UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

DEPARTAMENTO ACADÊMICO DE ENGENHARIA QUÍMICA

BACHARELADO EM ENGENHARIA QUÍMICA

GABRIEL TEIXEIRA SANTOS

PEDRO LUCAS ALVES MORAIS

RODRIGO SHOITI SIMÕES

CONSTRUÇÃO E AUTOMATIZAÇÃO DE UM REATOR CSTR PARA PRÁTICAS DE CONTROLE DE SISTEMAS ONLINE

TRABALHO DE CONCLUSÃO DE CURSO

PONTA GROSSA

2019

GABRIEL TEIXEIRA SANTOS

PEDRO LUCAS ALVES MORAIS

RODRIGO SHOITI SIMÕES

CONSTRUÇÃO E AUTOMATIZAÇÃO DE UM REATOR CSTR PARA PRÁTICAS DE CONTROLE DE SISTEMAS ONLINE

Trabalho de Conclusão de Curso como requisito parcial à obtenção do título de Bacharel em Engenharia Química, do Departamento Acadêmico de Engenharia Química da Universidade Tecnológica Federal do Paraná, Câmpus Ponta Grossa.

Orientador: Profº. Dr. Everton Moraes Matos

PONTA GROSSA

2019

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Ponta Grossa
1.1.1 Curso de Engenharia Química

TERMO DE APROVAÇÃO

CONSTRUÇÃO E AUTOMATIZAÇÃO DE UM REATOR CSTR PARA PRÁTICAS DE CONTROLE DE SISTEMAS ONLINE

por

Gabriel Teixeira Santos
Pedro Lucas Alves Morais
Rodrigo Shoiti Simões

Monografia apresentada no dia 02 de setembro de 2019 ao Curso de Engenharia Química da Universidade Tecnológica Federal do Paraná, Campus Ponta Grossa. Os candidatos foram arguidos pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

_____
Prof. Dr. Cesar Augusto Canciam
(UTFPR)

_____
Profa. Dra. Maria Regina Parise
(UTFPR)

_____
Prof. Dr. Everton Moraes Matos
(UTFPR)
Orientador

_____
Profa. Dra. Juliana de Paula Martins
Responsável pelo TCC do Curso de Engenharia Química

- O Termo de Aprovação assinado encontra-se arquivado na Secretaria Acadêmica -

AGRADECIMENTOS

Gostaríamos de registrar nossos agradecimentos e dedicar esse trabalho a todos que de alguma forma colaboram para sua realização, tanto com conhecimento como orientação durante momentos de dificuldades.

Agradecemos em especial ao Prof. Dr. Everton Moraes Matos por toda orientação e dedicação a elaboração deste trabalho. A nossas famílias pelo apoio e paciência durante esse percurso. A todos nossos companheiros de laboratórios, em especial ao Julian José de Brito e ao João Felipe Lopes Iamarino pelas dicas tão valiosas durante sua realização.

E por último, mas não menos importante a todos os professos responsáveis por nos transmitir tanto conhecimento durante os cinco anos do curso que sem dúvidas foram indispensáveis para a realização de todo o estudo presente nesse trabalho.

RESUMO

SANTOS, Gabriel Teixeira; MORAIS, Pedro Lucas Alves; SIMÕES, Rodrigo Shoiti **Construção e automatização de um reator CSTR para práticas de controle de sistemas online**. 2019. 57 f.. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Química) - Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2019.

O presente trabalho foi desenvolvido tendo como objetivo a construção de um reator enjaquetado CSTR utilizando um microprocessador Arduino® para controle de temperatura e nível, um microprocessador ESP32 da marca DOIT para a comunicação online. Adicionalmente, foi construída caixa de isopor com placas de Peltier e dissipadores para atuar como sistema de refrigeração. Os experimentos realizados mostraram resultados consistentes para um controlador proporcional de nível e temperatura. Apesar de resultados coerentes, o sistema de resfriamento não atingiu potência o suficiente para garantir sua eficiência em testes longos.

Palavras-chave: Automatização. Controle de nível. Controle de temperatura. Arduino. Internet das Coisas.

ABSTRACT

SANTOS, Gabriel Teixeira; MORAIS, Pedro Lucas Alves; SIMÕES, Rodrigo Shoiti **construction and automation of a CSTR reactor for control practices with online communication**. 2019. 57 p.. Work of Conclusion Course (Graduation in Chemical Engineering) - Federal University of Technology - Paraná. Ponta Grossa, 2019.

The present paper was developed with the objective of constructing a CSTR reactor with a cooling jacket using an Arduino® microprocessor for level and temperature control and a DOIT brand ESP32 microprocessor for online communication. Additionally, a styrofoam box was built with Peltier boards and heat sinks to act as a refrigeration system. The experiments showed consistent results for a proportional control of level and temperature. Although the coherent results, the refrigeration system have not achieved the necessary potency to ensure its efficiency in long tests.

Keywords: Automation. Level Control. Temperature Control. Arduino. Internet of Things.

LISTA DE ILUSTRAÇÕES

LISTA DE SÍMBOLOS

K_p - Constante de proporcionalidade
K_{cr} - Constante de proporcionalidade crítica
F - Faraday
Ω - Ohm
P_{cr} - Período crítico
T_i - Tempo integrativo
T_d - Tempo derivativo
V - Volts
CSTR - *Continuous stirred-tank reactor model*
P - Controlador proporcional
PI - Controlador proporcional-integral
PID - Controlador proporcional-integral-derivativo
EVA - Espuma vinílica acetinada

SUMÁRIO

2 INTRODUÇÃO

O controle de processos é uma área de conhecimento de fundamental importância para a formação de engenheiros químicos, que consiste no monitoramento e manipulação de variáveis, visando a segurança e conformidade de especificações exigidas. No curso de Engenharia Química da UTFPR - Câmpus Ponta Grossa, dentre as disciplinas que compõem a grade curricular, tem-se a disciplina de Laboratório de Engenharia Química 3, na qual são realizados experimentos de controle de processos aplicados a um reator CSTR.

