura 2.15. Suporte para conexão da elétrica e motor DC no tubo metálico.

Junto com a estrutura do secador de cabelo, utilizado na construção deste aparato, havia duas chaves seletoras que foram aproveitadas, apresentadas na Fig. 2.16. Originalmente a chave da esquerda, na Fig. 2.16 (a) servia para selecionar a temperatura do ar do secador de cabelo e a chave da direita a velocidade do ar soprado. Todavia, ao alterar a velocidade do vendo, do circuito original, isso afetava igualmente a temperatura do secador de cabelo, pois ao reduzi a velocidade pela metade, reduzia-se a tensão do motor DC pela metade e sequencialmente, reduzia-se a tensão da resistência elétrica.

(a) (b)

Figura 2.16. (a) Chaves seletoras em detalhes. (b) Chaves seletoras.

Ao desmembrar os circuitos, resolvemos manter as chaves seletoras, porém elas estão conectadas apenas à resistência elétrica. Essa decisão foi tomada para que possamos alterar fisicamente quantas resistências estarão conectadas em paralelo e se desejamos aplicar apenas a metade da tensão nas resistências, para que possamos avaliar a planta de controle em vários aspectos distintos.

2.2.5 SENSOR DE TEMPERATURA

Inicialmente a planta de controle foi planejada para ser usada com sensores analógicos, especificamente o LM35. Todavia, ao analisar a implementação desse tipo de sensor, foi observado que a distância entre o sensor, no tubo metálico, e o microcontrolador demandava um cabo longo e isso proporcionava uma alta incidência de ruídos de leitura. A solução proposta era a implementação de um circuito capaz de filtrar e amplificar o sinal de cada sensor ou utilizar sensores que possam enviar o sinal através de uma comunicação digital.

A solução mais prática para implementar na planta de controle foi utilizar sensores que possam enviar sinais digitais. O fato de se transmitir um sinal no formato digital não implica que ele não sofrerá distorções devido à ruídos, contudo, na recepção deste sinal no microcontrolador envolve o processo de regeneração. Isto propicia uma taxa de recuperação do sinal original maior, caso fossemos transmitir um sinal analógico.

O sensor escolhido para o aparato experimental foi o DS18B20, apresentado na Fig. 2.17, por ser um sensor térmico que apresenta uma boa faixa de operação para as grandezas de temperatura esperadas da planta de controle. Além disso, o sensor possui uma estrutura

interna que fornece em sua saída um sinal digital baseado o protocolo de transmissão 1-wire™.

Figura 2.17. Sensor DS18B20 [3].

O protocolo 1-wire™ foi desenvolvido pela empresa *Dallas Semiconductor* e permite a comunicação digital entre dispositivos da série 1-wire e computadores. Por definição deste protocolo, a transmissão dos dados dos dispositivos para os leitores é feita através de um único condutor, sendo que por convenção, os condutores de referência não são considerados.

A transmissão de dados, através do protocolo 1-wire™, em um único condutor é feita independentemente do número de dispositivos que possa existir. Ou seja, em nossa planta de controle, os três sensores empregados podem transmitir o sinal digital para a mesma porta do microcontrolador Arduino Uno.

O protocolo de comunicação do sistema 1-wire™ utiliza níveis lógicos convencionais CMOS/TTL, no qual o nível lógico 0 (zero) é representado por uma tensão máxima de 0,8 Vcc e o nível lógico 1 (um) por uma tensão mínima de 2,2 Vcc [DALLAS SEMICONDUCTOR, 2002].

A estrutura interna do sensor DS18B20 é apresentada na Fig. 2.18 em forma de diagrama de blocos. O bloco *"64-BIT ROM AND 1-wire PORT"* armazena um código de série exclusivo para cada sensor. Este código é fundamental para a utilização do protocolo 1-wire™, pois é através deste código que é possível identificar a temperatura especificamente de cada sensor, uma vez que todos estão conectados à mesma porta digital do microcontrolador. A utilização deste protocolo no Arduino, para fazer a leitura dos sensores, é feita através de duas bibliotecas chamadas *OneWire.h* e *DallasTemperature.h*, conforme apresentado no Apêndice A. Para evitar erros de leitura dos bits que são enviados ao microcontrolador, existe uma série de bits que que contêm um controle de redundância cíclica (CRC – *Cyclic Redundancy Check).*

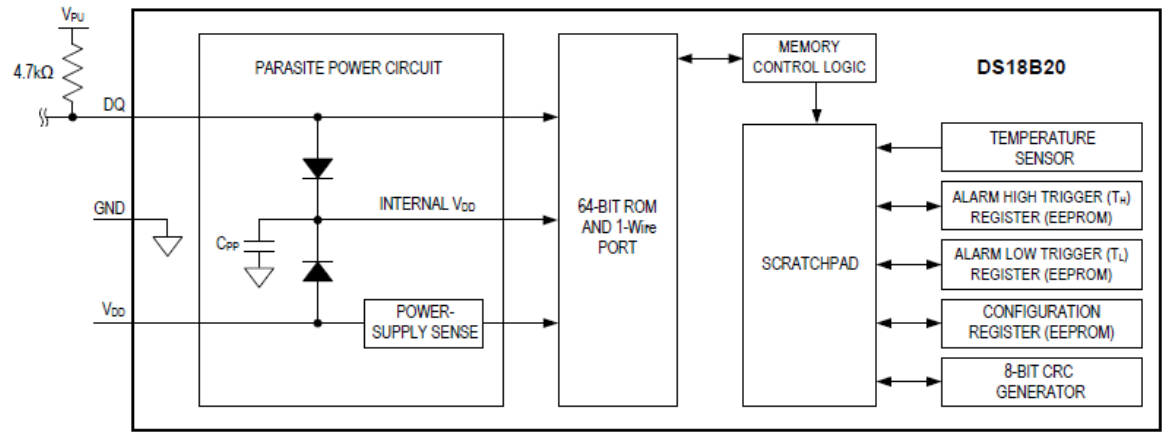


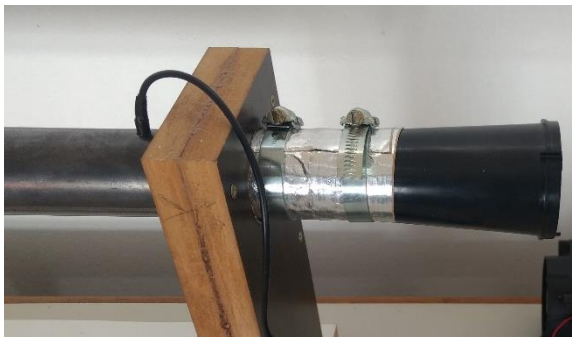
Figura 2.18. Estrutura interna do sensor DS18B20.

No bloco *scratchpad* contém a memória do registrador de temperatura de 2 bytes, que armazena a saída digital do sensor de temperatura. Além disso, o *scratchpad* fornece acesso aos registros de disparo de alarme superior e inferior de 1 byte (TH e TL) e ao registro de configuração de 1 byte. O registro de configuração permite que o usuário defina a resolução da conversão temperatura para digital para 9, 10, 11 ou 12 bits. Importante destacar que os registradores TH, TL e de configuração não são voláteis (EEPROM), portanto, eles manterão dados quando o dispositivo estiver desligado.

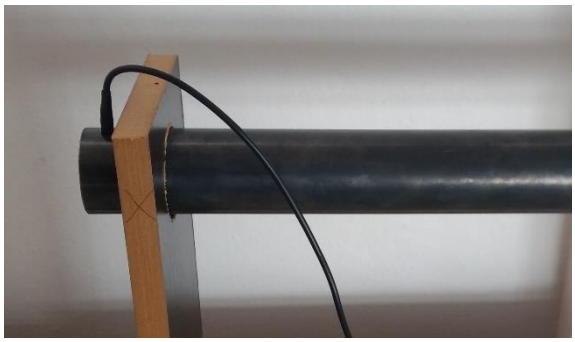
Com relação a estes parâmetros, fizemos alguns testes alterando a resolução do conversor de temperatura, uma vez que em máxima resolução, o sinal demora cerca de 750ms para ser identificado e processado. Contudo, a redução da resolução do sensor de temperatura, que pode ser configurada para 9, 10, 11 ou 12 bits, correspondendo a incrementos de 0,5 ° C, 0,25 ° C, 0,125 ° C e 0,0625 ° C, respectivamente, resulta na perca de qualidade dos valores de temperatura do aparato experimental. Além disso, o fabricante informa que o sensor opera na faixa de -55°C à 125°C (-67°F à 257°F), sendo que a acurácia de ±0,5°C está presente na faixa de -10°C à 85°C e fora dessa faixa o fabricante não garante tal precisão.

Como a leitura do sinal desse sensor demora cerca de 750ms, foram necessárias algumas providencias na programação do microcontrolador. Os códigos que contém essas providencias estão nos Apêndices A e B, sendo que o primeiro código foi usado para fazer as leituras e aplicar os comandos diretamente pela porta serial do computador na comunicação com o microcontrolador e o segundo código foi escrito para uma comunicação entre o microcontrolador e o *software* Matlab. A ação tomada, diante à demora da leitura do sinal do sensor, é fazer com o código interrompa a aplicação dos ângulos de disparo para o TRIAC a cada 30 mil ciclos do código. Isso garante que o microcontrolador poderá controlar o ângulo de disparo para o TRIAC e que também faça as leituras do sensor, pois caso isso não fosse feito, a cada ciclo do código do microcontrolador haveria uma pausa de 750ms atrapalhando a aplicação do sinal para o TRIAC

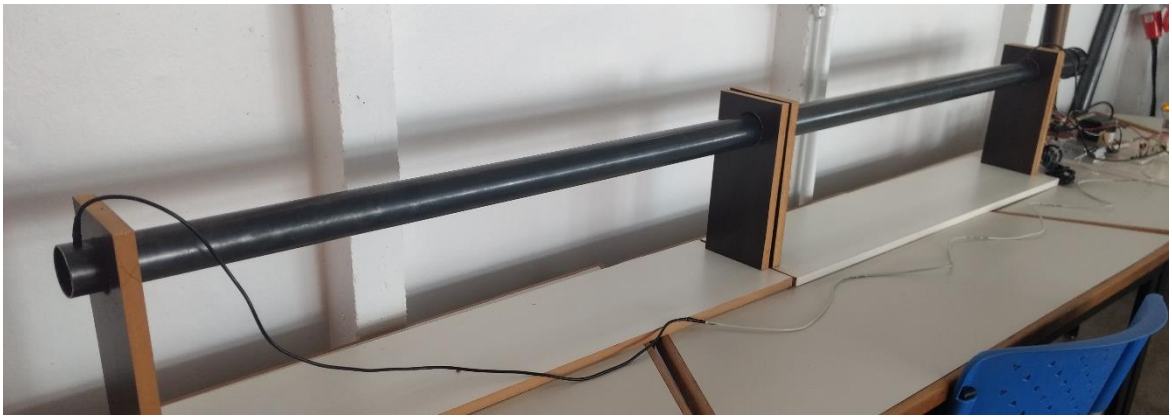
Como já apresentado, na Fig. 2.1, a planta de controle possui três sensores de temperatura, como está apresentado na Fig. 2.19 (c). O primeiro sensor foi colocado na entrada do tubo metálico, conforme a Fig. 2.19 (a), para mensurar a temperatura do ar que sai diretamente dos atuadores. O segundo sensor térmico foi colocado na saída de ar na tubulação, conforme a Fig. 2.19 (b), para mensurar a temperatura do ar após o processo de transporte térmico. O último sensor está afastado do tubo metálico em uma posição qualquer, desde que não tenha influência direta de algum sistema térmico, justamente para medir a temperatura ambiente. Todos os sensores foram conectados na mesma porta digital junto com um resistor *pull-up* de 4,7 KΩ, conforme a orientação do fabricante através do *datasheet* para utilização do protocolo 1-wire™.



(a)



(b)



(c)

Figura 2.19. (a) Sensor de entrada. (b) Sensor de saída. (c) Tubo metálico com os sensores instalados.

Ao construir todo o aparato experimental, é necessário que haja o condicionamento dos sinais gerados pelos sensores empregados antes de prosseguir com o projeto. É preciso também obter alguns dados do funcionamento da planta, pois precisamos saber qual a faixa de temperatura que a planta vai trabalhar. A planta de controle possui 3 resistências para o aquecimento do ar, sendo que quanto mais resistências são usadas, mais a planta irá aquecer. Logo, através destes testes iniciais, precisamos definir quantas resistências vamos usar para uma boa faixa de operação.

Depois de verificar o funcionamento de todos os componentes e ajustar o código do Arduino, realizamos um teste utilizando todas as resistências e aplicando vários degraus de referência, com amplitudes de 70% da tensão na carga e 30%, a fim de procurar uma faixa de operação ideal. Os degraus duraram períodos diferentes, pois também precisamos observar quanto tempo a planta demora para chegar ao regime permanente. O resultado está mostrado na Fig. 2.20.

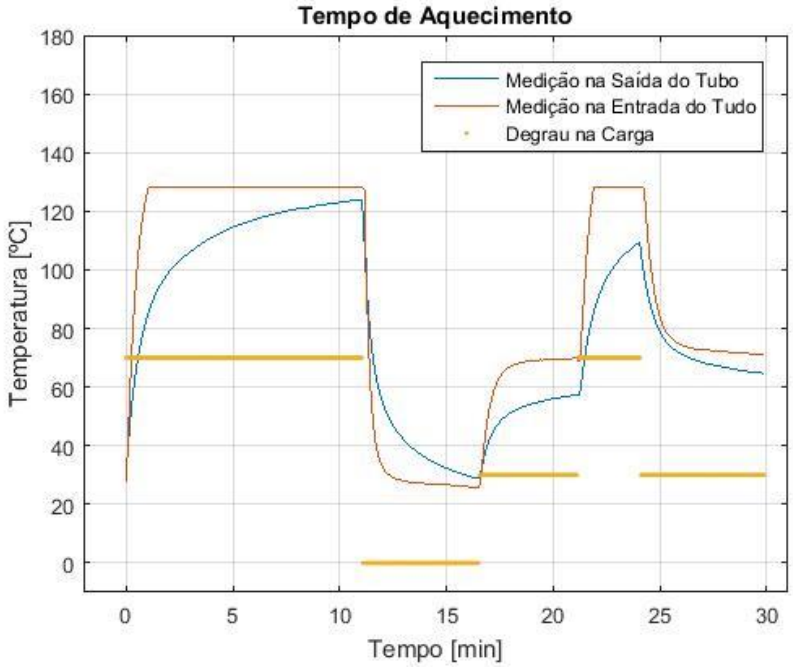


Figura 2.20. Primeiro teste de funcionamento da plana de controle.

Como podemos observar, a temperatura aplicada no sensor de entrada foi maior que a temperatura que o sensor pode mensurar e podemos dizer, assim, que a medição do sensor saturou quando a entrada de referência foi de 70% da tensão. Para uma entrada de 30% os sensores não chegaram a saturar, mas para este valor de tensão aplicado na carga, a temperatura produzida já era alta com relação a faixa de operação do sensor DS18B20.

Outra observação importante, com relação ao gráfico apresentado na Fig. 2.20 é que a temperatura no sensor de saída do tubo metálico não chegou ao regime permanente durante os 10 min iniciais do teste. Dessa forma, precisamos analisar a planta por um período de tempo maior.

Depois de analisar quantas resistências seriam necessárias para que a faixa de operação da planta de controle estivesse dentro de uma faixa em que o sensor pudesse mensurar da melhor forma possível, chegamos à conclusão de que a melhor configuração é usar uma resistência apenas e a entrada de referência entre 30% e 60%. Uma opção também proposta é utilizar duas resistências, todavia, neste caso os testes seguintes estarão susceptíveis a terem um aquecimento maior que o sensor possa mensurar.

Outro fato importante, com relação à faixa de operação está relacionado a Fig. 2.11 (b) , onde temos uma faixa linear entre o ângulo de disparo do TRIAC e a tensão RMS aplicada na resistência. É fundamental termos uma faixa de operação que não acione o ângulo de disparo em 100% nem em 0%. Assim, acionando até 60% da tensão da carga e começando a atuar na carga com uma tensão a partir de 30% da tensão nominal, estamos trabalhando em uma faixa de tensão onde a relação com o ângulo de disparo se aproxima da linearidade.

Com todos os parâmetros analisado e medidos, podemos apresentar a Tab. 2.1, onde consta as variáveis contidas no sistema. Importante destacar que apesar de termos o sensor medindo a temperatura de entrada do tubo metálico, essa informação não interessará ao objetivo do projeto de controle que temos.

Símbolo	Unidade	Descrição	Classificação
T_e	°C	Temperatura de entrada do tubo metálico.	Variável Medida.
T_s	°C	Temperatura de saída do tubo metálico.	Variável Controlada.
T_A	°C	Temperatura ambiente.	Variável de Distúrbio.
V_{RMS}	% Volts	Tensão na carga gerada pelo circuito do TRIAC.	Variável Manipulada.
V_{PWM}	%Volts	Tensão aplicada no motor pelo circuito amplificar do PWM (Ponte H).	Variável de Distúrbio.
E	°C	Temperatura de referência fornecida ao sistema em malha fechada.	Variável de Referência.
$G_{entrada}$	$^{°C}/_{V}$	Relação entre a porcentagem da tensão aplicada na entrada e a temperatura do sensor de entrada do tubo metálico.	Função de Transferência.
$G_{saída}$	$^{°C}/_{V}$	Relação entre a porcentagem da tensão aplicada na entrada e a temperatura do sensor de saída do tubo metálico.	Função de Transferência.
$Erro$	°C	Sinal de erro em malha fechada aplicado no PID	Variável Interna.
Sig	Volts	Sinal gerado pelo PID para controle da tensão aplicada na resistência	Variável Interna.