Os laboratórios atribuídos ao curso possuem um equipamento que possibilitaria os ajustes de temperatura, nível e pH para as práticas de controle, que atualmente não se apresenta em condições de uso. Dessa necessidade, teve-se a ideia da construção de um reator enjaquetado com controle de temperatura, nível e agitação por intermédio do microprocessador Arduino® para a realizações de aulas práticas. Além das funções convencionais de um sistema controlado, o reator faz uso de um circuito eletrônico para comunicação via *Wifi.*

No presente estudo, fez-se a construção de um reator cilíndrico enjaquetado CSTR de aço inox e utilizou-se sensores para a medir variáveis e controlá-las eletronicamente utilizando o microprocessador Arduino® e permitindo uma visualização por plataforma online (*Cayenne*).

2.1 OBJETIVOS

2.1.1 Objetivo Geral

Construir um reator CSTR enjaquetado de escala laboratorial com controle de volume e de temperatura com conexão online via *Wifi.*

2.1.2 Objetivos Específicos

- Construir um reator CSTR enjaquetado de aço inox 304 cilíndrico, com agitador e aberto;

- Utilizar sensores eletrônicos para a interação entre o sistema físico e virtual;
- Utilizar um microprocessador Arduino® para implementar um sistema de controle de volume e temperatura;
- Implementar comunicação via *Wifi*;

2.2 ESTRUTURA DO TRABALHO

O trabalho está dividido em 5 Capítulos. O Capítulo 2 apresenta a Fundamentação Teórica utilizada no desenvolvimento do trabalho. O Capítulo 3 trata toda a metodologia com a qual o trabalho foi realizado. O Capítulo 4 apresenta resultados e discussão dos experimentos. E por fim, o Capítulo 5 descreve a conclusão do trabalho, assim como propõe sugestões na continuidade aos estudos.

3 FUNDAMENTAÇÃO TEÓRICA

3.1 REATORES CSTR

Existem três tipos de reatores que são operados em regime estacionários, que são os reatores contínuos com tanque agitado ou simplesmente CSTR (*Continuous Stirred-Tank Reactor*), os reatores com escoamento empistonado e os reatores com leito fixo. Os reatores com escoamento empistonado são geralmente utilizados em reações gasosas, enquanto os reatores CSTR são aplicados em reações líquidas (FOGLER, 2009). A Figura 1 apresenta um modelo ilustrativo de um reator CSTR encamisado.

Figura 1 - Reator CSTR

Fonte: FÁBREGA (2012)

Além de serem operados em estado estacionário com escoamento contínuo, os reatores CSTR, devido ao seu tanque agitado, possibilita considerá-lo perfeitamente misturado. Isso garante que todas as propriedades dentro do reator sejam consideradas uniformes independente do tempo ou posição, tais como concentração, temperatura, pH e velocidade de reação. Isso permite que a medida de cada uma dessas propriedades possa ser feita em qualquer ponto de um reator CSTR (FOGLER, 2009).

3.2 SISTEMAS DE CONTROLE

O controle automático é um campo inerente a qualquer área da engenharia e da ciência. Um sistema pode ser definido como uma série de componentes que atuam juntos para um único propósito. Ao unir-se ambas definições, o termo sistema de controle obtém o significado de uma série de componentes tais como sensores, processadores e outros elementos que visam controlar um processo (OGATA, 2010).

Existem alguns modos de controlar um sistema, dentre eles os mais usuais são os sistemas em malha aberta e fechada. Um sistema em malha aberta, de acordo com OGATA (2010), não possui realimentação de informações. Isso quer dizer que as entradas são predefinidas por outros parâmetros que não são saídas do processo.

Por outro lado, uma malha fechada requer essa realimentação que não existe na malha aberta. Um sistema de malha fechada objetiva controlar um parâmetro do sistema (variável controlada) por meio de outro parâmetro mais facilmente alterado (variável manipulada). Por exemplo, em um refrigerador, a variável controlada é a temperatura da câmara refrigeradora e a manipulada é a vazão de fluido refrigerante do equipamento. Essas malhas utilizam de quatro componentes principais: Sensores; Comparador; Controlador; e Elemento Final de controle ou atuador (OGATA, 2010).

Como exemplificado anteriormente, o sensor do refrigerador seria um termopar dentro da câmara e o elemento final é a potência da bomba de fluido refrigerante.

O comparador calcula a diferença entre a variável lida pelo sensor e quanto essa variável deveria ser para manter o sistema controlado (denominado de “set point”), essa diferença se denomina erro (OGATA, 2010). No exemplo do refrigerador, o erro seria calculado entre a temperatura lida pelo termopar e a temperatura desejada para o sistema.

O controlador determina, a partir do erro, a intensidade da resposta que será passada para o elemento final. Existem alguns modelos diferentes para os controladores, entre eles existem os controladores “*Bang-Bang*”, P, PI e PID. Cada um destes possui um funcionamento específico. Por exemplo, o controlador P atribui

uma resposta ao elemento final proporcional ao erro. Já o PID atribui uma resposta que consiste na combinação de uma parcela proporcional, uma integrativa e uma derivativa do erro (OGATA, 2010).