Tabela 2.1. Tabela das variáveis do processo.

Inicialmente pensamos em considerar a temperatura ambiente como um distúrbio do sistema, pois queremos obter um valor constante para a temperatura de saída do tubo metálico independente das variações da temperatura ambiente. Entretanto, a temperatura ambiente deve ser considera como uma segunda entrada do nosso sistema, pois mesmo antes de iniciarmos o processo, já existe uma temperatura senso mensurada na entrada decorrente da temperatura ambiente.

CAPÍTULO 3 – FUNDAMENTOS DO PROCESSO DE TRANSMISSÃO DE CALOR

No sistema, que construímos e estamos analisando, tem dois tipos de sistemas físicos que o caracterizam. O objetivo desse capítulo é abordar as naturezas físicas que envolvem a nossa planta de controle a fim de compreendermos a modelagem que faremos.

O primeiro tópico deste capítulo abordará o principal sistema física contido na planta de controle que é o sistema térmico. O tópico aborda as principais ideias de sistemas térmicos, em sua generalidade. O segundo tópico deste capítulo abordará um sistema que trabalha em conjunto com o processo de transmissão térmica, que é o sistema fluídico. Esta disciplina é relevante, pois temos um fluído, o ar, que é aquecido e soprado no interior de um tubo metálico. O assunto não constitui o principal fator de nossa modelagem, pois o foco está em mensurar a temperatura final, mas por se tratar de algo que influencia nos resultados, devemos compreender seus princípios.

3.1 SISTEMAS TÉRMICOS

Em sistemas físicos reais, o processo de transferência de calor de um ponto para outro será definido pela intensidade com a qual o calor é transferido de um ponto a certa temperatura para outro com temperatura inferior. A intensidade com a qual o calor é transferido dependerá das características físicas do meio, pois existe uma resistência térmica envolvida nesse processo. Contudo, a **resistência térmica** dependerá do modo em que ocorre a transferência de calor, podendo ser por condução, convecção ou radiação.

Contextualizando, em nossa planta de controle, temos a convecção de calor da resistência elétrica para o ar ao seu redor. O ar aquecido é soprado ao longo de um tubo feito de aço galvanizado, cuja espessura da chapa mede 2 mm, e ao longo desse caminho há a radiação de calor para o tubo metálico.

O processo de convecção que estamos analisando ocorre a partir do ar aquecido na resistência até a saída do ar do tubo metálico e o ar que se encontra dentro do tubo fornece uma resistência térmica. Em primeira análise, a resistência térmica existente no processo de radiação não nos interessa, uma vez que nossa análise de transmissão de calor é unidimensional, ou seja, estamos analisando a temperatura do ar que entra e sai do tubo metálico, apenas.

A resistência térmica, que anunciamos, pode ser comparada com a resistência elétrica através de uma analogia elétrica da transmissão de calor, conforme a Fig. 3.1.

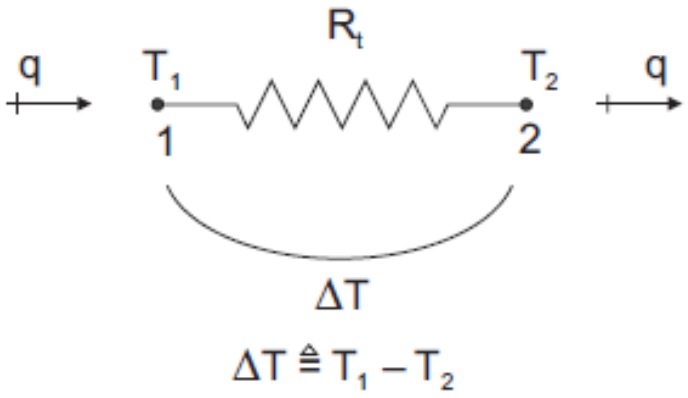


Figura 3.1. Representação esquemática de uma resistência térmica [5].

A expressão matemática, que descreve o processo físico do nosso modelo é dada por:

$$\Delta T = R_t . q \tag{12}$$

Onde:

$$\Delta T = T_1 - T_2 \tag{13}$$

A temperatura de entrada no tubo metálico é dada por T_1 e a temperatura de saída é dada por T_2. A resistência térmica do ar no interior do tubo metálico é dada por R_t e a taxa de transferência de calor entre a entrada e saída do tudo é dada por q.

Como anunciado, a resistência térmica está sujeita ao modo que ocorre o processo de transferência de calor. O principal modelo abordado em nossa planta de controle é a transmissão de calor por convecção e nesse caso, a resistência térmica é dada por: [5]

$$R_t = \frac{1}{h.A} \tag{14}$$

Onde h é o coeficiente de transferência de calor por convecção e A é a área da seção reta do tubo metálico, por onde passa o ar aquecido. Ou seja, a diferença de temperatura da entrada para a saída do tubo metálico é dada por:

$$T_1 - T_2 = \frac{q}{h.A} \tag{15}$$

Sabemos que em nosso processo térmico há a radiação de calor no processo de troca de energia entre o tubo metálico e o ar que circula em seu interior. Na primeira análise, o processo de radiação não interessa para a análise do nosso processo. De maneira geral, para determinados tipos de materiais e configuração físicas e geométricas, temos que a taxa de transferência de calor, no processo de radiação, é dada por: [5]

$$q = C_R(T_1^4 - T_2^4) \tag{16}$$

O valor de C_R é muito pequeno, o que resulta num valor elevado da resistência térmica, se comparado a resistência térmica que temos no nosso processo de convecção. Assim, enquanto estamos aquecendo o ar e soprando ele pelo tubo metálico, o processo de radiação

não terá grande efeito, todavia, ao logo do tempo a radiação de calor vai aquecendo o tubo metálico.

Durante esse tempo, um processo térmico, assemelhado ao capacitivo, acontece. Em determinado instante de tempo, a taxa de transferência de calor que entra no tubo metálico $q_i(t)$ e a taxa de transferência de calor que sai do tubo é $q_o(t)$ e podemos caracterizar o calor líquido armazenado no tubo em instantes de tempos distintos por:

$$\int_{t_0}^{t} [q_i(\lambda) - q_o(\lambda)] d\lambda \tag{17}$$

Este armazenamento líquido é assemelhado a capacitância elétrica e podemos igualar a Eq. 17 ao seguinte termo:

$$\int_{t_0}^{t} [q_i(\lambda) - q_o(\lambda)] d\lambda = C[T(t) - T(t_0)] \tag{18}$$

Onde C é a **capacitância térmica** $[^{J}/_{K}]$ do tubo metálico e $T(t) - T(t_0)$ é a variação da temperatura do tubo ao longo do tempo em que o ar aquecido flui em seu interior. Podemos reescrever a Eq. 18 da seguinte forma:

$$T(t) = T(t_0) + \frac{1}{C} \int_{t_0}^{t} [q_i(\lambda) - q_o(\lambda)] d\lambda \tag{19}$$

Embasado na Eq. 19 podemos analisar o processo térmico com mais abrangência. Após o processo de aquecimento, depois que o sistema atinge o regime permanente, o tubo metálico armazenou energia em forma de calor, contudo, após esse tempo, ao iniciar o processo de resfriamento o tubo metálico fornecerá energia para o ar que está fluindo no seu interior. Ou seja, mesmo que não estejamos mais fornecendo energia ao sistema, o sistema possui energia, em forma de calor, armazenada e toda a vez que houver um processo térmico, onde a temperatura no interior do tubo metálico for diferente da temperatura de seu interior, um processo capacitivo térmico ocorrerá.

Todavia, a definição que construímos para a capacitância térmica do tubo metálico é considerado o tubo como um corpo inteiro, ou seja, não fizemos uma análise pontual de transferência de calor. Como estamos falando de um corpo sólido, a temperatura seria uniforme em todos os instantes de tempo em todo o objeto caso ele tivesse uma condutividade térmica infinita, todavia isso não existe na natureza. Dessa forma, não haverá uniformidade de temperatura ao longo do tubo metálico durante o regime transitório do sistema.

3.2 SISTEMA FLUÍDICO

No livro Modelagem Dinâmica de Sistemas e Estudo da Resposta [5], o autor trata do assunto de sistemas fluídicos apresentando um modelo semelhante ao que encontramos em nossa

planta de controle. Estamos interessados nesse modelo de sistema para compreender como a vazão de ar pelo tubo reage nas trocas de calor.

Na Fig. 3.2 temos uma tubulação onde há uma vasão constante Q em regime permanente e as pressões estáticas P_1 e P_2 correspondentes as pressões de entrada e saída da tubulação, respectivamente.

Figura 3.2. Tubulação com escoamento em regime permanente [5].

Temos que a diferença de pressão que entra para a pressão que sai é em função da vazão Q. Se há diferença entre as pressões em função da vazão, existe algo que dificulta o fluxo de ar e podemos associar essa dificuldade à resistência, da mesma forma que fizemos com a resistência térmica. A expressão que define tudo isso é a seguinte:

$$P_1 - P_2 = R_f . Q \tag{20}$$

O termo R_f é definido como **resistência fluídica** entre a entrada e a saída da tubulação. Na Fig. 3.3 temos a representação da resistência fluídica de nosso sistema.

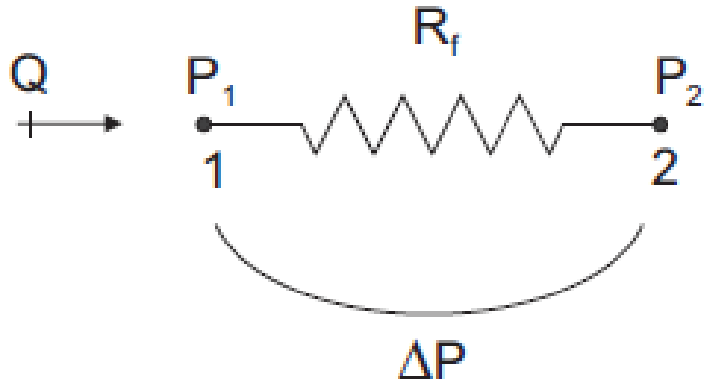


Figura 3.3. Representação de uma resistência fluídica [5].

Em outros experimentos, conforme mostrado pelo autor [5], a Eq. 20 tem uma boa relação com os dados experimentais de ΔP e Q quando o escoamento dentro do tubo é laminar. Quando o regime é turbulento, a mesma relação se torna não-linear, todavia, se aplicar a linearização da função, o conceito de resistência fluídica continua valendo.

O regime turbulento é caracterizado quando a velocidade do fluído é relativamente alta e nesse caso, o movimento das partículas de fluído são aleatórios em todas as direções acrescido do movimento de direção predominante de todo o fluído.

No caso do regime permanente podemos utilizar um parâmetro adimensional para prever o tipo de escoamento que está caracterizado, esse parâmetro é chamado de **Número de Reynolds (N_R).** Este parâmetro caracteriza as relações entre as forças de inércia e viscosa do fluido e a equação de descreve a relação é a seguinte:

$$N_R \triangleq \frac{\rho . D_T . \bar{V}}{\mu} \tag{21}$$

A Eq. 21 [5] leva em consideração escoamentos dentro de tubos com paredes lisas e seção circular, semelhante ao tubo metálico que utilizamos em nossa planta de controle. Temos que ρ é a massa específica do fluido, D_T é o diâmetro do tubo, $\bar{V}$ é a velocidade média do escoamento e μ a sua viscosidade.

Na nossa planta de controle, o único parâmetro relacionado ao número de Reynolds é a velocidade do ar no interior da tubulação. Não nos interessa criarmos um sistema onde o escoamento no interior do tubo metálico seja turbulento a fim de mantermos a linearidade do sistema.

3.3 PROPOSTA DE IDENTIFICAÇÃO

Além dos fundamentos físicos do sistema, precisamos analisar as técnicas de controle para a modelagem da nossa planta de controle. Nas literaturas de controle dinâmico, os sistemas térmicos são apresentados como um modelo com retardo de transporte, ou tempo morto, conforme apresentado na Equação (22).

$$G(s) = K \frac{1}{s+1} e^{-Ts} \tag{22}$$

Contudo, ao analisarmos o condicionamento dos sinais dos sensores, apresentado na Fig. 2.19, não fica claro que o nosso processo físico possui um tempo morto significativo. Ou seja, ao aplicarmos uma entrada de referência no sistema, os sinais apresentados pelos sensores de temperatura possuem uma resposta tão rápida, que o tempo morto está na grandeza de alguns segundos. Diante disso, podemos seguir a orientação apresentada por Ogata [1], onde ele afirma que se a constante de tempo referente ao atraso de transporte for pequena, podemos aplicar a simplificação da Equação (23)

$$e^{-Ts} = \frac{2 - Ts}{2 + Ts} \tag{23}$$

Assim, com essa simplificação, podemos propor que o modelo original seja alterado para um modelo onde haja 1 zero e 2 polos, conforme a Eq. 24:

$$G(s) = K \frac{1}{s+1} \left(\frac{2 - Ts}{2 + Ts} \right) \tag{24}$$

Ao comentar sobre os efeitos dos sensores no desempenho de sistema, Ogata [1] afirma que “a resposta de um sensor térmico é normalmente do tipo de segunda ordem superamortecido”. Ou seja, um modelo que identifica um sensor térmico apresenta 2 polos.

Este tipo de modelo apresenta um grau maior que o modelo clássico para processos térmicos. Dessa forma, como as informações do nosso processo estão pautadas em sensores térmicos, podemos reforçar a proposta de um modelo que possua 2 polos.

Quanto ao número de zeros, podemos propor 1 zero para o nosso modelo. Essa proposta é feita a partir de modelos experimentais feitos pelo professor Adolfo Bauchspiess em seus experimentos de Identificação de Sistemas Dinâmicos.

CAPÍTULO 4 – IDENTIFICAÇÃO E MODELAGEM

Como é possível observar, o processo que estamos trabalhando é complexo, assim como muitos outros processos físicos no mundo real. Dessa forma, precisamos criar um modelo que caracterize da melhor forma possível nosso processo térmico, para que possamos compreendê-lo e controla-lo da melhor forma possível. Luiz Carlos Felício comenta em seu livro [5] que é praticamente impossível descrever todos os aspectos físicos reais que acontecem em um sistema, assim, temos que decidir quais as características devemos considerar e ignorar.

Como comentamos no capítulo 3, existem aspectos físicos em nosso processo físico que são mais relevantes e que é o alvo principal de nossa análise. Assim, por se tratar de um trabalho que estuda um sistema térmico, o nosso modelo privilegia o sistema térmico.

Para identificarmos o modelo do sistema dinâmico que estamos estudando, precisamos obter dados experimentais que relacionem as variáveis de entrada, perturbações e a saída do processo. Dessa forma, criamos um ensaio que possa apresentar o comportamento da planta de controle e para isso aplicamos um sinal pseudoaleatório, também conhecido como PRBS (*Pseudo Random Binary Signal*), pois ele gera vários pulsos com a mesma amplitude, mas com diversos períodos de duração, conforme mostrado na Fig. 4.1.

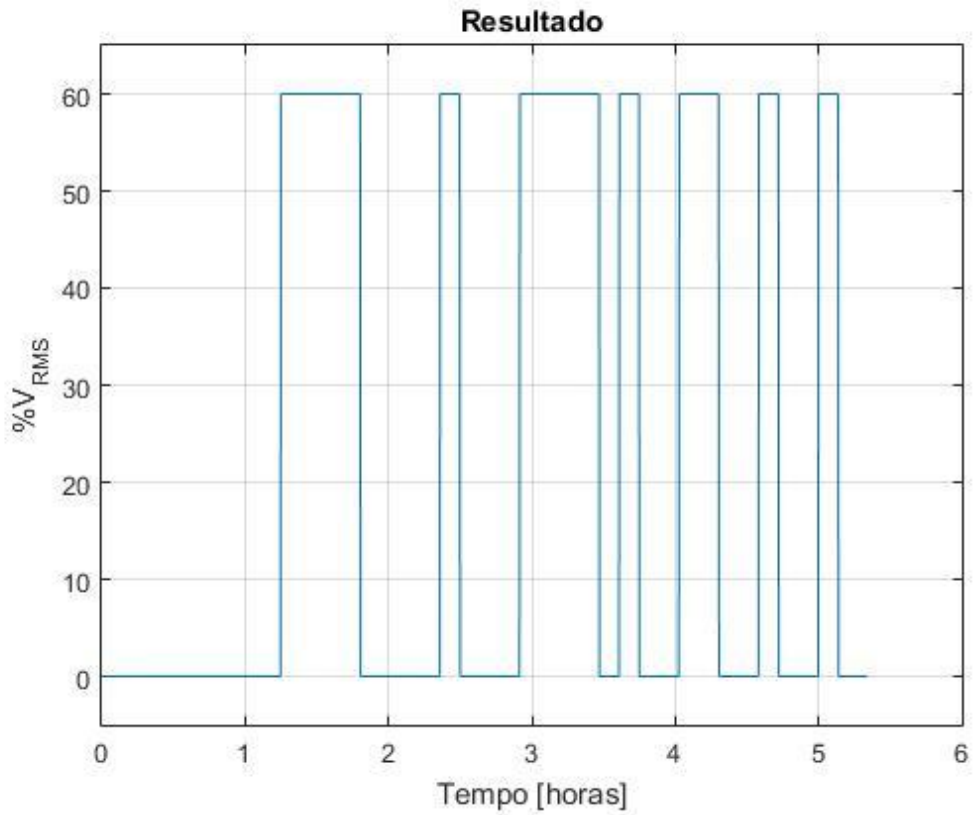


Figura 4.1. Sinal pseudoaleatório.

Na identificação de sistemas dinâmicos, a utilização de PRBS é mais comum. Os pulsos aplicados podem assumir apenas dois valores distintos, que no nosso caso é em 60% e 0%, permanecendo instantes de tempos aleatórios em cada um desses níveis. Contudo, o tempo mínimo ou máximo de permanência em cada nível é determinado matematicamente. O cálculo para se determinar os períodos de permanência em cada nível serve para o sinal PRBS excitar uma grande faixa de frequências com uma boa intensidade.

Além disso, esse tipo de sinal é importante para a identificação do sistema, pois ele apresenta diversos períodos de duração do um pulso e como não sabemos ainda qual é o comportamento da planta, como o tempo para regime permanente e seus transitórios. A forma como esse sinal foi gerado é apresentado no Apêndice C, através do *software* MATLAB.

Para aplicar o sinal na planta de controle, o MATLAB precisa enviar o sinal pseudoaleatório ao microcontrolador através da porta serial do computador. Esta operação é simples, mas deve-se tomar o cuidado com a formatação dos números que serão escritos na porta serial, pois o microcontrolador foi programado para ler apenas números inteiros como entrada.

Nesta comunicação foi preciso uma atualização do código do microcontrolador que usamos no Capítulo 2 – Descrição do Aparato Experimental, pois é preciso que o microcontrolador, além de receber o sinal do MATLAB, ele fornece apenas os valores da temperatura. Como o código inicial, tínhamos acesso aos valores de temperatura e enviávamos valores de entradas, precisamos atualiza-lo para que o microcontrolador imprima na porta serial do computador apenas os valores necessários, sem textos descrevendo quais são esses valores.

O Arduino lê as temperaturas dos 3 sensores de temperatura e os armazena em um vetor, que é enviado ao MATLAB e este, por sua vez, vê o vetor e identifica a que sensor pertence cada valor contido no vetor. O próprio código escrito no MATLAB apresenta o tempo de amostragem do sistema para a leitura destes sensores, que é cerca de 10 segundos. Este tempo pode parecer um pouco longo na primeira análise, todavia, como os processos térmicos tendem a perdurar um longo período, o número de amostrar obtidas fornecerá um resultado satisfatório.

Além disso, é difícil diminuir esse tempo de amostragem devido as características construtivas que temos em nossa planta de controle, uma vez que, como apresentado no capítulo anterior, a leitura dos dados enviados pelo sensor, demora um tempo consideravelmente longo. Os ensaios em conjunto com a identificação e modelagem do sistema serão apresentados nas seções a seguir.

4.1 ENSAIO COM SINAL PSEUDOALEATÓRIO – PRBS

Após algumas análises de como melhor configurar a programação do sinal pseudoaleatório, o sinal, mostrado na Fig. 4.1, foi utilizado no processo. Foi tomado o cuidado para que os

sinais gerados pudessem apresentar períodos tais que o processo atingisse o regime permanente, durante o aquecimento e o resfriamento da planta. Esse cuidado é importante, pois o processo térmico em questão se apresenta como um processo lento.

O resultado do teste é apresentado no gráfico da Fig. 4.2. A duração desse teste foi de aproximadamente 6 horas e foram coletados os dados dos 3 sensores e estão distribuídos no plano conforme a informação da legenda.

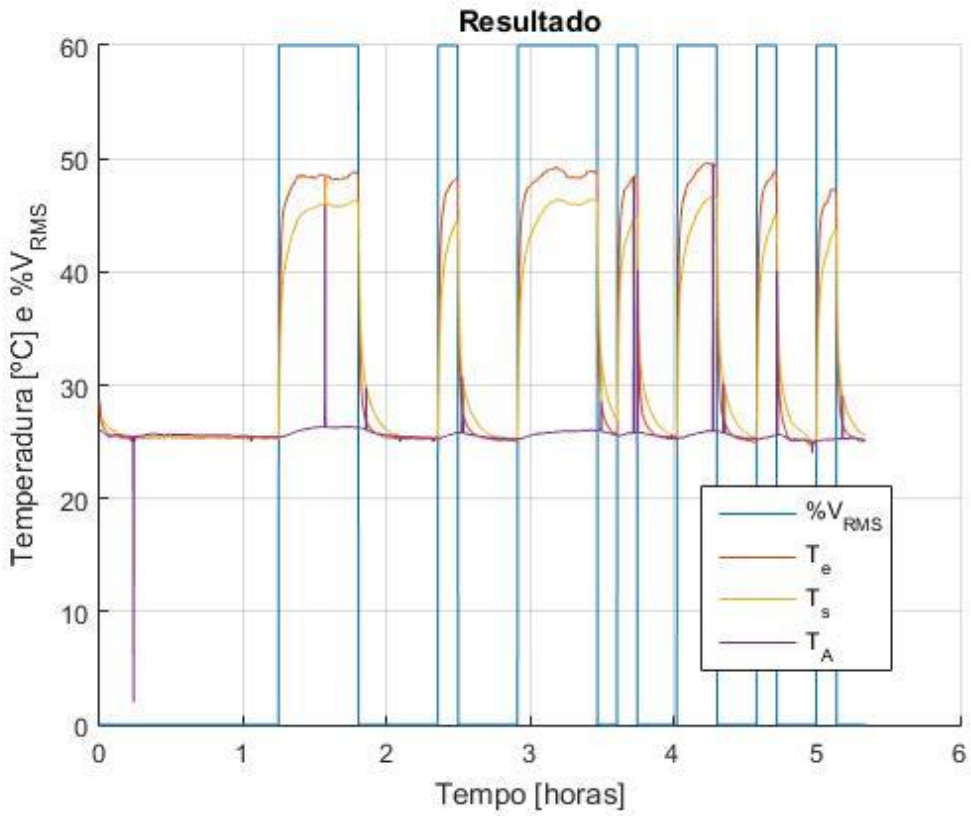


Figura 4.2. Resultado do ensaio com o sinal Pseudoaleatório.

Apesar dos sensores serem digitais, eles estão sujeitos a interferências ambiente e essas interferências podem provocar erros de leitura. Quando isso acontece, o microcontrolador marca o valor de -127ºC e podemos notar isso no gráfico nos picos de temperatura aleatórios que não correspondem a uma expectativa física real do sistema.

Para melhorar a apresentação dos resultados da Fig. 4.2, podemos remover os Outliers da temperatura ambiente. Esse processo também é feito no MATLAB, mas em um novo conjunto de códigos, que estão apresentados no Apêndice D com o uso da ferramenta estatística chamada amplitude interquartil (ou Intervalo Interquartil – IIQ), ou do inglês, *Interquartile Range (IQR).*

O intervalo interquartil oferece um critério matemático e probabilístico para identificar os valores discrepantes das nossas temperaturas, os chamados *Outliers*. Segundo o professor Marcelo de Souza Lauretto [6], o intervalo interquartil estabelece limites superior e inferior

para os valores admissíveis em uma dada amostra de informações, conforme a seguinte equação:

$$Limite\ Inferior = Q_1 - c.IQR \quad (25)$$

$$Limite\ Superior = Q_3 + c.IQR \quad (26)$$

No código do MATLAB, do Apêndice D, definimos a constante $c = 1{,}50$, como é de costume usar para eliminar os *outliers*.

O resultado desse processo é mostrado na Fig. 4.3 e podemos observar que os valores dos sensores, plotados no gráfico, estão melhor apresentados. Além disso, a referência na entrada do nosso processo, também está condizente com os parâmetros mensurados no sistema.

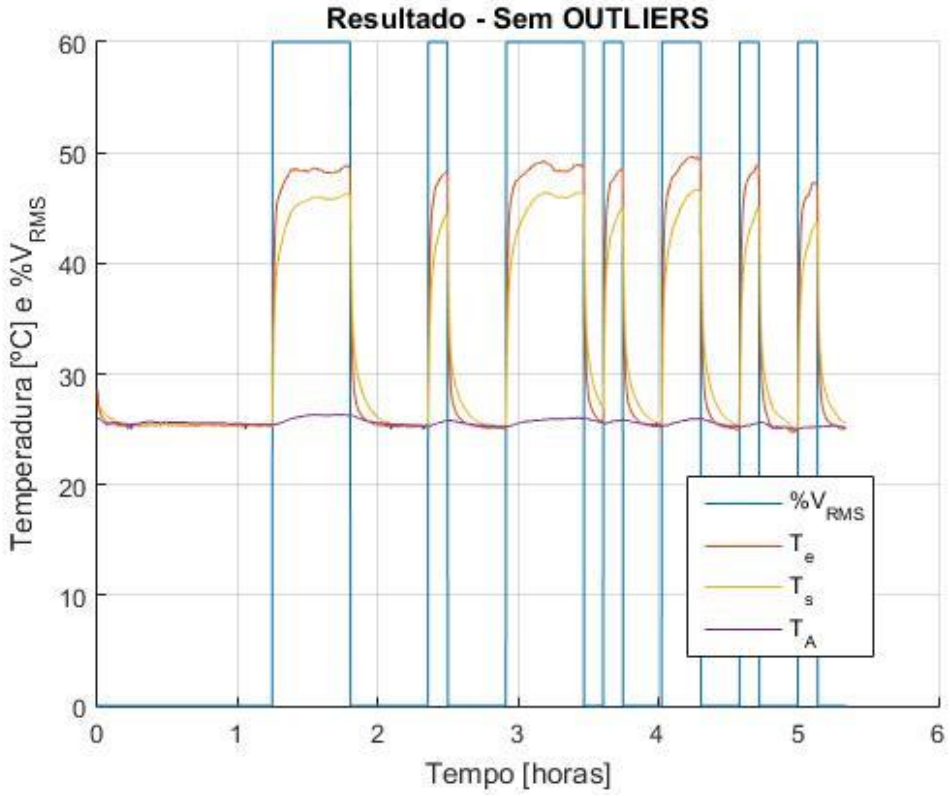


Figura 4.3. Ensaio com o sinal Pseudoaleatório com a remoção dos Outliers.

A partir dos resultados apresentados, podemos reiterar os dados esperados quanto a amostragem do sistema. Ao final de seção anterior, informamos que a amostragem feita no MATLAB, dos sensores do aparato experimental, era de cerca de 10 segundos e como podemos observar na Fig. 4.3, a quantidade de amostras obtidas ao longo do processo foi satisfatória para obtermos as curvas de resposta. A partir disso, devemos identificar o modelo matemático. A informação do período de amostragem é fundamental nesse processo.

4.2 MODELO MATEMÁTICO

A partir dos dados levantados no ensaio, que foram apresentados na seção anterior, avançamos na análise do problema procurando identificar um modelo matemático que se aproxime do comportamento da planta de controle. No nosso caso, temos as entradas e saídas do sistema e a partir disso queremos encontrar a função de transferência.

No próximo capítulo, vamos procurar validar a função de transferência que queremos identificar nesse capítulo. Isso é importante porque não basta apresentarmos uma equação que caracterize a planta de forma a fornecer apenas as saídas de uma determinada entrada e para outras entradas não corresponda ao comportamento da planta. Ou seja, queremos identificar uma função de transferência que para diversas entradas, forneça saídas que correspondam à realidade que a planta forneceria.

Assim, contextualizando para o nosso problema em específico, a engenharia aqui aplicada busca identificar o modelo do sistema de forma aproximada que represente com maior fidelidade o problema que estamos envolvidos.

A ferramenta que utilizamos para identificar o sistema foi a toolbox do MATLAB chamada *System Identification*, mostrada na Fig. 4.4, que em uma possível tradução é Identificação de Sistemas e foi projetada para resolver problemas como o nosso. É possível ter acesso a essa ferramenta no MATLAB através de dois meios, ou pelo menu Apps presente na parte superior do programa ou escrevendo na janela de comando o comando *ident*.

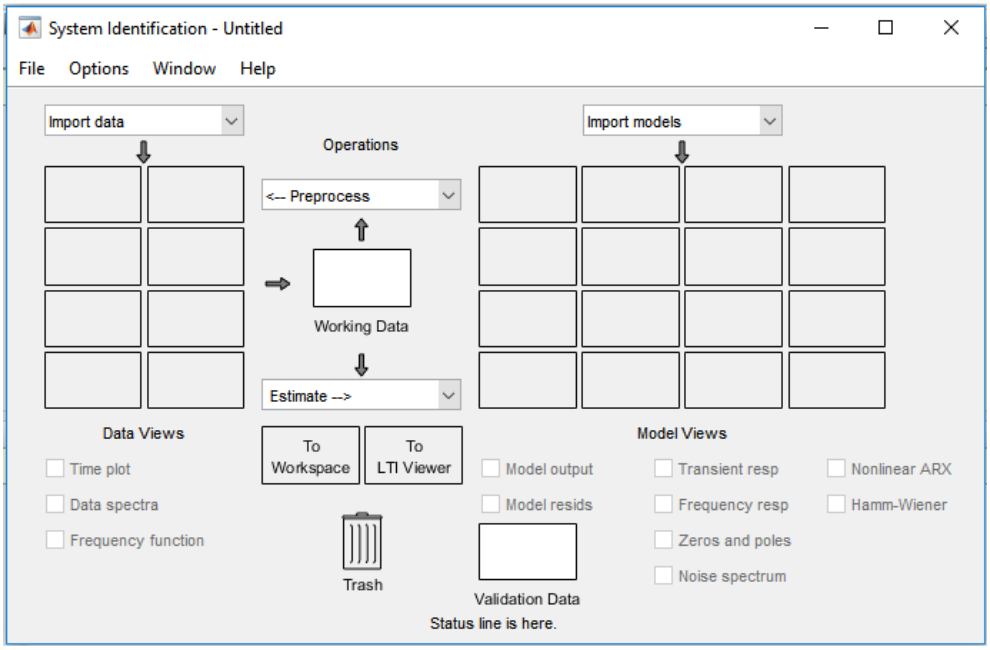


Figura 4.4. *System Identification.*

O MATLAB fornecerá a janela da Fig. 4.4 onde é possível importar os dados de entrada e saída do sistema, em vários tipos de domínios. É possível escolher qual tipo de processo

queremos utilizar na identificação e podemos enviar os resultados dessa operação, no caso, a função de transferência, para o MATLAB.

Fizemos a identificação para a função de transferência de entrada do tubo metálico e para a saída. Ou seja, fizemos uma identificação para os valores de saída do sensor que fica na extremidade de entrada do tubo metálico e uma identificação para os dados de saída do sensor que fica na extremidade de saída do tubo metálico.

Para o sensor que fica na entrada do tubo metálico obtivemos a seguinte função de transferência:

$$G_{entrada}(s) = \frac{T_e(s)}{V_{RMS}(s)} = \frac{0{,}3764s + 0{,}0007775}{s^2 + 1{,}182s + 0{,}0005309} \quad (27)$$

A função de transferência que melhor atendeu a nossa realidade, que envolve um processo térmico, possui dois polos e um zero, como mostrado na Eq. 27. O resultado esperado dessa função de transferência é mostrado na Fig. 4.5, através da curva $\hat{T}_e$. Contudo, não é apresentado, através desse modelo, a representação no atraso de transporte, ou o tempo morto, do processo, pois, como podemos observar na Fig. 4.3, a resposta do sistema à uma entrada degrau é praticamente instantânea. Assim, não precisamos colocar na função de transferência um elemento exponencial que tende a um valor muito próximo de 1 (um), pois essa constante não afetará a resposta transitória do sistema, muito menos em regime permanente.

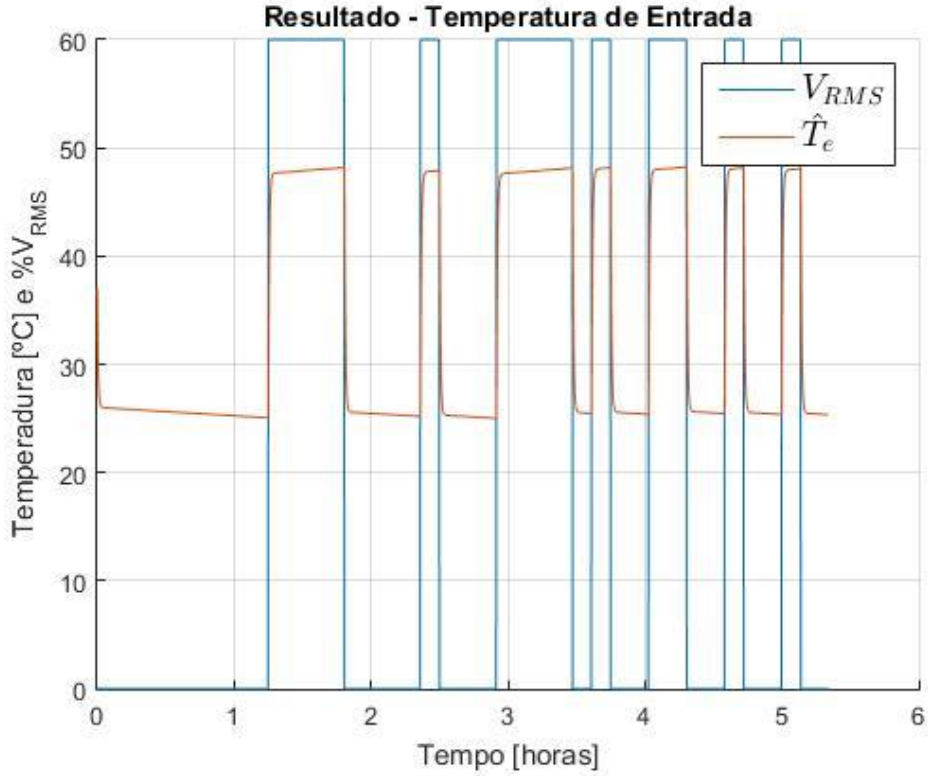


Figura 4.5. Resultado da temperatura de entrada esperada em função da variação da tensão de carga aplicada na resistência.