No último controlador citado, o PID, descrito pela Equação 1, sendo os coeficientes K_p ganho proporcional que relaciona o erro de um processo a um fator de controle G(s), sendo s uma variável da equação da transformada de La Place, T_i tempo integrativo é um fator que corrige um erro considerando sua duração e dessa forma o amortecendo, e T_d tempo derivativo é um fator que corrige o erro antes que o mesmo se manifeste. Para que o controlador seja eficiente é necessário que esses parâmetros estejam em sintonia, e, para isso, existem vários métodos para obtenção desses parâmetros.

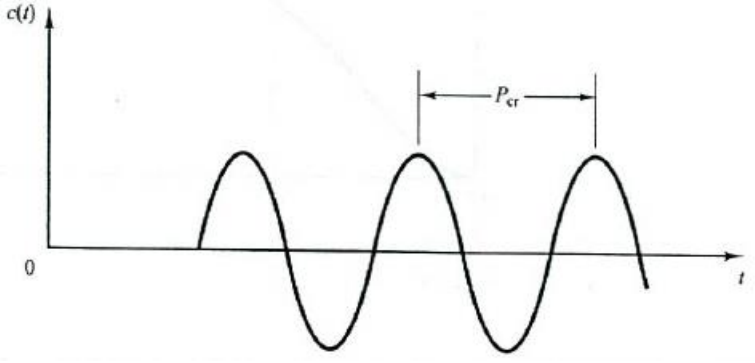
$$G(s) = K_p \left(1 + \frac{1}{T_i \cdot s} + T_d \cdot s\right) \quad (1)$$

3.3 MÉTODO DE SINTONIA DE ZIEGLER-NICHOLS

Um dos métodos mais utilizados para a sintonia de controladores foi desenvolvido por Ziegler-Nichols, que foram responsáveis pelo desenvolvimento de dois métodos de sintonia, um deles em malha aberta ou processo da curva de reação e o outro em malha fechada ou ganho último.

O método aplicado a malha fechada consiste em inicialmente definir T_i como tendendo ao infinito e T_d igual a zero. Após isso, varia-se o termo proporcional K_p até se obter um comportamento semelhante a oscilação, conforme presente na Figura 2. Logo, o valor de K_p que gera esse comportamento é então denominado constante crítica representado pelo símbolo K_{cr}. Por essa mesma oscilação, obtém-se também o período crítico, P_{cr}. Esses valores são essenciais para a aplicação do método de Ziegler-Nichols. Os valores de K_{cr} e P_{cr} são aplicados nas equações do Quadro 1 para se obter os valores sintonizados de K_c, T_i e T_d do controlador (OGATA, 2010).

Figura 2 - Gráfico de oscilação e sintonia de Ziegler-Nichols



Fonte: OGATA (2010)

Quadro 1 - Regras para a sintonia de controladores

Tipo de controlador	K_p	T_i	T_d
P	$0{,}5K_{cr}$	∞	0
PI	$0{,}45K_{cr}$	$0{,}833P_{cr}$	0
PID	$0{,}6K_{cr}$	$0{,}5P_{cr}$	$0{,}5K_{cr}$

Fonte: adaptado de OGATA (2010)

Os valores determinados pela regra de Ziegler-Nichols são utilizados na Equação 1 que é responsável por descrever o controlador PID do Sistema.

3.4 AUTOMATIZAÇÃO DE SISTEMAS E ARDUINO®

O uso de computadores para auxiliar na indústria é algo que começou no final do século passado e vem crescendo exponencialmente. Nos dias atuais, encontra-se diversos sistemas que realizam funções antes desempenhadas por trabalhadores, verificando variáveis e realizando ações por meio de algoritmos. Tais sistemas, conhecidos como CPS ou Sistemas Ciberfísicos (*Cyber-Physical Systems*), fazem uma integração entre processos físicos e computadores, podendo incluir redes de comunicação para acesso remoto. Ajeev (2015) descreve CPS como sistemas de computação comunicando-se entre si e interagindo com o mundo físico por meio de sensores e atuadores. Para Coelho (2016), "é o resultado da evolução tecnológica dos computadores, dos sensores, e das tecnologias de comunicação".

A presença e a atuação de um CPS em uma indústria diminui a necessidade de supervisão, criando uma automatização e permitindo significante diminuição do número de pessoas necessárias para realizar diversas funções. A comunicação de diversos CPSs cria um Sistema de Produção Ciberfísico ou CPPS (*Cyber-Physical Production Systems*). Deloitte (2015a) descreve os CPPSs como uma das bases para a Indústria 4.0, também denominada por alguns como a 4ª Revolução Industrial.

Com tal expansão sem precedentes de sistemas automatizados, surgiram também projetos visando simplificar a criação de sistemas simples com comunicação e/ou controle com computadores. Uma das principais plataformas de desenvolvimento de microcontroladores é o Arduino®, devido a sua facilidade de uso para pessoas sem conhecimentos técnicos e a vasta gama de informações na internet (fórum do Arduino®, outros fóruns, blogs, canais de vídeos online e livros) (MCROBERTS, 2011). Devido à grande popularização, é possível encontrar projetos prontos, necessitando realizar a conexão do microprocessador com os respectivos sensores e atuadores.

3.5 CONEXÃO COM A INTERNET

A internet é uma rede global interconectada de computadores trocando informações em qualquer lugar e a qualquer hora. Seu uso possibilitou comunicação dentro de empresas com maior facilidade, desde setores em um mesmo local até diferentes franquias da mesma empresa. Com a evolução dos computadores, tornou-se possível evoluir essa comunicação para servir ao propósito de controlar áreas de uma indústria tendo informações de outras áreas, proporcionando uma grande automatização (SLEVIN, 2000).