Segundo a estimativa do MATLAB, o ajuste da função de transferência da Eq. 27 ficou em 85,61%. Isso reitera a ideia passada por Luiz Carlos Felício, que a identificação aqui feita não é exata, mas um valor aproximado da realidade que queremos expressar. Em seu livro, Ogata [1] também informa que um sistema pode ser representado de diferentes maneiras, ou seja, podemos ter mais de uma função de transferência que pode representar um sistema dependendo inclusive do aspecto físico que se deseja caracterizar.

Para melhor identificar os polos e o zero da função de transferência do sensor de entrada, podemos apresentar a função conforme a Eq. 28:

$$G_{entrada}(s) = 0{,}3764\frac{s + 0{,}0020656}{(s + 1{,}18155)(s + 0{,}000449)} \tag{28}$$

Para o sensor que fica na saída do tubo metálico obtivemos a seguinte função de transferência:

$$G_{saida}(s) = \frac{T_s(s)}{V_{RMS}(s)} = \frac{0{,}9909s + 0{,}001789}{s^2 + 2{,}633s + 0{,}001325} \tag{29}$$

Da mesma forma, que a função de transferência para o sensor que fica na entrada do tubo metálico, nesse caso utilizamos uma função com dois polos e um zero. O resultado esperado dessa função de transferência, para a entrada pseudoaleatória, que usamos na identificação, é mostrada na Fig. 4.6, através da curva $\hat{T}_s$. Segundo a estimativa do MATLAB, a função de transferência da Eq. 29 possui um ajuste da função de transferência de 91,24%.

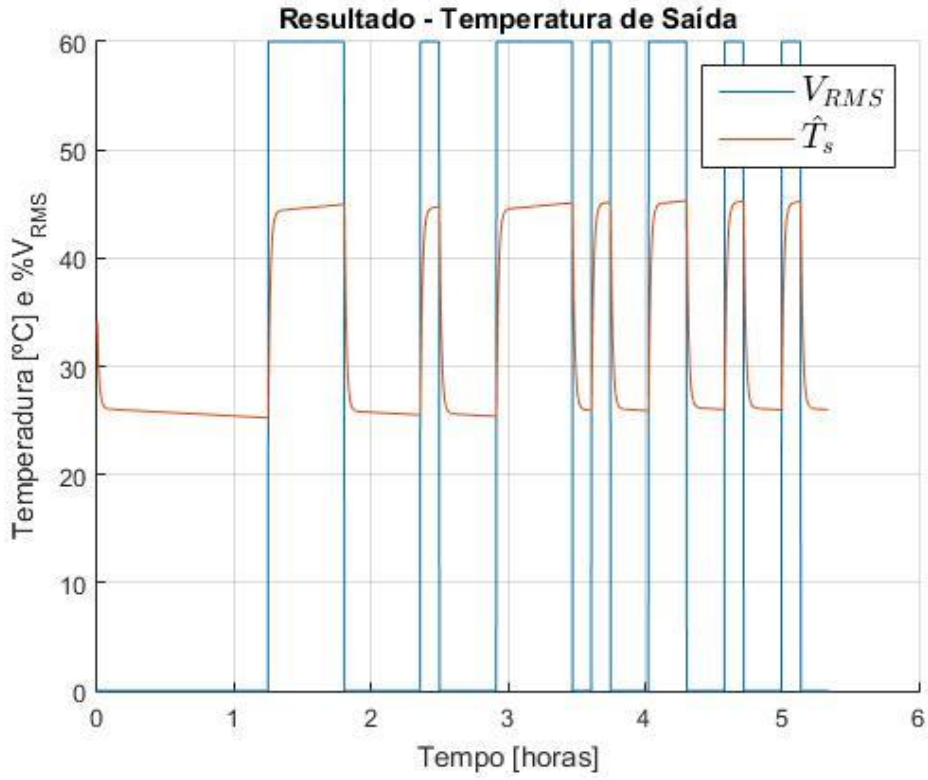


Figura 4.6. Resultado da temperatura de saída esperada em função da variação da tensão de carga aplicada na resistência.

Para melhor observar os polos e o zero da função de transferência que nos interessa para controle do processo, podemos apresentar a Eq. 30:

$$G_{saida}(s) = 0{,}9909\frac{s + 0{,}001805}{(s + 2{,}6325)(s + 0{,}0005033)} \quad (30)$$

Importante ressaltar que nas Fig. 4.5 e Fig. 4.6, as curvas $\widehat{T_e}$ e $\widehat{T_s}$ se referem às simulações das funções de transferência $G_{entrada}(s)$ e $G_{saida}(s)$, respectivamente, no software MATLAB. Ou seja, as curvas $\widehat{T_e}$ e $\widehat{T_s}$ são projeções feitas do que devemos esperar da temperatura de entrada (T_e) e da temperatura de saída (T_s), respectivamente. Por fim, para o nosso modelo, precisamos analisar as perturbações e demonstrar como elas agem ao longo do processo.

Ao identificarmos as funções de transferência, consideramos duas entradas no sistema, uma vez que, ao iniciarmos o processo, a temperatura inicial não é nula. Ou seja, consideramos que o sistema possui uma outra entrada que é a temperatura ambiente. O diagrama de blocos, apresentando como as entradas são dispostas no modelo, está na Fig. 4.7.

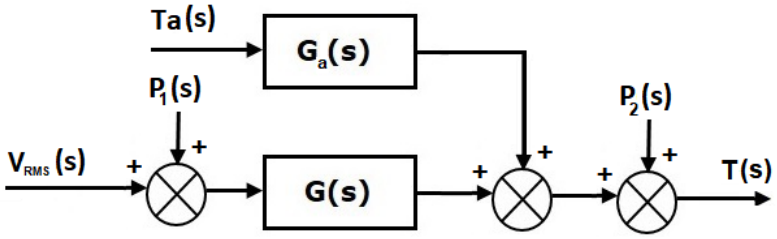


Figura 4.7. Diagrama de blocos do modelo em malha aberta.

O diagrama de blocos, da Fig. 4.7, é utilizado tanto no modelo feito para o sensor de entrada, quanto para o sensor de saída. Ou seja, a Fig. 4.7 pode ser usada com a função de transferência da entrada ou com a função de transferência de saída do tubo metálico, onde foi colocado G(s). Em ambos os casos, a entrada de referência é a mesma nos dois casos.

A função $G_a(s)$ é usada para a entrada da temperatura ambiente. Nesse caso, se considerarmos a entrada de referência na resistência elétrica nula, a única entrada do sistema será a troca de calor que a planta faz com o ambiente. Ou seja, se a temperatura do ambiente aumentar, a temperatura nos sensores aumentará proporcionalmente. Em qualquer situação, com a planta de controle funcionando, a velocidade do ar soprado no tubo metálico é constante, assim as variações que acontecem no ambiente, também acontecem no tubo metálico e para obtermos um modelo mais simplificado, consideramos que a função de transferência $G_a(s)$ é igual à um.

Quanto as perturbações colocadas no modelo, a perturbação $P_1(s)$ considera os ruídos que podem acontecem nas variações de temperatura da entrada de referência. Isso pode ocorrer, uma vez que é possível acontecer variações na tensão aplicada na resistência elétrica. A perturbação $P_2(s)$ considera os ruídos de medida dos sensores.

Esse conjunto de fatores vão definir como devemos abordar o projeto de controle e malha fechada e o que devemos esperar do sistema operando com a realimentação.

4.3 VALIDAÇÃO DO MODELO

Obtido um modelo matemático que representa a resposta da planta de controle no domínio *s*, precisamos avaliar o modelo a fim de averiguar se há validade nas informações obtidas. Caso exista uma discrepância dos dados que a função de transferência gerar para com os dados reais da planta de controle, devemos voltar à identificação do nosso processo.

Para a entrada pseudoaleatória, que aplicamos no sistema, a própria *toolbox* do MATLAB, que usamos para identificar, já faz uma análise informando a porcentagem de quanto o modelo identificado encontrado se aproxima dos dados reais. Como mostrado no subcapítulo anterior, a função de transferência para o sensor de entrada corresponde a 85,61% e podemos observar essa comparação na Fig. 4.8.

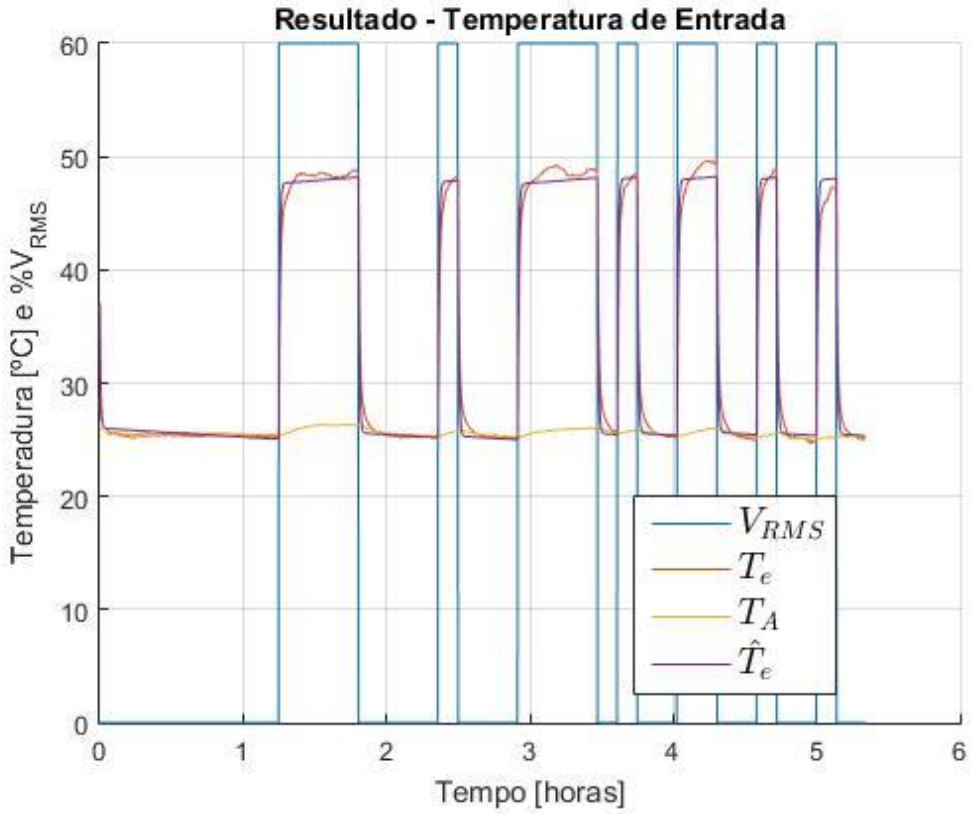


Figura 4.8. Comparativo entre as curvas da tensão aplicada na resistência, das temperaturas de entrada e ambiente e da temperatura de entrada esperada do sistema.

Para a resposta da função de transferência do sensor que fica na saída do tubo metálico, a *toolbox* do MATLAB informou que a função e transferência se aproximou em 91,24% da resposta da planta. Podemos observar a comparação entre a função de transferência e a resposta real da planta para este sensor na Fig. 4.9.

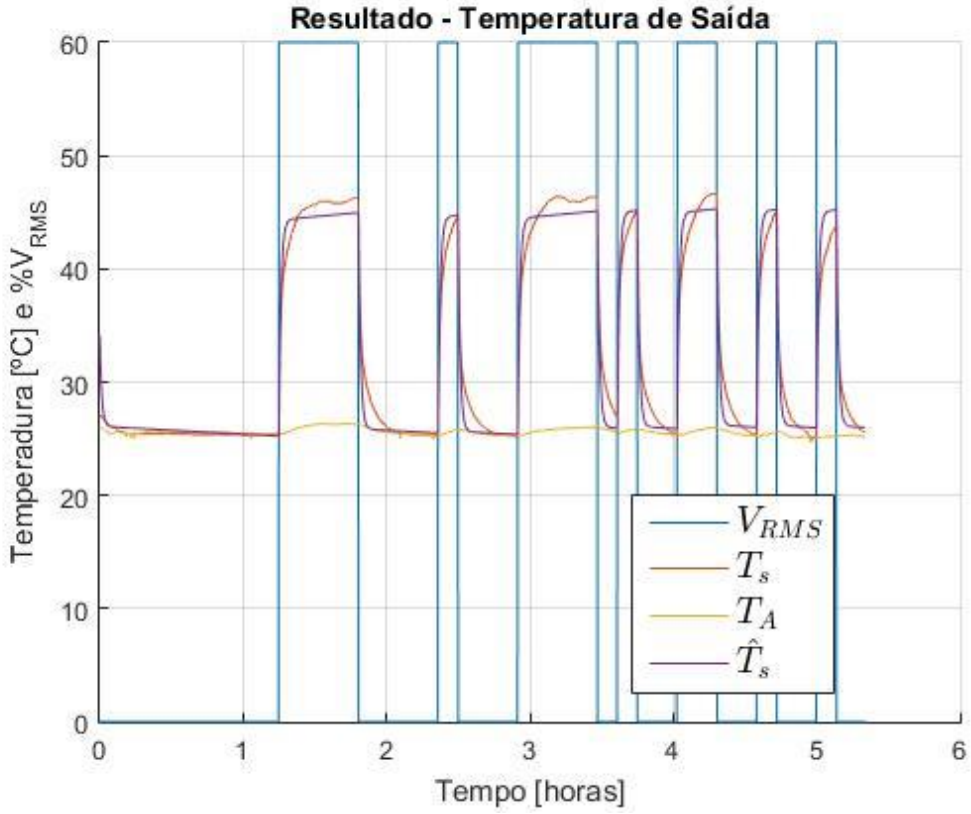


Figura 4.9. Comparativo entre as curvas da tensão aplicada na resistência, das temperaturas de saída e ambiente e da temperatura de saída esperada do sistema.

Como podemos observar, as funções de transferência obtidas no sistema se aproximam bastante da realidade. As discrepâncias vão aparecer naturalmente, pois o sistema real está sujeito a perturbação do ambiente que está inserido. Temos impresso nos gráficos a temperatura ambiente de um ponto onde a planta está e podemos observar que essa temperatura não é constante, ou seja, isso já é motivo para caracterizar uma perturbação natural do sistema. A planta de controle é alimentada diretamente da rede elétrica, ou seja, perturbações no sistema elétrico podem causar oscilações na planta.

Para aprofundar a análise da validação, um teste semelhante ao de identificação foi realizado, porém, com a amplitude de entrada diferente, a fim de observar o comportamento da planta e se a função de transferência que modelamos fornecerá valores próximos à realidade. Assim, a amplitude de entrada foi reduzida pela metade, ficando em 30%. O resultado do teste está apresentado na Fig. 4.10.

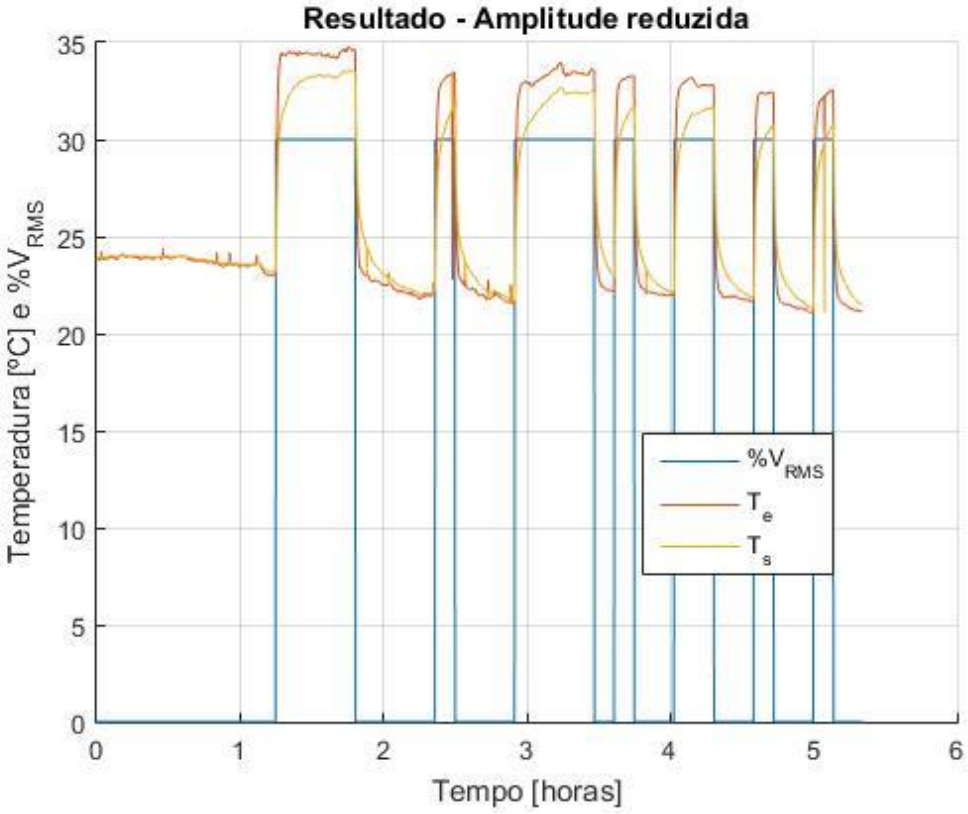


Figura 4.10. Teste com a amplitude reduzida.

No gráfico, podemos observar que a temperatura final da planta, tanto para o sensor de entrada, quanto para o sensor de saída reduziu. Outra observação importante é que neste teste, a temperatura ambiente se comportou diferente do teste anterior e dessa vez, ela reduziu cerca 4ºC ao longo do período do teste. O comparativo gráfico para o sensor de entrada entre os dados da planta e o resultado da função de transferência é mostrado na Fig. 4.11. Para o sensor de saída, o comparativo está apresentado na Fig. 4.12.

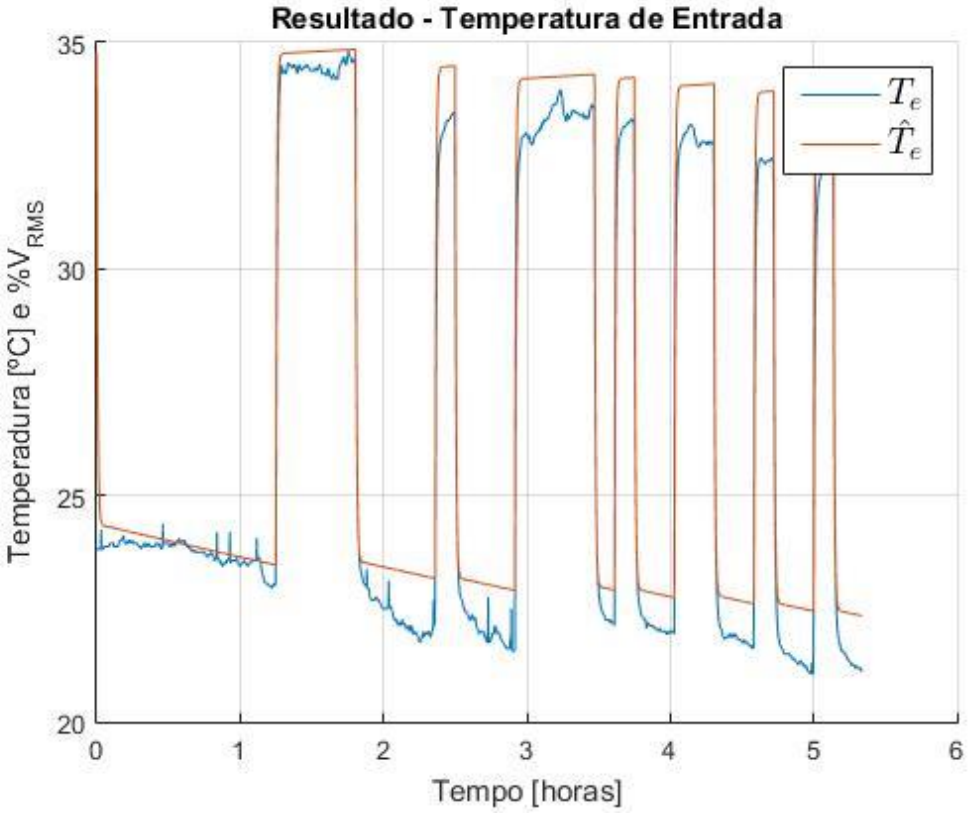


Figura 4.11. Comparativo com amplitude reduzida entre o valor da temperatura enviada pelo sensor de entrada e a temperatura de entrada esperada através da função de transferência.

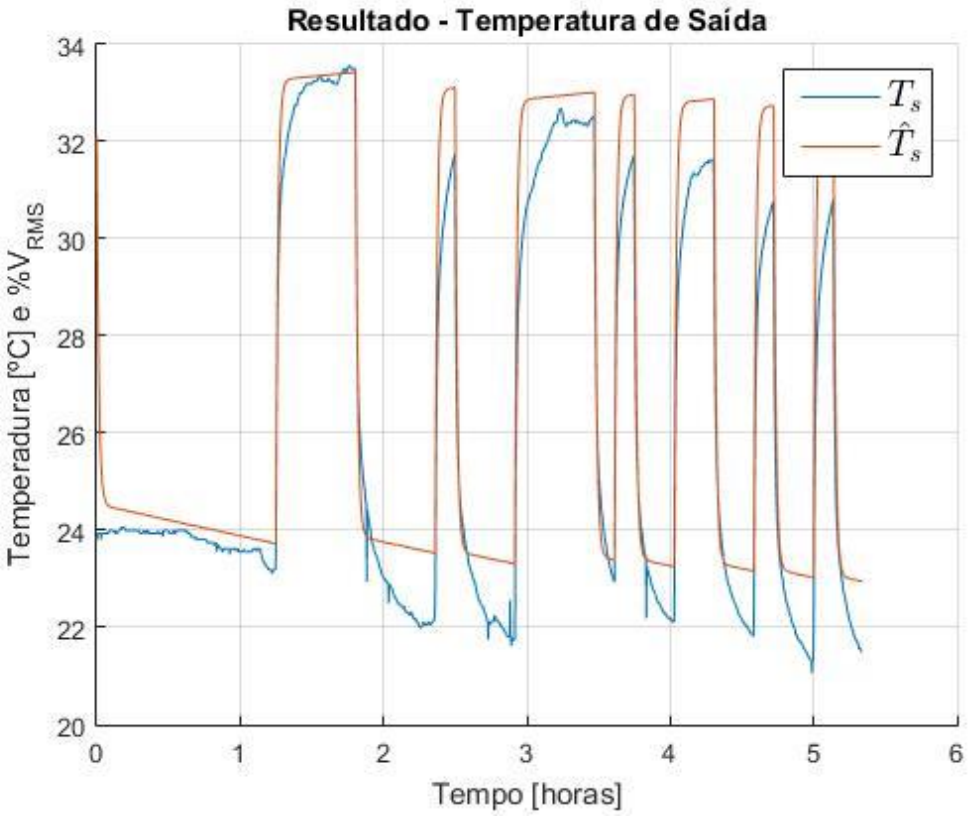


Figura 4.12. Comparativo com amplitude reduzida entre o valor da temperatura enviada pelo sensor de saída e a temperatura de saída esperada através da função de transferência.

Como mostramos, a temperatura de entrada variou ao longo do tempo. Isso é relevante pois a modelagem foi feita considerando a temperatura ambiente uma constante, logo, podemos observar que, principalmente para o sensor na saída do tubo metálico, a função de transferência apresentou uma diferença para o valor apresentado na planta.

Contudo, a função de transferência chegou a se aproximar dos valores dados pela planta, considerando que houve a variação da temperatura ambiente. Neste caso, podemos considerar a temperatura ambiente uma perturbação que não podemos identificar uma função de transferência para ela. Dessa forma podemos tentar amenizar esse distúrbio no projeto de controle, mas a planta estará sujeita a esse tipo de distúrbio.

CAPÍTULO 5 – PROJETO DE CONTROLE DINÂMICO

A identificação e a modelagem do sistema são requisitos necessários para que possamos desenvolver uma técnica de controle dinâmica. A ideia de desenvolver uma técnica de controle dinâmica para a plana é que ela possa ajustar a entrada do processo a fim de minimizar as perturbações e forneça na saída a temperatura mais próxima possível da esperada.

Como desejamos eliminar ou minimizar ao máximo as perturbações e a diferença da temperatura fornecida com relação à temperatura esperada, o primeiro passo que devemos dar na direção do projeto de controle é pensar em um sistema de malha fechada. Conforme Katsuhiko Ogata [1], as perturbações previstas no sistema podem ser compensadas pelo controle em malha fechada e o sistema fica suscetível somente às perturbações. Espera-se que para um bom projeto de controle os distúrbios não-previstos não tenham grande impacto no sistema. Para Dorf e Bishop [7], o controle dinâmico com realimentação é precioso, pois dá ao engenheiro a capacidade de ajustar a resposta transitória, reduzindo significativamente o efeito de perturbações.

Assim, a projeto de controle dinâmico em malha fechada deve ficar conforme o diagrama de blocos da Fig. 5.1. A ideia desse projeto é poder analisar as duas entradas do sistema, a entrada de referência e a temperatura ambiente, e a partir delas encontrar a diferença da temperatura de saída da planta de controle. Com essa diferença de temperatura, de entrada e saída, espera-se que o sistema possa se ajustar a fim de minimizar essa diferença.

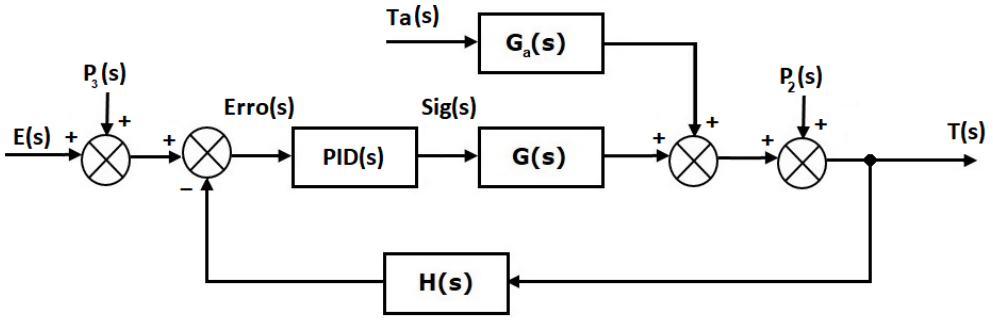


Figura 5.1. Diagrama de blocos para o projeto em malha fechada.

A diferença entre a referência e a temperatura de saída da planta gera um sinal que chamamos de erro, mostrado no diagrama de blocos. Este Erro(s) será aplicado no sistema de controle e o objetivo do projeto é minimizá-lo, pois assim, se houver um distúrbio no sistema ou se a temperatura ambiente variar, o sistema de controle gerará um novo sinal Sig(s) que deverá ajustar a planta de controle. Esse processo de controle dinâmico vai ao encontro do que o falamos no começo do capítulo a respeito de minimização de perturbações. A implementação desse sistema foi feita através do MATLAB conforme o gráfico mostrado no

Apêndice E. Adiante, mostraremos como foi projetado e implementado o bloco do PID(s), assim como o bloco de realimentação H(s).

Na indústria, existem basicamente três tipos de controladores de temperatura: liga/desliga, Proporcional e o Proporcional Integral Derivativo, também conhecido como PID [8]. O controlador lida e desliga aciona a saída do sistema apenas quando a temperatura atinge determinado ponto pré-ajustado e é utilizado quando não se deseja precisão na variável controlada do processo, além do que ele não minimiza as perturbações degradando a qualidade o sistema.

O desenvolvimento do controle proporcional elimina os ciclos do controlador liga e desliga, porém, ele diminui a potência média que é fornecida ao sistema. Esse tipo de controlador fornece ao sistema uma entrada proporcional ao erro entre a entrada e a saída, porém a sua sensibilidade ao erro é maior, o que causa e perda de potência fornecida.

Já o controlador PID utiliza o controle proporcional, mas com ações integrativas e derivativas juntos. Estes outros tipos de controle ajudam o sistema a compensar as alterações e melhoram a resposta ao erro de entrada. Com este tipo de controlador também é possível diminuir a resposta transitório do sistema e para sistemas térmicos, onde a resposta transitória costuma ter um período de tempo maior, essa ação otimiza o sistema. Assim, para que o nosso projeto tenha a melhor resposta no controle em malha fechada, é preciso projetar um controlador PID.

5.1 CONTROLADOR PID

Apresentando em detalhes o controlador PID, no domínio da frequência, temos o seguinte diagrama de blocos do projeto de controle, na Fig. 5.2. Podemos observar nessa figura que o bloco PID(s) foi substituído por vários outros blocos representando os ganhos Kp, Ki e Kd, referentes aos ganhos das operações de proporção, integração e derivação. Também há os blocos que representam as operações do PID e o ganho geral K do controlador.

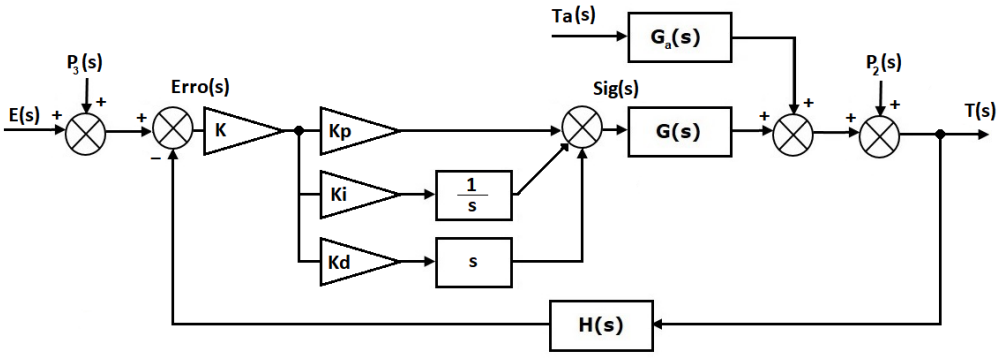


Figura 5.2. Diagrama de blocos de malha fechada com PID(s).

Estas informações do PID, sobre os ganhos e configurações do diagrama de blocos do controlador, são importantes para utilizarmos uma *toolbox* do MATLAB para projetar o controlador. A *toolbox* que utilizamos para o projeto foi o Sisotool e para abrirmos ela com a função de transferência que desejamos, usamos o seguinte comando: "*sisotool(tf_saida)*". O MATLAB abre as janelas mostradas na Fig. 5.3.

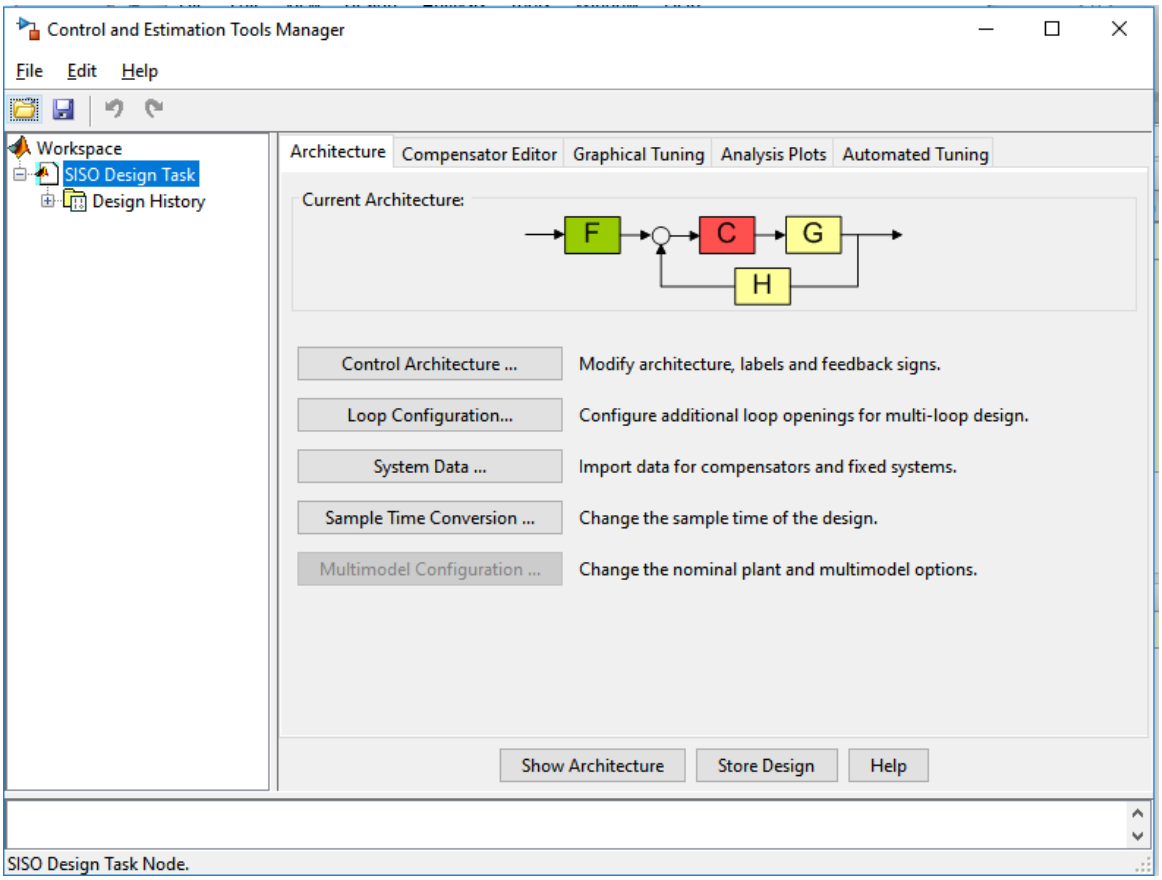
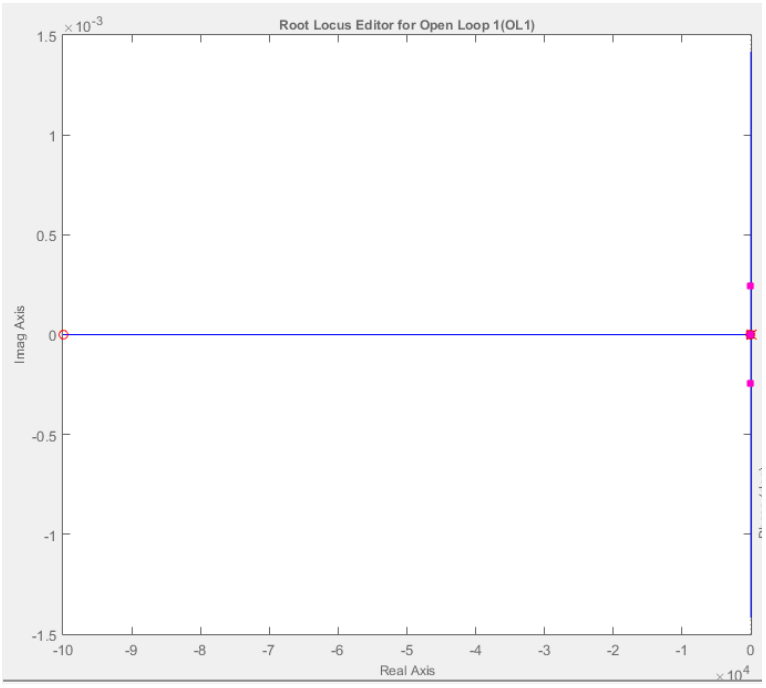


Figura 5.3. Sisotool.