A praticidade da comunicação sem fios vem se tornando útil para conectar objetos do dia a dia com a rede. Essa conexão, de acordo com SANTOS et al. (2016), "viabilizará, primeiro, controlar remotamente os objetos e, segundo, permitir que os próprios objetos sejam acessados como provedores de serviços". Segundo Soundmaeker et al.(2010), conectar objetos à internet significa criar a Internet das Coisas (*IoT, Internet of Things*). A *IoT*, além de diversos outros usos, permite que aplicações domésticas sejam criadas para controlar o ambiente remotamente. Com

esse intuito, faz-se necessária uma plataforma de baixo custo e de fácil controle para o usuário, um dos pilares do Arduino®. Oliveira (2017) apresenta o uso do Arduino® com o microcontrolador ESP, mostrando que é totalmente integrado, possui processamento suficiente para controlar diversas atividades e a um custo de U$ 2,50.

Apesar da praticidade e do baixo custo de criar projetos *IoT* sem muito conhecimento na área, a necessidade de um servidor em muitos casos acaba atrapalhando a conclusão destes. Atualmente, diversas plataformas para o controle e o armazenamento de dados na nuvem estão no mercado para serem utilizadas por meio de um computador ou smartphone, como Blynk e Cayenne. Muitas dessas plataformas apresentam versões gratuitas para o desenvolvimento dos projetos. Dibaba (2018) cita como principais vantagens em se usar a plataforma *Cayenne,* a velocidade e a facilidade de criar projetos atrelada a uma interface focada na parte gráfica, sem a necessidade de conhecimentos prévios na área de computação.

3.6 RESFRIAMENTO

Inicialmente, o maior objetivo para resfriamento é reduzir a temperatura dos alimentos para sua preservação. Com o passar dos tempos, outras utilidades foram encontradas para abaixamento da temperatura, entretanto, a mais relevante é a manutenção térmica de sistemas de corpos e fluidos. Esse sistema pode ser uma cultura de microrganismos sensíveis à temperatura, uma sala de estar ambientada, ou, voltando ao seu objetivo primordial, o interior de uma geladeira.

Métodos distintos podem retirar o calor fazendo com que a temperatura seja afetada. Esses métodos incluem exposição à um corpo com menor temperatura, um gás que sofreu expansão isentálpica, um líquido na iminência da evaporação ou um equipamento que exerce o efeito termoelétrico.

O efeito termoelétrico é a conversão direta de tensão elétrica em diferença de temperatura e vice-versa. Isso significa que se um equipamento termoelétrico for ligado em uma tensão elétrica, uma parte dele irá esquentar e outra irá esfriar, esse efeito também é conhecido como efeito Peltier. Silverio (2012) descreve esse efeito como "a produção de um gradiente de temperatura em uma junção de dois semicondutores de materiais diferentes quando submetidos a uma tensão elétrica

em um circuito fechado". Esse método é interessante pela sua praticidade e por ser mais amigável com o meio ambiente do que as outras formas citadas.

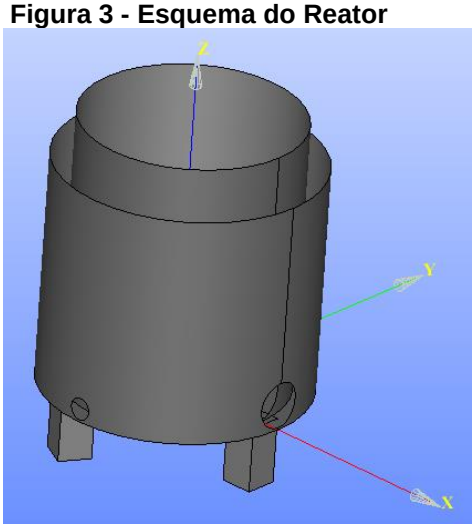
4 METODOLOGIA

Esta seção trata de todas as etapas desde a construção, montagem e programação realizadas durante o desenvolvimento deste trabalho. Cada um dos tópicos será tratado individualmente a seguir.

4.1 CONSTRUÇÃO DO REATOR

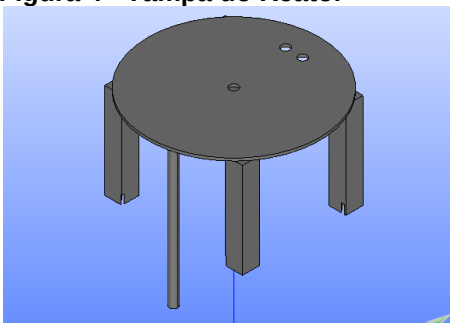
A construção do reator iniciou-se idealizando dois cilindros ocos de aço inox 304, destampados e concêntricos, sendo o interno o próprio reator e o externo, sua jaqueta. O conjunto contou com três pés para suporte, uma saída de esgotamento do reator, uma entrada e uma saída para a jaqueta, entrada para uma resistência elétrica e, uma tampa acoplável no topo do cilindro interno com furos para inserção de um motor de passo. O esquema, criado por meio do *software* Salomé é apresentado nas Figuras 3 e 4 e suas dimensões, no Quadro 2.