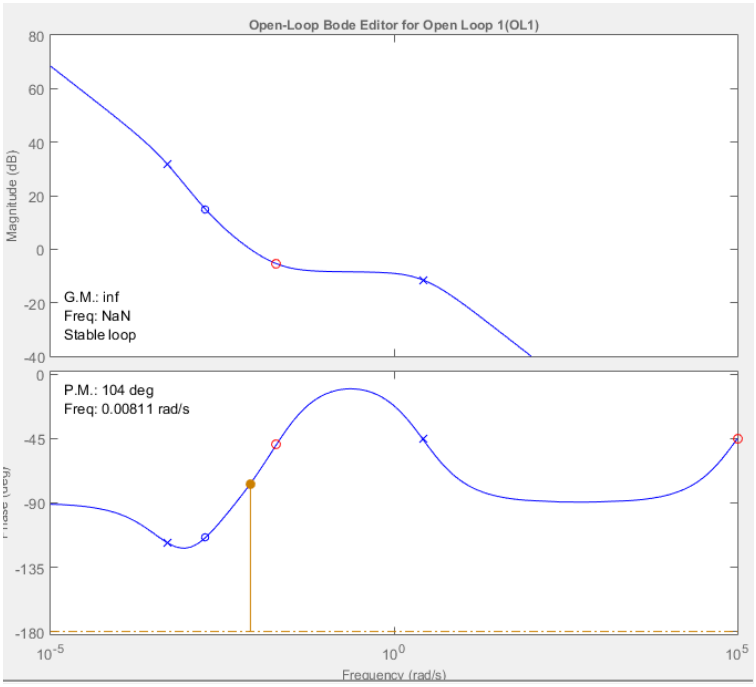
A partir dessa *toolbox*, podemos projetar o controlador e observar a resposta do processo tanto quanto os polos e zeros, quando os diagramas de bode. Como padrão, temos que H(s) do nosso sistema é unitário, ou seja, H(s) = 1. Encontramos que a melhor configuração de ganhos do controlador gera a função de transferência apresentada na Eq. 30.

$$PID(s) = \frac{0{,}00052s^2 + 1{,}011s + 0{,}01944}{s} \tag{31}$$

Para chegar à essa expressão, utilizamos a função *Automated Tuning* da própria *toolbox* a fim de criarmos os primeiros parâmetros, entretanto, não baseamos o nosso controlador somente à essa ferramenta. Para ajustar os ganhos e as posições dos zeros, temos uma ferramenta no *Sisotool* que permite que façamos ajustes através do gráfico do plano *s* e através do diagrama de Bode. O resultado gráfico dos ajustes, que resultaram na Eq. 30, está apresentado na Fig. 5.4.



(a)



(b)

Figura 5.4. (a) Plano *s* apresentado pelo *Sisotool*. (b) Diagrama de Bode.

A implementação deste controlador, no projeto, foi feita através do próprio MATLAB, assim reescrevemos o código que tínhamos para o projeto em malha aberta, conforme mostrado no Apêndice E. Antes o código gerava um sinal de referência para enviar para o microcontrolador e este enviava um sinal para os atuadores. Agora, o que nos interessa não é mais aplicar um sinal de referência, mas a partir da referência, que gerarmos, aplicar um sinal de erro no controlador, ou seja, o sistema lerá os parâmetros de saída e comparará com a referência e

irá gerar um novo sinal, chamado de erro, que é a diferença entre a referência do sistema e a sua saída.

O período de amostragem do sistema se manteve em torno de 10 segundos. Contudo, cada amostra é processada no MATLAB para gerar o sinal de erro, que é enviado ao microcontrolador. Assim, quando o *software* faz uma leitura, o tempo gasto para enviar o sinal de erro para o Arduino é maior.

Devemos destacar que o projeto de controlador PID foi feito somente para a função de transferência do sensor de saída do tubo metálico. O projeto foi feito dessa maneira pois a temperatura que queremos controlar é a temperatura de saída do tubo metálico justamente para controlarmos o projeto com o atraso de transporte ao longo deste tubo. As consequências deste projeto de controle serão apresentadas a seguir.

5.2 CÁLCULO DO SINAL DE ERRO DO SISTEMA

Para a compreensão do projeto do controlador PID, é importante destacar que o cálculo do sinal de Erro dado ao PID é feito com base na temperatura ambiente e na temperatura de referência.

O que foi determinado, através dos códigos do MATLAB, para que o sinal de PID fosse calculado, é que o sinal de Erro, mostrado na Fig. 5.2, fosse composto pela soma da temperatura de referência (E) e da temperatura ambiente (T_A) menos a temperatura fornecida pelo sistema (T). A Eq.

5.3 RESULTADO DO CONTROLADOR

Como comentado, no subcapítulo anterior, o nosso projeto em malha fechada foi feito apenas para a função de transferência do sensor que fica na saída do tubo metálico, uma vez que um dos objetivos do nosso trabalho é analisar o processo de transferência de calor com atraso de transporte.

Quando analisamos o sistema em malha aberta, o transporte de calor ao longo do tubo metálico não possuía um atraso significativo, ou seja, assim que um degrau na entrada de referência acontecia, a temperatura na saída do tubo já começava a aumentar. Também em malha aberta, o transitório possuía uma pequena demora e o sistema era sujeito a perturbações ambientes, conforme foi mostrado na Fig. 4.3. Ao implementar o projeto em malha fechada, aplicamos um sinais de referência para ver como a planta de controle agiria e essa referência atua em conjunto com a temperatura ambiente de forma aditiva, ou seja, a temperatura ambiente, que já está atuando no sistema, é acrescida da temperatura de referência. Assim, quando a referência do sistema é nula, a saída do tubo metálico deve ser

a própria temperatura ambiente. Quando a temperatura de referência for 30°C, a saída do sistema deverá ser a temperatura ambiente mais 30°C . O resultado está apresentado na Fig. 5.5.

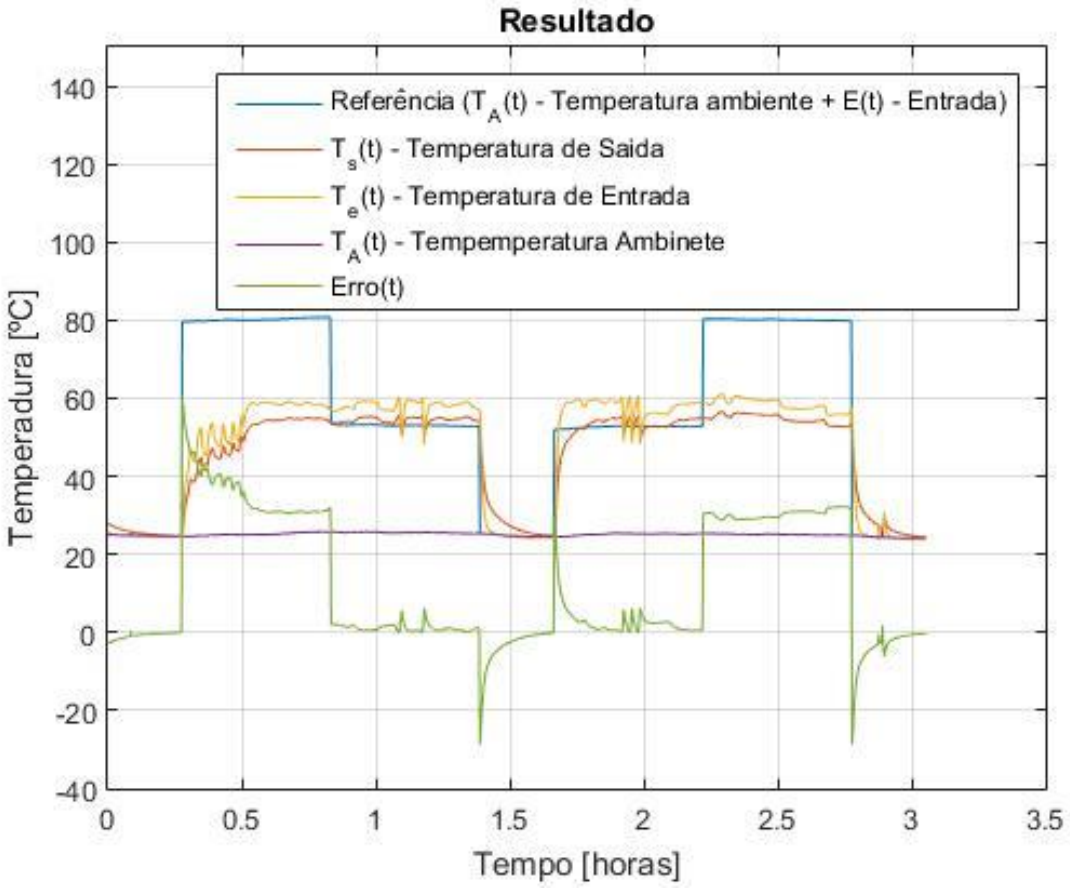


Figura 5.5. Teste em malha fechada com 1 resistência.

Chama a atenção o fato de que quando a entrada de referência muda de 80 para 50, a temperatura de saída do tubo metálico não altera. O que acontece na planta, quando a entrada muda, é que o erro passa de 50, para zero (apenas a temperatura ambiente). Assim, podemos concluir que a planta de controle não consegue zerar o erro em regime para essa entrada, em outras palavras, na atual configuração do sistema, a planta de controle não consegue aquecer o bastante para que o erro chegue à zero.

Devido à essa saturação, do aparato, o sinal calculado do PID pedia para aplicar na resistência uma tensão muito além do que poderia ser fornecida. Podemos observar esse sinal na Fig. 5.6, em que ele chega a gerar um valor de 300% da tensão RMS na resistência. Como o sistema pode aplicar no máximo 100% da tensão RMS na resistência, podemos observar que próximo a amostra 200, o sinal que efetivamente é enviado ao Arduino é de 100%. Além disso, quando o valor do Erro era menor que zero, era preciso que a tensão aplicada na resistência fosse completamente desligada, pois o PID não é capaz de gerar um

sinal para resfriar o ar, logo, podemos observar quando o Erro é negativo, o sinal que efetivamente é enviado ao Arduino é nulo.

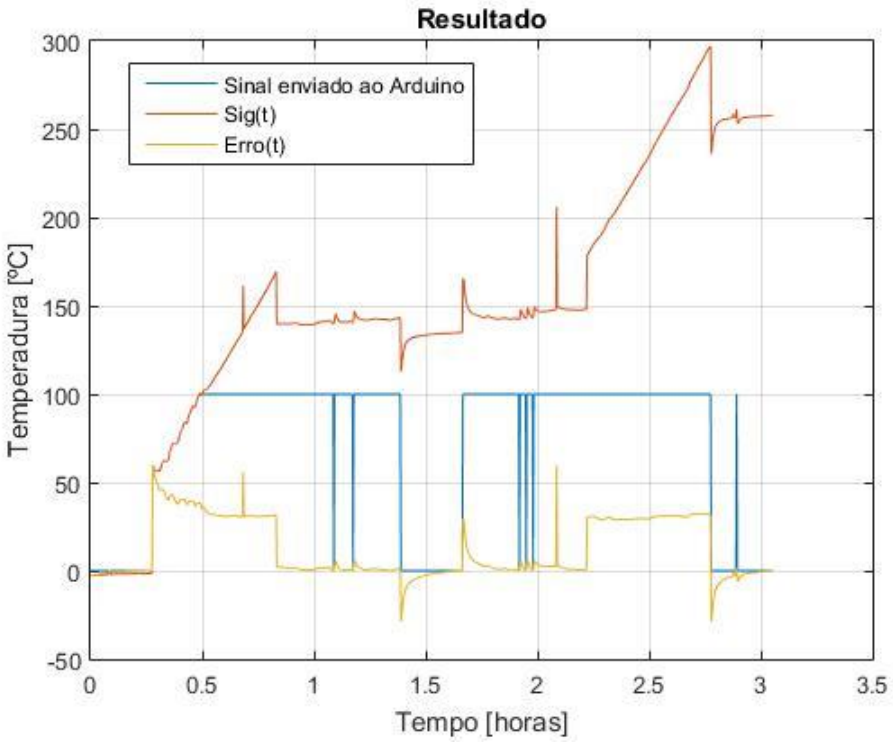


Figura 5.6. Análise do sinal enviado ao Arduino.

Todavia, o nosso aparato experimental foi construído com 3 resistências e assim, ele possui a capacidade de gerar mais calor. Dessa forma, se colocarmos outra resistência para a entrada de 60%, a planta de controle poderá fornecer um erro nulo. Porém, não podemos ligar mais uma resistência no momento em que a planta não consegue zerar o erro e depois desligar, precisamos deixar a planta de controle funcionando com duas resistências agora.

Um novo teste foi feito, com duas resistências ligadas e o resultado é apresentado na Fig. 5.7. Nesse teste, podemos observar que a planta de controle aqueceu mais para a entrada de 60% e apresentou a mesma temperatura, que o teste anterior, para e entrada de 30%. Ou seja, podemos aproveitar o modelo que identificamos até o momento e também o controlador PID.

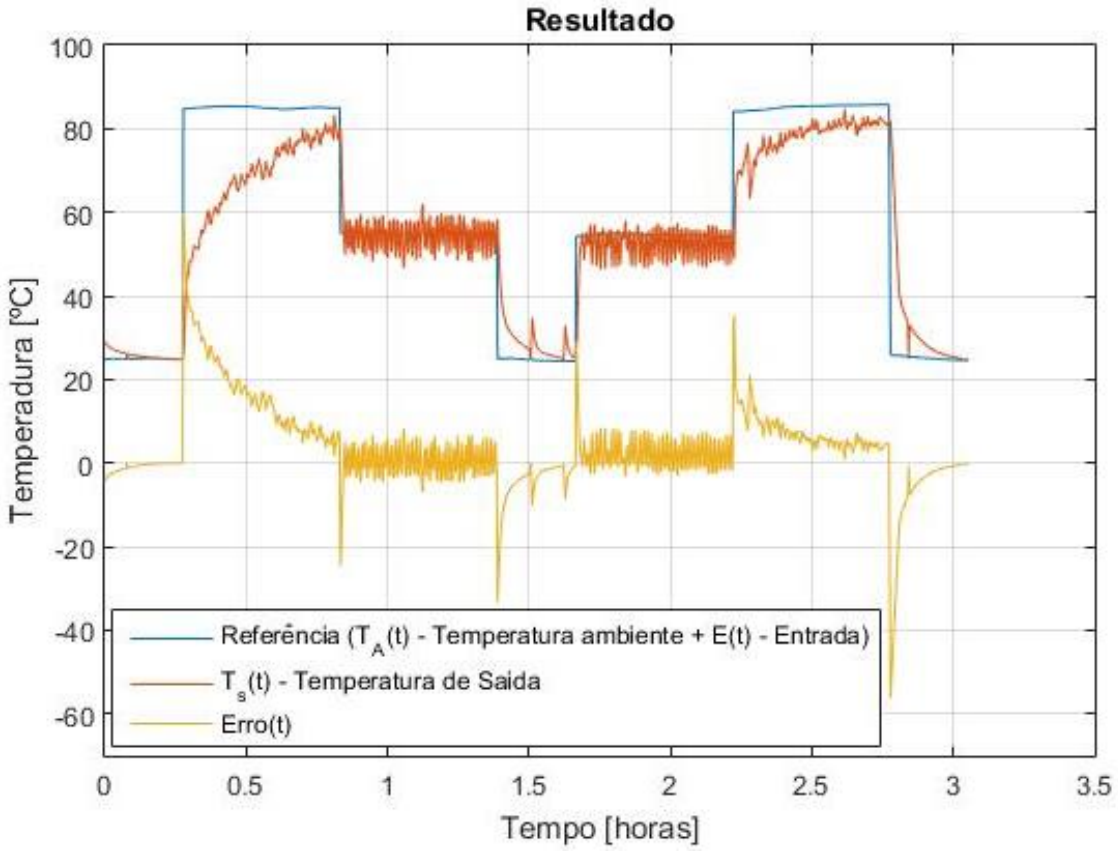
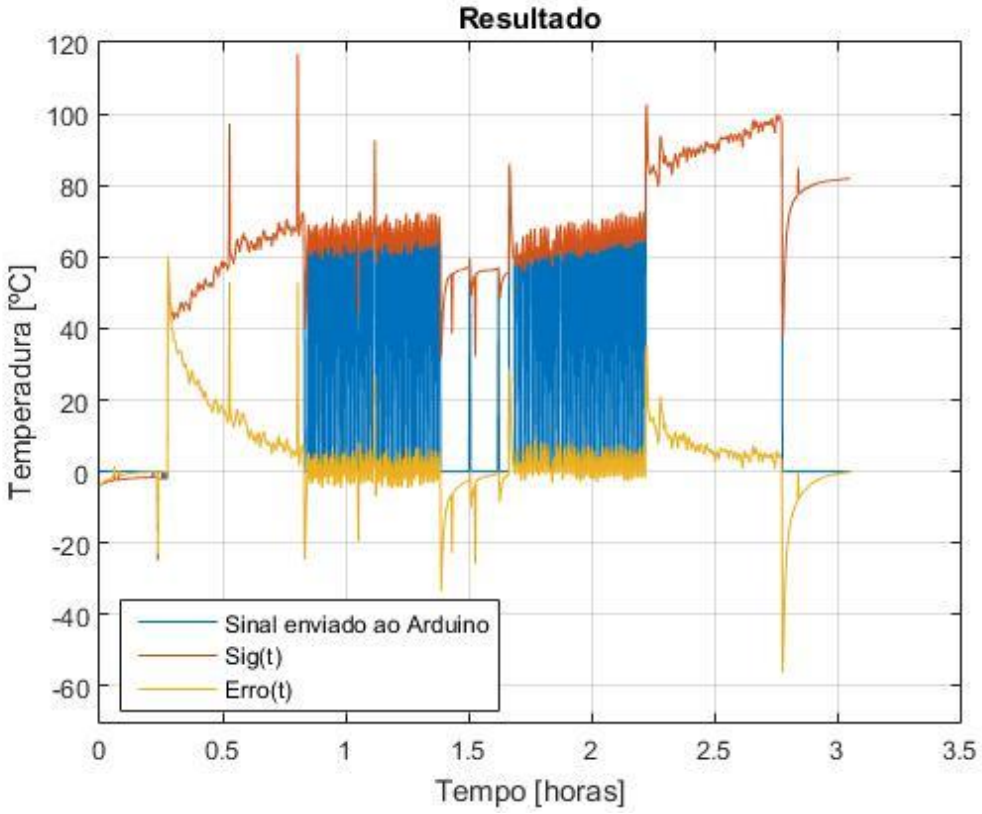


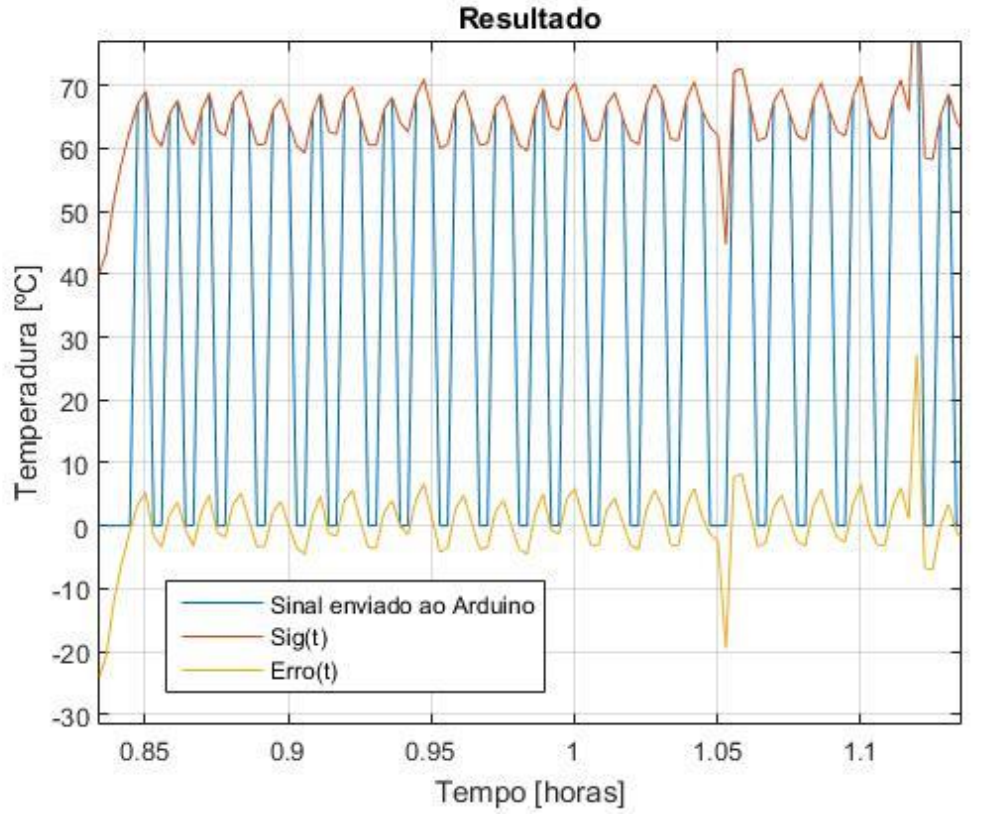
Figura 5.7. Teste de malha fechada com duas resistências.

Podemos observar, na Fig. 5.7, que o erro demora para chegar em zero, pois o sistema em questão possui um transitório mais demorado. Porém, quando o sistema chega ao erro nulo, há variações no valor da temperatura, que podem ser bem observadas quando no gráfico nos períodos de aproximadamente 1 horas e 2 horas. Essas variações acontecem devido a uma característica da programação feita no microcontrolador, pois, ao executar os comandos para calcular o ângulo de disparo para o aquecimento do sistema, o microcontrolador não pode fazer a leitura dos sensores e quando. Quando a leitura dos sensores é feita, o microcontrolador interrompe o pulso responsável pelo disparo do TRIAC.

Neste caso, diferente do que aconteceu na Fig. 5.6, o sinal enviado ao Arduino se comportou diferente e está apresentado na Fig. 5.8 (a). Como o sistema não saturou, o Erro enviado ao PID chegou à zero e ficou variando em torno deste ponto, assim, toda vez que o sinal de Erro era negativo, o sinal Sig aplicado na resistência era desligado e quando era positivo, o sinal do enviado ao Arduino era o sinal do próprio PID. Esse chaveamento é apresentado na Fig. 5.8 (b), onde podemos observar o sinal enviado ao Arduino desligando para valores de erro menores que zero e igual ao valor do PID quando o erro era maior que zero.



(a)



(b)

Figura 5.8. (a) Sinal enviado ao Arduino com a planta funcionando com duas resistências. (b) Detalhe do sinal enviado ao Arduino nos instantes em que o Erro oscila em torno do zero.

É esperado que o sinal de Erro enviando ao PID fique oscilando em torno do ponto nulo, dada uma margem para tal oscilação. Neste projeto, não foi determinada a margem de variação do erro em regime permanente. As oscilações em torno do erro provocam essas oscilações no acionamento da carga resistiva, assim, consideramos essa característica é intrínseca do sistema.

Apesar disso, quando o sistema está resfriando, como podemos observar no período de 1 hora e meia, as oscilações não acontecem. Esse fato é dado também à programação feita no MATLAB para o controle em malha fechada, pois quando acontece o resfriamento, o erro é negativo, como se pode observar. Como este processo térmico não foi projetado para resfriar o ar que circula no tubo metálico, quando o sistema detecta o erro negativo, ele interrompe o aquecimento.

CAPÍTULO 6 – CONCLUSÃO

Nesse projeto construímos um aparato experimental para analisarmos um processo térmico onde existe um atraso no transporte da massa térmica. Baseado no conceito de Katsuhiko Ogata [1], sobre uma planta, passamos a designar o aparato experimental como planta de controle.

A planta de controle foi feita com uma resistência elétrica, que possui um dispositivo baseado no TRIAC, para controle de ângulo de disparo, juntamente com um motor DC, que era controlado por PWM e uma ponte H, para amplificar o sinal do PWM a fim de atender a necessidade do motor. O ar aquecido era soprado por um tubo metálico de dois metros de comprimento e em cada extremidade do tubo, há um sensor de temperatura cujo sinal enviado é digital, uma vez que este sensor tem em sua construção uma série de componentes, que foram apresentados, para enviar este sinal. Também há um sensor de temperatura que mede a temperatura ambiente. Todos esses componentes eletrônicos foram interligados através do microcontrolador Atmega 328, integrado no Arduino UNO.

Com os testes realizados, podemos concluir que a planta de controle poderia dissipar mais calor que o sensor poderia mensurar, dessa forma tomamos o cuidado para que a faixa de temperatura, que trabalhamos, não superasse a temperatura máxima do sensor.

Também baseados nos materiais que empregamos na construção do aparato experimental, podemos concluir que o tubo metálico utilizado possui uma capacitância térmica. Essa capacitância poderia continuar fornecendo calor no ar de dentro do tubo mesmo depois que a resistência fosse desligada. No mesmo princípio, observamos que a temperatura ambiente influencia na temperatura inicial do sistema, antes mesmo de aplicarmos qualquer tensão na resistência. Logo, podemos concluir que o sistema, que deveríamos modelar, possuía mais de uma entrada.

Baseado no fato que nosso modelo possui mais de uma entrada, o modelo que melhor se aproximou do real possui dois polos e um zero. Para confirmar este fato, apresentamos nos seções de validação, dados que corroboram para o modelo apresentado com um erro entre o modelo e o real pequeno. Nos baseamos nas observações de Luiz Carlos Felício [5] para concluir que o modelo obtido não seria 100% igual ao real e assim consideramos as equações, no domínio da frequência validas para a nossa planta.

Para o controle em malha fechada, projetamos um controlador PID com o auxílio da ferramenta SISOTOOL do MATLAB. O sinal de erro, para esse modelo passou a analisar a temperatura ambiente dinamicamente, diferente do modelo em malha aberta, onde foi considerada a temperatura ambiente como uma constante e as variações dessa temperatura como perturbações do sistema.

Podemos concluir que o modelo com atraso de transporte, que projetamos e construímos, possui uma constante de tempo longo e isso reflete no controlador em malha fechada, que demora mais que o modelo em malha aberta para chegar ao regime permanente.

6.1 TRABALHOS FUTUROS

Como proposta para futuras análises dessa planta de controle, sugerimos analisar um controlador em espaço de estados. Como este aparato experimental está sujeito a uma entrada constante, que é a temperatura ambiente, a análise através do espaço de estados poderá ajudar a criar um modelo mais fiel a realidade considerando a temperatura ambiente.

Também como proposta, sugerimos utilizar outros sensores de temperatura, com uma escala maior, para que seja possível analisar a planta em temperaturas mais elevadas. Isso pode caracterizar melhor alguns fenômenos térmicos, como a capacitância térmica do tubo metálico e consequentemente pode alterar a identificação do modelo.

Baseado nas análises feitas no sistema em malha fechada, o processo térmico em questão não foi projetado para fazer o resfriamento do ar que circula no tubo metálico. Dessa forma, sugerimos que em futuro trabalhos se faça a identificação do sistema para a resposta do motor ao degrau e na sequência, o projeto em malha fechada do sistema para essa entrada também. Ou seja, que a abordagem dada ao projeto com uma entrada de referência para a resistência térmica e uma entrada de referência para o motor DC.

Ainda analisado o sistema em malha fechada, observamos um problema no sinal gerado pelo PID para enviar ao microcontrolador, conforme mostrado na Fig. 4.6. O controlador estimou valores de tensão para serem aplicados na resistência além do que o aparato experimental poderia fornecer, uma vez que há um limite de tesão que pode ser aplicada na resistência. Dessa forma, é sugerido à trabalhos futuros, um projeto de controlador para a resistência elétrica, que possa superar o problema do controlador estimar valores além de 100% da tensão aplicada na carga.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] K. Ogata, Engenharia de Controle Moderno, São Paulo: Pearson Prentice Hall, 2010.

[2] MathWorks, MATLAB: The Language of Technical Computing, Natick: The MathWorks, Inc, Version 5.

[3] FILIPEFLOP COMPONENTES ELETRÔNICOS EIRELI , “http://www.filipeflop.com/,” Componentes Eletrônicos, [Online]. Available: http://www.filipeflop.com/. [Acesso em 03 06 2017].

[4] M. H. Rashid, Eletrônica de Potência: circuitos, dispositivos e aplicações, São Paulo: Makron Books, 1999.

[5] L. C. Felício, Modelagem da dinâmica de sistemas e estudo da resposta, São Carlos: RiMa, 2010.

[6] M. d. S. Lauretto, “Escola de Artes, Ciências e Humanidades da Universidade de São Paulo,” [Online]. Available: http://www.each.usp.br/lauretto/ACH0021_2015/aula06.pdf. [Acesso em 17 Junho 2017].

[7] R. C. D. H. Bishop, Sistemas de Controle Modernos, Rio de Janeiro: LTC, 8 edição.

[8] E. d. S. Zambaldi, “Controle automatizado de fornos para tratamento térmico em aços,” *Dissertação (mestrado acadêmico) – Universidade Federal de Lavras,* vol. 1, nº 1, p. 115, 2016.

[9] I. Barbi, Eletrônica de potência, 5ª edição ed., Florianópolis: Do Autor, 2005.

[10] N. S. Nise, Control systems engineering, 6th edition ed., Rio de Janeiro: LTC, 2013.

[11] Arduino, “Arduino - Genuino,” Arduino , [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardMega2560#. [Acesso em 27 Junho 2016].

[12] A. Bauchspiess, “Página Pessoal,” Universidade de Brasília, [Online]. Available: http://www.ene.unb.br/adolfo/Lectures/ISD.html. [Acesso em 25 06 2017].

APÊNDICES

Apêndice A

```
/*
 * Código para APARATO EXPERIMENTAL PARA CONTROLE DE SISTEMAS TÉRMICOS COM ATRASO DE TRANSPORTE
 *
 * Universidade de Brasília
 * Faculdade de Tecnologia
 *
 * Trabalho de Conclusão de Curso
 * 1 semestre - 2017
 *
 * Davi Menezes Visintainer
 */
#include <OneWire.h>
#include <DallasTemperature.h>

//=====================================================================
// SENSOR DS18B20

#define ONE_WIRE_BUS 8                // Porta do pino de sinal do DS18B20
OneWire ds(ONE_WIRE_BUS);             // Define uma instancia do oneWire para comunicacao com o sensor
DallasTemperature sensors(&ds);
DeviceAddress sensor1 = {0x28, 0xFF, 0x77, 0x2A, 0x70, 0x16, 0x05, 0x89};
DeviceAddress sensor2 = {0x28, 0xFF, 0xDF, 0x43, 0x93, 0x16, 0x04, 0xFA};
DeviceAddress sensor3 = {0x28, 0xFF, 0xA9, 0x3D, 0x93, 0x16, 0x04, 0xB6};

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

int select=0;                 //VARIÁVEL USADA PARA SELEÇÃO (MENU)
int velocit;
int j = 0;                    //Variável para o vetor
char b[3];                    //Vetor que lê a entrada da porta serial
unsigned int pwm;
unsigned int t = 0;
unsigned int tempo = 0;
int PWM = 9;
float t1 = 0;
float t2 = 0;
float t3 = 0;

//PINOS ENTRADA-SAIDA
int sync = 13;                //sinal de sincronismo com a rede elétrica (entrada)
int trigger = 12;             //pino que aciona o MOC3020 (saída)
int motor = 8;
//MOTOR A
int IN1 = 2 ;
int IN2 = 4 ;
int velocidadeA = 3;
//MOTOR B
int IN3 = 6 ;
```

```
int IN4 = 7 ;
int velocidadeB = 5;

double timeToWait = 8333;  //tempo de espera após a passagem pelo zero para disparar o TRIAC (em
microssegundos)
int pulseWidth;                 //tempo de duração do pulso de disparo
int state = 0;                  //status do sinal de sincronismo
int previousState = 0;          //estado anterior do sinal de sincronismo
const int halfCycle = 8333;     //tempo em microssegundos de um semiciclo da rede elétrica (constante)
double potencia;                //Potência em porcentagem na carga
int i = 0;                      //Variável para o vetor
char a[4];                      //Vetor que lê a entrada da porta serial

void setup() {

  pinMode(IN1,OUTPUT);
  pinMode(IN2,OUTPUT);
  digitalWrite(IN1,HIGH);
  digitalWrite(IN2,LOW);
  pinMode(IN3,OUTPUT);
  pinMode(IN4,OUTPUT);
  digitalWrite(IN3,HIGH);
  digitalWrite(IN4,LOW);
  pinMode(velocidadeA,OUTPUT);
  pinMode(velocidadeB,OUTPUT);

  pinMode(trigger, OUTPUT);
  pinMode(sync, INPUT);
  digitalWrite(trigger, LOW);  //assegura que a saída irá iniciar em zero

  Serial.begin(9600);
  sensors.begin();
  Serial.println("SISTEMA PRONTO");
  Serial.println("ESCOLHA QUAL VARIÁVEL ALTERAR:");
  Serial.println("1 - ALTERAR A VELOCIDADE DO MOTOR;");
  Serial.println("2 - ALTERAR A TENSÃO NA CARGA.");

}

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

void printTemperature(DeviceAddress deviceAddress)
{
  float tempC = sensors.getTempC(deviceAddress);
  if (tempC == -127.00)
  {
    Serial.print(tempC);
  }
  else
  {
//    Serial.print("C: ");
    Serial.print(tempC);
  }
}
```

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

void loop() {

boolean turnOff = 0;              //flag para indicar quando deve-se desligar a carga
boolean maxPower = 0;          //flag para indicar quando deve-se transmitir potência máxima para a carga

if (Serial.available()){
  select = Serial.read();
  Serial.println(select);
}
if (select == 49){
  Serial.println("VOCE SELECIONOU A OPCAO 1");
  funcao1();
}
if (select == 50){
  Serial.println("VOCE SELECIONOU A OPCAO 2");
  if (pwm>=38){
    funcao2();
  }else{
    Serial.println("PRIMEIRO LIGUE O MOTOR");
    funcao1();
  }
}
if (select>50){
  Serial.println("Escolha a opcao 1 ou 2");
  select=0;
}
t = t + 1;
if (t >= 30000){
  sensors.requestTemperatures();
  printTemperature(sensor1);
  Serial.print(", ");
  printTemperature(sensor2);
  Serial.print(", ");
  printTemperature(sensor3);
  Serial.println();
  t = 0;
}
if(timeToWait <= 300)              //Habilita flags de acordo com o valor de timeToWait
    maxPower = 1;
else if(timeToWait >= 7900)
    turnOff = 1;
if( (!maxPower)&&(!turnOff) ){     //Calcula tempo de duração do pulso de disparo
    pulseWidth = halfCycle - timeToWait - 1000;
    if(pulseWidth < 0)
        pulseWidth = 0;
}
state = digitalRead(sync);         //Le estado do sinal de sincronismo
if(maxPower)                       //Se a potência for máxima, deixa o trigger acionado direto
    digitalWrite(trigger, HIGH);
else if( (turnOff == 0)&&(state != previousState) ){      //Se o estado do sinal de sincronismo mudou e a carga
não deve ficar apagada
    previousState = state;                  //armazena estado atual
    delayMicroseconds(timeToWait);          //espera tempo para disparo
```