Figura 3 - Esquema do Reator



Fonte: Autoria própria

Figura 4 - Tampa do Reator



Fonte: IAMARINO, 2017

Tabela 1 - Dimensões do reator CSTR

Dimensões	Valores
Raio interno	90mm
Raio externo	93mm
Raio da Jaqueta	120mm
Altura do tanque	250mm
Altura da Jaqueta	230mm
Volume total	6,36L
Altura dos pés	80mm
Comprimento dos pés	15mm
Largura dos pés	15mm
Rosca para dreno do reator	½''
Rosca para entrada do fluido refrigerante	½''
Rosca para dreno do fluido refrigerante	½''
Rosca para acoplamento da resistência	1''
Altura da haste de agitação	300mm
Comprimento das pás de agitação	50mm

Fonte: Autoria própria

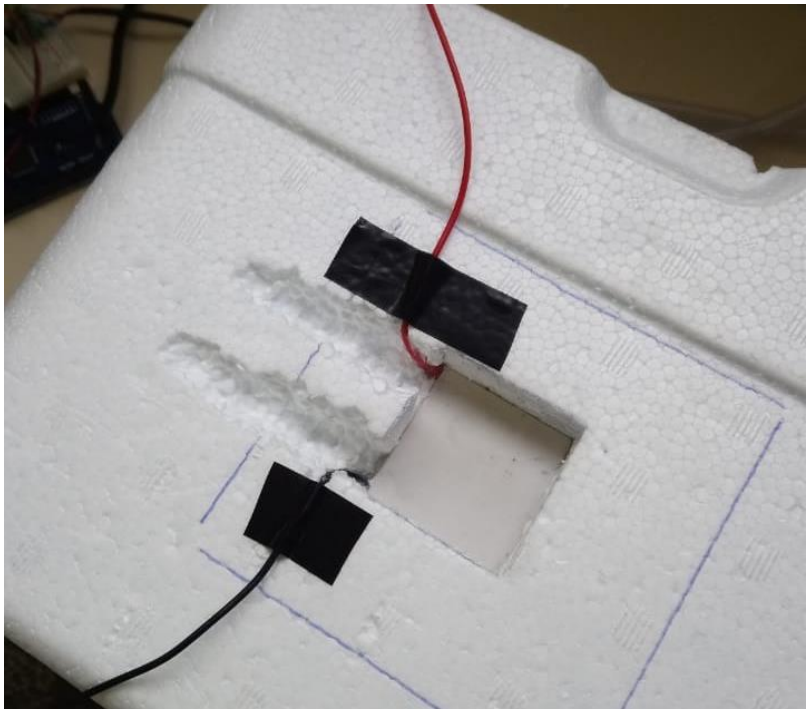
O projeto foi enviado para uma empresa para sua confecção. Adicionou-se ao reator cotovelos, conexões hidráulicas, válvulas, resistência elétrica e saídas para mangueiras. Após a construção do reator estar finalizada, adicionou-se isolamento térmico cobrindo a jaqueta com uma camada de EVA.

4.2 CONSTRUÇÃO DO SISTEMA DE REFRIGERAÇÃO

O sistema é composto por uma caixa térmica de isopor com 8 litros de capacidade, quatro placas de Peltier, quatro dissipadores metálicos 10x10 cm, quatro dissipadores por fluxo de água do tipo *Water Block*, duas bombas hidráulicas, dois radiadores e oito *coolers* de computador. A caixa possui dois furos na sua lateral superior nas dimensões das mangueiras para o bombeamento de líquido refrigerante para a jaqueta e de volta dela. Para realizar o resfriamento, utilizou-se placas de Peltier com dissipadores metálicos no "lado frio", que removerá calor do fluido refrigerante (água) e dissipadores com fluxo de água no "lado quente", que

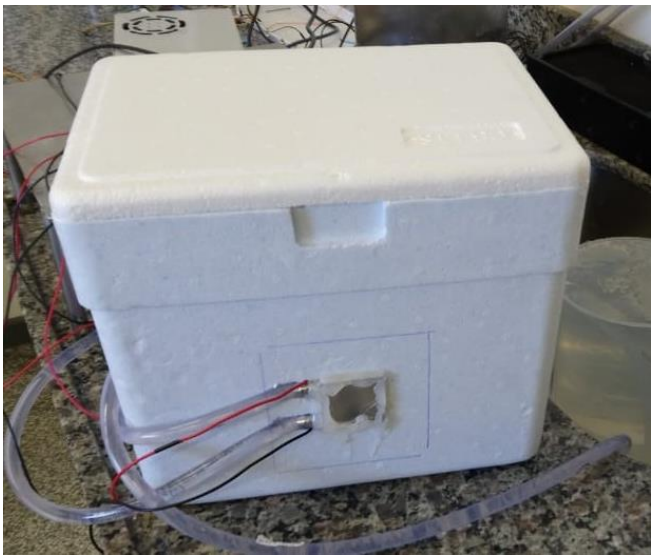
dissipam o calor retirado pela placa. Foram feitos quatro orifícios no isopor da caixa nas dimensões da placa de Peltier (4x4 cm). Dentro dos orifícios, foram colocadas as placas com o “lado frio” virado para dentro da caixa e o “lado quente” para fora. Os dissipadores metálicos foram acoplados na parte interna da caixa de refrigeração, sobre as placas de Peltier e os dissipadores por fluxo foram acoplados na parte externa sobre a placa. Ambos os dissipadores foram acoplados na placa de Peltier utilizando pasta térmica para melhor transferência de energia. Nota-se que como a espessura da parede de isopor é muito maior que a espessura da placa termoelétrica, boa parte do dissipador por fluxo fica dentro do orifício do isopor. Com a caixa térmica finalizada, adicionou-se mangueiras para a passagem de água pelos dissipadores por fluxo. As bombas empurram água de um reservatório para dentro dos dissipadores de fluxo. As Fotografias 1, 2 e 3 demonstram a construção descrita acima.

Fotografia 1 - Placas de Peltier acopladas no isopor



Fonte: Autoria própria

Fotografia 2 - Dissipador de Fluxo acoplados no isopor



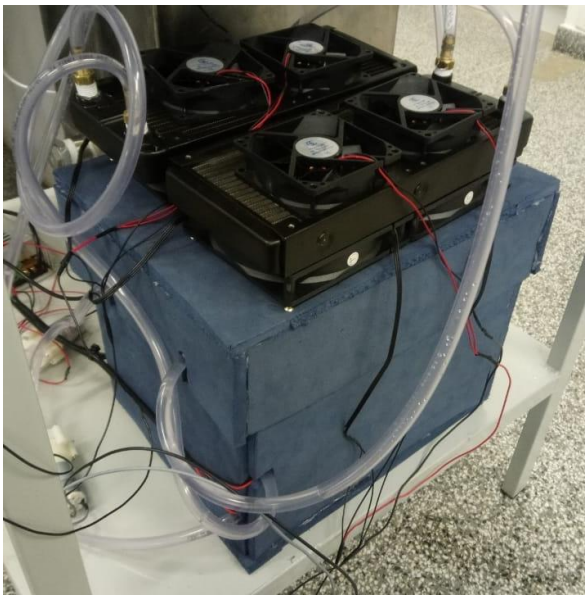
Fonte: Autoria própria

Fotografia 3 - Ligação do sistema de Refrigeração

Fonte: Autoria própria

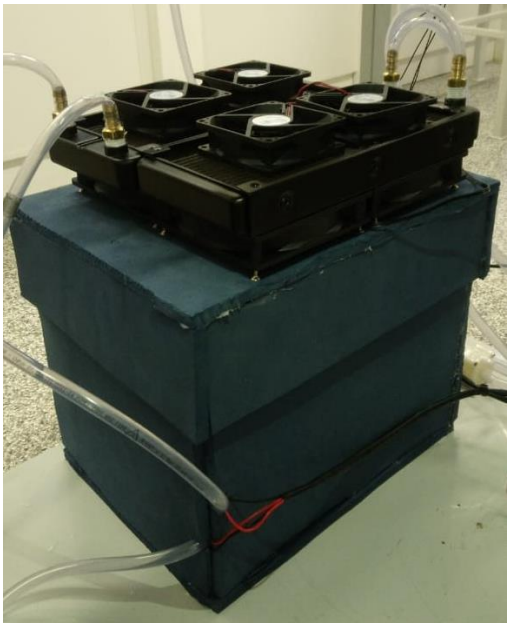
Antes de retornar para o reservatório, o fluxo de água é enviado para um radiador equipado com quatro *coolers* de computador para melhor arrefecimento. Nota-se, que cada bomba alimenta água para dois dissipadores de fluxo ligados em série e, então, para um radiador antes de retornar ao reservatório. É importante denotar que o bombeamento de água deve ser o suficientemente rápido para tirar calor dos dissipadores de fluxo, porém, suficientemente lento para melhor arrefecimento nos radiadores. Finalmente, visando melhorar o isolamento térmico da caixa de isopor, adicionou-se duas camadas de EVA ao seu exterior, tomando cuidado com as saídas das mangueiras citadas acima. As Fotografias 4, 5, 6 e 7 mostram o sistema finalizado.

Fotografia 4 - Sistema de Refrigeração Completo



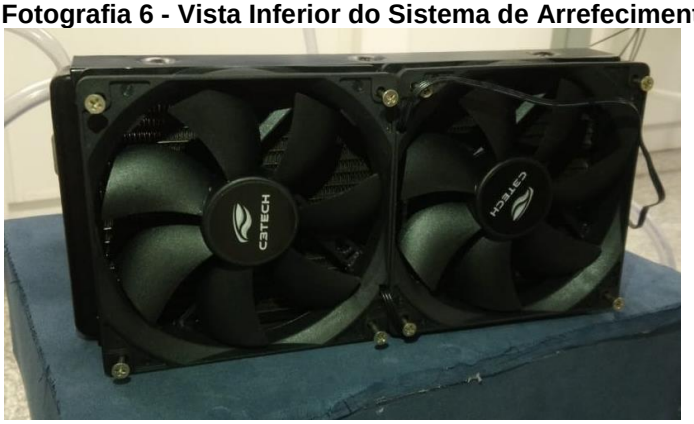
Fonte: Autoria própria

Fotografia 5 - Vista Lateral do Sistema de Arrefecimento



Fonte: Autoria própria

Fotografia 6 - Vista Inferior do Sistema de Arrefecimento



Fonte: Autoria própria

Fotografia 7 - Reservatório de Água para Arrefecimento

Fonte: Autoria própria

4.3 INSTALAÇÃO ELETRÔNICA DE SENSORES E INSTRUMENTOS

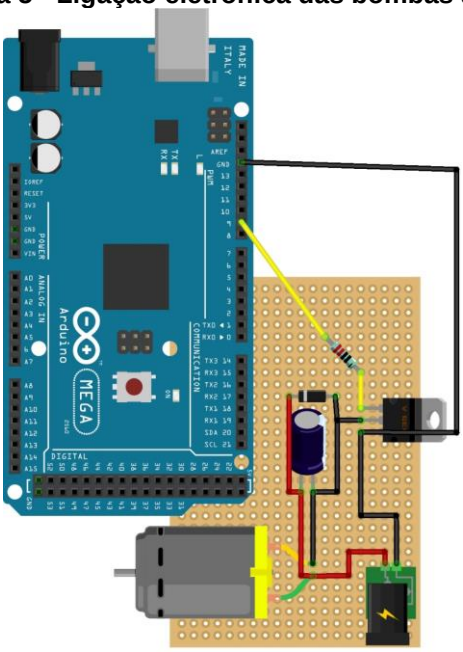
A instalação incluiu a conexão de bombas hidráulicas, motor de passo, placas de Peltier, sensores de temperatura, sensor de pressão e resistência térmica a fontes de energia e/ou ao microprocessador Arduino®. Foram necessárias também duas fontes de alimentação externas de 12V, uma para os equipamentos referentes à caixa de refrigeração e outra ao reator, e uma externa de 220V. Cada um dos equipamentos será explicado individualmente a seguir. Para a confecção dos circuitos eletroeletrônicos, foram necessárias todas as informações de todos os equipamentos elétricos, eletrônicos e instrumentos, as quais podem ser encontradas nas especificações do fabricante (*datasheet*).