```
   digitalWrite(trigger, HIGH);              //aciona sinal de disparo (trigger)
   delayMicroseconds(pulseWidth);            //aguarda largura de pulso
   digitalWrite(trigger, LOW);               //desliga pulso de disparo (trigger)
}//end if
}//FIM DO LOOP
//=====================================================================
void funcao1(){
Serial.println("INFORME O VALOR PERCENTUAL QUE DESEJA APLICAR NO PWM");
delay(3000);
if (Serial.available()){                     //verifica se chegou algum dado na serial
   for (int j=0; j<3; j++) {
   b[j] = Serial.read();                     //Lê o byte mais recente disponível na serial
   delay(50);
  }
   velocit = atoi(b);                        //Convertendo o vetor no inteiro
   pwm = map(velocit,0,100,0,255);           //Converte a potência no ângulo de defasagem
   Serial.print("O VALOR DE PWM CORRESPONDENTE A ESSA VELOCIDADE E: ");
   Serial.println(velocit);                  //Mostra a potência que foi colocada na porta
}
analogWrite(velocidadeA,pwm);
//OPÇÕA DE SEGURANÇA - NÃO MANTER A TENSÃO NA CARGA SEM QUE HAJA FLUXO DE AR
if (pwm == 0){
  timeToWait = 7900;
}
  select=0;
}//FIM DA FUNÇÃO 1
//=====================================================================
void funcao2(){
Serial.println("INFORME O VALOR PERCENTUAL DA TENSAO QUE SEJA APLICAR NA CARGA");
delay(3000);
if (Serial.available()){                        //verifica se chegou algum dado na serial
  for (int i=0; i<4; i++) {
  a[i] = Serial.read();                         //Lê o byte mais recente disponível na serial
  delay(50);
 }
 Serial.println(a);
  potencia = atof(a);                           //Convertendo o vetor no inteiro
  Serial.print("A TENSAO FORNECIDA SERA CORRESPONDENTE A PORCENTAGEM DE: ");
  Serial.println(potencia);                     //Mostra a potência que foi colocada na porta
  timeToWait = (-76)*potencia + 7900;
  //timeToWait = map(potencia,0,100,7900,300);      //Converte a potência no ângulo de defasagem
  Serial.println(timeToWait);
}
select=0;
}//FIM DA FUNÇÃO 2
```

Apêndice B

```
/*
 * Código para APARATO EXPERIMENTAL PARA CONTROLE DE SISTEMAS TÉRMICOS COM ATRASO DE TRANSPORTE
 * ATUALIZADO PARA CONTROLE EM MALHA FECHADA
 *
 * Universidade de Brasília
 * Faculdade de Tecnologia
 *
 * Trabalho de Conclusão de Curso
 * 1 semestre - 2017
* Davi Menezes Visintainer
 */
//============================================================
// SENSOR DS18B20
#include <OneWire.h>
#include <DallasTemperature.h>
// Porta do pino de sinal do DS18B20
#define ONE_WIRE_BUS 8
// Define uma instancia do oneWire para comunicação com o sensor
OneWire ds(ONE_WIRE_BUS);
DallasTemperature sensors(&ds);
DeviceAddress sensor1 = {0x28, 0xFF, 0x77, 0x2A, 0x70, 0x16, 0x05, 0x89};
DeviceAddress sensor2 = {0x28, 0xFF, 0xDF, 0x43, 0x93, 0x16, 0x04, 0xFA};
DeviceAddress sensor3 = {0x28, 0xFF, 0xA9, 0x3D, 0x93, 0x16, 0x04, 0xB6};
//============================================================
int select=0;                //VARIÁVEL USADA PARA SELEÇÃO (MENU)
int velocit;
//variavel auxiliar
int j = 0;                   //Variável para o vetor
char b[3];                   //Vetor que lê a entrada da porta serial
unsigned int pwm;
unsigned int t = 0;
unsigned int tempo = 0;
int PWM = 9;
float t1 = 0;
float t2 = 0;
float t3 = 0;
//PINOS ENTRADA-SAIDA
int sync = 13;               //sinal de sincronismo com a rede elétrica (entrada)
int trigger = 12;            //pino que aciona o MOC3020 (saída)
int motor = 8;
//MOTOR A
int IN1 = 2 ;
int IN2 = 4 ;
int velocidadeA = 3;
//MOTOR B
int IN3 = 6 ;
int IN4 = 7 ;
int velocidadeB = 5;
double timeToWait = 8333;    //Tempo de espera após a passagem pelo zero para disparar o TRIAC (em microssegundos)
int pulseWidth;              //tempo de duração do pulso de disparo
```

```
int state = 0;                    //status do sinal de sincronismo
int previousState = 0;            //estado anterior do sinal de sincronismo
const int halfCycle = 8333;       //tempo em microssegundos de um semiciclo da rede elétrica (constante)
int potencia;                     //Potência em porcentagem na carga
int i = 0;                        //Variável para o vetor
char a[8];                        //Vetor que lê a entrada da porta serial
void setup() {
  pinMode(IN1,OUTPUT);
  pinMode(IN2,OUTPUT);
  digitalWrite(IN1,HIGH);
  digitalWrite(IN2,LOW);
  pinMode(IN3,OUTPUT);
  pinMode(IN4,OUTPUT);
  digitalWrite(IN3,HIGH);
  digitalWrite(IN4,LOW);
  pinMode(velocidadeA,OUTPUT);
  pinMode(velocidadeB,OUTPUT);
  pinMode(trigger, OUTPUT);
  pinMode(sync, INPUT);
  digitalWrite(trigger, LOW);  //assegura que a saída irá iniciar em zero
  Serial.begin(9600);
  sensors.begin();
}
void loop() {
boolean turnOff = 0;          //flag para indicar quando deve-se desligar a carga
boolean maxPower = 0;         //flag para indicar quando deve-se transmitir potência máxima para a carga
  if (Serial.available()){
    select = Serial.read();
  }
if (select == 49){
  funcao1();
}
if (select == 50){
  if (pwm>=38){
    funcao2();
  }else{
    funcao1();
  }
}
if (select>50){
  select=0;
}
t = t + 1;
if (t >= 18000){
  sensors.requestTemperatures();
  t1 = sensors.getTempC(sensor1);
  t2 = sensors.getTempC(sensor2);
  t3 = sensors.getTempC(sensor3);
  Serial.print(t1);
  Serial.print("  ");
  Serial.print(t2);
  Serial.print("  ");
  Serial.print(t3);
  Serial.print("  ");
  t = 0;
```

```
}
if(timeToWait <= 300)                    //Habilita flags de acordo com o valor de timeToWait
   maxPower = 1;
else if(timeToWait >= 7900)
   turnOff = 1;
if( (!maxPower)&&(!turnOff) ){           //Calcula tempo de duração do pulso de disparo
   pulseWidth = halfCycle - timeToWait - 1000;
   if(pulseWidth < 0)
      pulseWidth = 0;
}
state = digitalRead(sync);               //Le estado do sinal de sincronismo
if(maxPower)                              //Se a potência for máxima, deixa o trigger acionado direto
   digitalWrite(trigger, HIGH);
else if( (turnOff == 0)&&(state != previousState) ){      //Se o estado do sinal de sincronismo mudou e a carga
não deve ficar apagada
   previousState = state;                 //armazena estado atual
   delayMicroseconds(timeToWait);         //espera tempo para disparo
   digitalWrite(trigger, HIGH);           //aciona sinal de disparo (trigger)
   delayMicroseconds(pulseWidth);         //aguarda largura de pulso
   digitalWrite(trigger, LOW);            //desliga pulso de disparo (trigger)
}//end if
}//FIM DO LOOP
//============================================================
void funcao1(){
delay(3000);
if (Serial.available()){
   for (int j=0; j<3; j++) {
   b[j] = Serial.read();
   delay(50);
  }
   velocit = atoi(b);                    //Convertendo o vetor no inteiro
   pwm = map(velocit,0,100,0,255);
}
analogWrite(velocidadeA,pwm);
//OPÇÕA DE SEGURANÇA - NÃO MANTER A TENSÃO NA CARGA SEM QUE HAJA FLUXO DE AR
if (pwm == 0){
 timeToWait = 7900;
}
 select=0;
}//FIM DA FUNÇÃO 1
```

```
//=============================================================
void funcao2(){
delay(3000);
if (Serial.available()){
  for (int i=0; i<8; i++) {
  a[i] = Serial.read();
  delay(50);
 }
  potencia = atoi(a);
  timeToWait = (-76)*potencia + 7900;
  //timeToWait = map(potencia,0,100,7900,300);   //Converte a potência no ângulo de defasagem
}
  select=0;
}//FIM DA FUNÇÃO 2
```

Apêndice C

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Código para leitura de temperaturas utilizando Arduino através da porta
% serial
% A Placa Utilizada foi o Arduino Uno.
%
% Código utilizado na Planta de Controle construída para apresentar o
% Trabalho de Conclusão de Curso
%
% Por: Davi Menezes Visintainer
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clc
clear all
close all

%% ACESSANDO A PORTA SERIAL
delete(instrfind({'Port'},{'COM3'}));       %Garantia de que a serial do
Arduino foi encerrada
s = serial('COM3', 'BaudRate', 9600);       %Parâmetros de acesso para
serial do Arduino da Planta de Controle
fopen(s);                                   %Abrindo a serial

%% DECLARAÇÃO DE VARÁVEIS

N = 2161;                                   %Número de Segundos +1 para 6
horas de teste
temp_saida     =  [];
temp_entrada   =  [];
temp_amb       =  [];
c = 1;

%% CRIAÇÃO DO SINAL DE ENTRADA PRBS

amp = 30;                                   %Amplitude do Sinal PRBS
po = 30;                                    %Ponto de operação escolhido
entrada = idinput(N,'prbs',[0,0.006],[-1 1]);
entrada = po + amp*entrada;
ent = 0;

%% ESCANEAMENTO
    pause(4);
    fprintf(s,'%i',1);
    pause(0.5);
    fprintf(s,'%i',50);

for i = 1:N
tic
    if (ent > entrada(i))                   %Avalia se o sinal PRBS mudou de
valor e envia o novo valor para o Arduino
        pause(4);
        fprintf(s,'%i',2);
        pause(0.5);
        fprintf(s,'%i',entrada(i));
        ent = entrada(i);
    elseif (ent <entrada(i))
        pause(4);
        fprintf(s,'%i',2);
        pause(0.5);
```

```
        fprintf(s,'%i',entrada(i));
        ent = entrada(i);
    end

    out = fscanf(s,'%e');               %Faz a leitura da serial e armazena
os valores dos sensores no vetor

    out1 = out(end-2);
    temp_saida(i) = out1;
    out2 = out(end-1);
    temp_entrada(i) = out2;
    out3 = out(end);
    temp_amb(i) = out3;

    display(out1)
    display(out2)
    display(out3)
toc
end

%% DESLIGAR A PLANTA

    pause(4);
    fprintf(s,'%i',2);
    pause(0.5);
    fprintf(s,'%i',50);
    pause(4);
    fprintf(s,'%i',1);
    pause(0.5);
    fprintf(s,'%i',0);
    pause(4);
    fprintf(s,'%i',2);
    pause(0.5);
    fprintf(s,'%i',0);

%% FECHAMENTO DA SERIAL
fclose(s);                              %Fechamento da Serial
delete(s);                              %Deletar o objeto Serial
clear('s');

time = 1:10:N*10;                       %Cada amostra de tempo é tomada a
cada 10 segundos
time = time/60/60;                      %Convertendo o vetor de segundos
para horas
entrada = entrada';                     %Transforma o vetor coluna no vetor
linha para os calculos de identificação

save('Ensaio');
```

```
figure(1)
plot(time,entrada,time,temp_saida,time,temp_entrada,time,temp_amb)
hold on
plot(time,temp_saida)
plot(time,temp_entrada)
plot(time,temp_amb)
hold off
grid on
title('Resultado');
xlabel('Número de amostras');
ylabel('Temperadura [°C] e %Tensão na Carga');
legend('Entrada','Temp1','Temp2','TempAmb')
```

Apêndice D

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Código para ajustar os valores lidos e identificar o processo
% A Placa Utilizada foi o Arduino Uno.
%
% Código utilizado na Planta de Controle construída para apresentar o
% Trabalho de Conclusão de Curso
%
% Por: Davi Menezes Visintainer
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clc
clear all
close all

load Teste_6horas.mat
load Ensaio.mat

%% ELIMINAÇÃO DOS OUTLIERS
IQR = iqr(temp3);
lowr=prctile(temp3,25)-1.5*IQR;
highr=prctile(temp3,75)+1.5*IQR;

%% ANALISE COM NOVOS VALORES
N = 1923;
time = 1:10:N*10;       %Cada amostra de tempo é tomada a cada 10 segundos
time = time/60/60;      %Convertendo o vetor de segundos para horas

n_time = time(temp3>lowr & temp3<highr);
n_temp3 = temp3(temp3>lowr & temp3<highr);
n_temp2 = temp2(temp3>lowr & temp3<highr);
n_temp1 = temp1(temp3>lowr & temp3<highr);
tempo = time*60;

ws1 = lsim(tf_entrada,entrada,tempo,[1 670]);
n_ws1 = ws1(temp3>lowr & temp3<highr);
ws1=n_ws1';

ws2 = lsim(tf_saida,entrada,tempo,[1 630]);
n_ws2 = ws2(temp3>lowr & temp3<highr);
ws2=n_ws2';

figure(1)
hold on
plot(time,entrada)
plot(n_time,n_temp2,n_time,n_temp3,n_time,ws2)
hold off
grid on
title('Resultado - Temperatura de Entrada');
xlabel('Tempo [H]');
ylabel('Temperadura [°C] e %Tensão na Carga');
legend('Porcentagem da tensão na Resistência','Temperatura da
Entrada','TempAmb','Função de Transferência')

figure(2)
hold on
plot(time,entrada)
plot(n_time,n_temp1,n_time,n_temp3,n_time,ws1)
```

```
hold off
grid on
title('Resultado - Temperatura de Saída');
xlabel('Tempo [H]');
ylabel('Temperadura [°C] e %Tensão na Carga');
legend('Porcentagem da tensão na Resistência','Temperatura da
Saída','TempAmb','Função de Transferêcia')
```

Apêndice E

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Código para leitura de temperaturas utilizando Arduino através da porta
% serial
% A Placa Utilizada foi o Arduino Uno.
%
% Código utilizado na Planta de Controle construída para apresentar o
% Trabalho de Conclusão de Curso
%
% Por: Davi Menezes Visintainer
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clc
clear all
close all

%% ACESSANDO A PORTA SERIAL
delete(instrfind({'Port'},{'COM3'}));       %Garantia de que a serial do
Arduino foi encerrada
s = serial('COM3', 'BaudRate', 9600);       %Parâmetros de acesso para
serial do Arduino da Planta de Controle
fopen(s);                                   %Abrindo a serial

%% DECLARAÇÃO DE VARÁVEIS
% N = 4321;                                 %Número de Segundos +1 para 12
horas de teste
N = 1100;
temp_saida       =  [];
temp_entrada     =  [];
temp_amb         =  [];
c = 1;

%% CONTROLADOR

num = [0.00052 52 1];
den = [1 0];
pi = tf(num,den);
pi = pi*0.019442;

%% CRIAÇÃO DO SINAL DE ENTRADA PRBS
ent1 = zeros(1,100);
ent2 = 60*ones(1,200);
ent3 = 30*ones(1,200);

entrada = [ent1 ent2 ent3 ent1 ent3 ent2 ent1];

ent = 0;
K = -1;
```

```
%% ESCANEAMENTO
    VelocidadeVento = 70;     % Acionamento do Motor
    pause(4);
    fprintf(s,'%i',1);
    pause(0.5);
    fprintf(s,'%i',VelocidadeVento);

for i = 1:N

tic
    if (i>2)
        time = 2:1:i;
        sig = lsim(pi,erro,time);
        K = floor(sig(i-1))
    end
    if (K>99)
        K = 100;
    end
    if (K<0)
        K = 0;
    elseif (erro(i-1)<0)
        K = 0;
    end
     if (ent > K)
        pause(1);
        fprintf(s,'%i',2);
        pause(0.5);
        fprintf(s,'%i',K);
        ent = K;
    elseif (ent <K)
        pause(1);
        fprintf(s,'%i',2);
        pause(0.5);
        fprintf(s,'%i',K);
        ent = K;
    end

    out = fscanf(s,'%e');

    out1 = out(end-2);
    temp_saida(i) = out1;
    out2 = out(end-1);
    temp_entrada(i) = out2;
    out3 = out(end);
    temp_amb(i) = out3;
    erro(i) = entrada(i)+temp_amb(i)-temp_saida(i);

    display(out1)
    display(out2)
    display(out3)
    display(erro)
toc
end

sig(i) = sig(i-1);
```

```
%% DESLIGAR A PLANTA

    pause(4);
    fprintf(s,'%i',2);
    pause(0.5);
    fprintf(s,'%i',50);
    pause(4);
    fprintf(s,'%i',1);
    pause(0.5);
    fprintf(s,'%i',0);
    pause(4);
    fprintf(s,'%i',2);
    pause(0.5);
    fprintf(s,'%i',0);


%% FECHAMENTO DA SERIAL
fclose(s);                              %Fechamento da Serial
delete(s);                              %Deletar o objeto Serial
clear('s');

time = 1:10:N*10;                       %Cada amostra de tempo é tomada a
cada 10 segundos
time = time/60/60;                      %Convertendo o vetor de segundos
para horas
entrada = entrada';                     %Transforma o vetor coluna no vetor
linha para os cálculos de identificação

save('Ensaio');

figure(1)
plot(time,entrada,time,temp_saida,time,temp_entrada,time,temp_amb,time,erro
,time,sig)
grid on
title('Resultado');
xlabel('Tempo [horas]');
ylabel('Temperadura [°C] e %Tensão na Carga');
legend('Entrada','Temp Saida','Temp Entrada','Temp Amb','Erro','Sinal');
```