A instalação física dos sensores e instrumentos muitas vezes requer componentes eletrônicos adicionais para funcionamento e/ou segurança. Esse trabalho apresentou, em cada uma de suas subsequentes seções, um diagrama apresentando a conexão eletrônica de tais componentes assim como suas especificações no respectivo texto (Ex: Resistência de 1kΩ).

4.3.1 Bomba

As bombas hidráulicas utilizadas, do tipo Mini Bomba de Água RS385, são alimentadas por uma fonte elétrica de 12V, duas delas referentes à caixa de resfriamento foram ligadas em série sem controle enquanto as outras três bombas foram ligadas em paralelo e controladas pelo Arduino®. Dessas três, duas são referentes ao controle de volume do reator (uma para entrada e outra para saída) e uma, à jaqueta (alimentação por bomba e a saída por gravidade). A ligação no Arduino® é demonstrada na Figura 5.

Figura 5 - Ligação eletrônica das bombas ao Arduino®



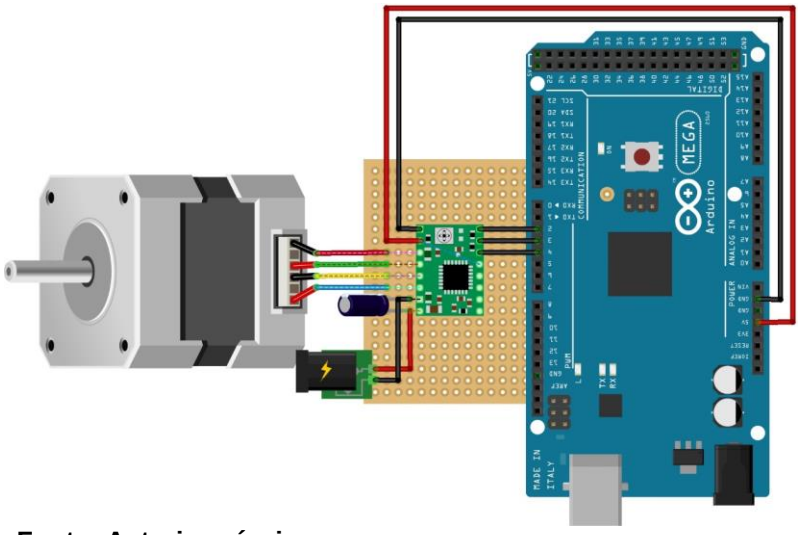
Fonte: Autoria própria

Na Figura 5 são apresentados o Arduino®, uma resistência de 1 kΩ, um diodo, um capacitor de 220 µF, um regulador de tensão de 5V, a bomba e a fonte de 12V.

4.3.2 Motor de Passo

O motor de passo é alimentado por uma fonte de 12V, porém, deve ter seus pinos ligados em uma placa A4988. Essa placa auxilia a controlar o motor por meio do Arduino®. Para tal conexão seja correta, os pinos *DIRECTION*, *STEP* e *SLEEP* devem ser ligados ao Arduino®. A ligação é demonstrada na Figura 6.

Figura 6 - Ligação eletrônica do motor de passo ao Arduino®



Fonte: Autoria própria

Na Figura 6 são apresentados o motor de passo NEMA17, um capacitor de 220 µF, a fonte de 12V, a placa A4988 e o Arduino®.

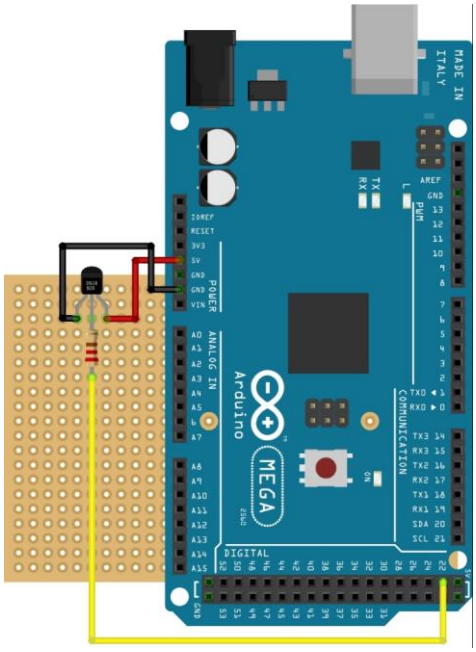
4.3.3 Placas de Peltier

As quatro placas de Peltier foram ligadas em paralelo em uma fonte de 12V, porém, diferentemente da maioria dos outros equipamentos, não existe controle sobre as mesmas, consistindo apenas de dois fios de alimentação, um positivo e um negativo.

4.3.4 Sensor de Temperatura

O sensor de temperatura do tipo J com haste de aço inox é ligado na própria fonte de 5V do Arduino® seguindo o esquema na Figura 7.

Figura 7 - Ligação eletrônica do sensor de temperatura ao Arduino®



Fonte: Autoria própria

Na Figura 7, apresentam-se o termopar, o Arduino® e uma resistência de 1 kΩ.

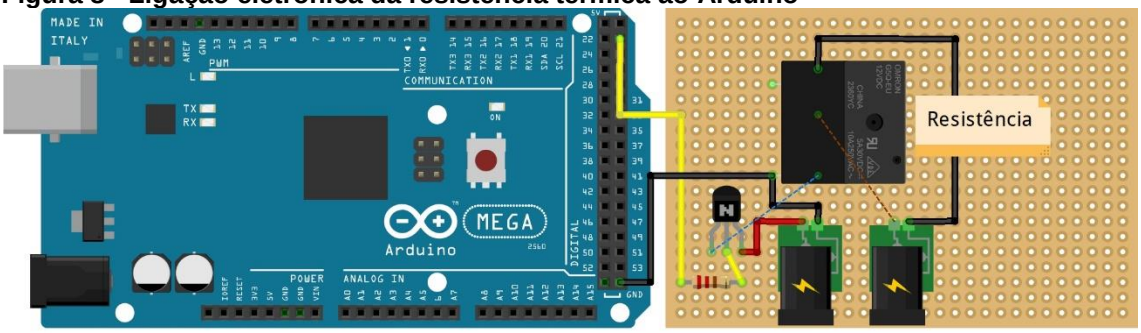
4.3.5 Sensor de Pressão

Diferentemente de outros equipamentos, o sensor de pressão não necessitou nenhum circuito adicional, apenas conectou os pinos da placa corretamente no microprocessador já garantiu que o dado será passado ao Arduino®.

4.3.6 Resistencia Térmica

A resistência não necessita de nenhum circuito intermitente para seu funcionamento, apenas ligá-la na fonte de alimentação de 220V já garante funcionamento. Entretanto, optou-se por implementar um sistema de segurança controlado pelo Arduino®. Dessa forma, a resistência só seria ligada uma vez que o Arduino® mande o comando para tal. O esquema da ligação segue na Figura 8.

Figura 8 - Ligação eletrônica da resistência térmica ao Arduino®



Fonte: Autoria própria

Na Figura 8, apresenta-se o Arduino®, um transponder NPN, um relê de 12V, duas fontes de energia, a da esquerda de 12V e a direita de 220V e uma resistência de 1 kΩ.

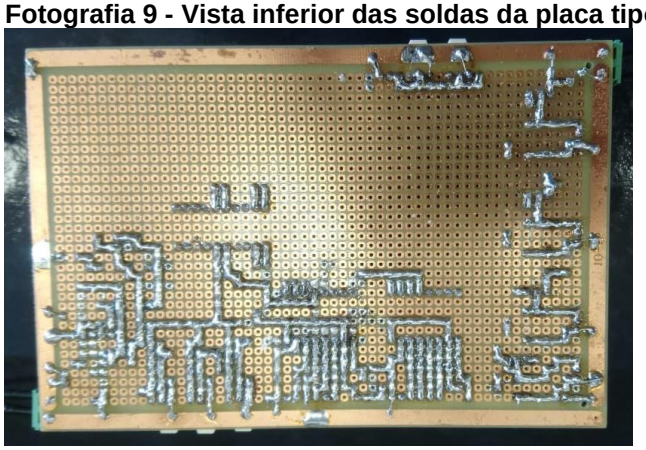
4.3.7 Confecção da Placa Tipo Ilha

Para realizar as conexões para a automatização dos equipamentos do reator, foi utilizado uma placa de solda tipo ilha, na qual cada conjunto de componentes eletrônicos é acoplada à placa com um ferro de solda e liga de estanho tornando o sistema mais visualmente limpo e prático. Com essa, é possível utilizar somente um terminal de alimentação 12V, um de 5V (do próprio Arduino®) e um de 220V, conectando os polos negativos das alimentações de 12V e 5V. Na placa foi implementada quatro terminais de bombas hidráulicas, dois de motor de passo, três de sensores de temperatura e um de resistência térmica. Nota-se que como apenas um terminal de 12V e 5V é implementado, os equipamentos que deverão compartilhar fontes de energia são ligados em paralelo. A Fotografia 8 mostra a parte superior da placa e a Fotografia 9 mostra a parte inferior na qual visualiza-se as soldas feitas na placa.

Fotografia 8 - Vista superior da placa tipo Ilha

Fonte: Autoria própria

Fotografia 9 - Vista inferior das soldas da placa tipo Ilha



Fonte: Autoria própria

4.4 COMUNICAÇÃO COM O MICROPROCESSADOR

Para o controle do sistema, foi necessário a criação de um programa no microprocessador Arduino® Mega 256 para coletar os dados dos sensores e enviar comandos para os equipamentos como bombas, resistência e motor de passo. Durante a confecção do programa, foram utilizadas bibliotecas de algoritmos do próprio Arduino® assim como outras fontes de código aberto (*Open Source*).

O Arduino® apresenta quatro tipos de portas: Digital, Analógica, PWM e Serial (*COMMUNICATION*). Cada uma tem um propósito e funções específicas que auxiliam a manipulação ou recepção de dados dos equipamentos externos ao microprocessador. As bombas e motores de passo utilizam os pinos PWM do Arduino®, os sensores de temperatura e resistência térmica utilizam pinos digitais e o sensor de temperatura usam pinos analógicos. O Arduino® recebe um sinal de cada sensor, na porta apropriada, que é então processado pelo algoritmo carregado ao

microprocessador e gera valores associados a cada sensor, que são apresentados ao usuário. Analogamente, o usuário pode enviar dados para os equipamentos manipulados, o Arduino® converte esse comando por meio do algoritmo em sinais para cada um dos equipamentos ligados às portas corretas.

4.4.1 Programação e Controle

Uma vez que todos os equipamentos estão corretamente ligados aos seus circuitos e ao Arduino® pelas portas corretas, obtém-se os valores dos sensores e enviam-se os comandos aos equipamentos, tudo por meio do algoritmo carregado ao microcontrolador. Alguns desses equipamentos não são controlados pelo usuário, mas sim por algoritmos de controle. No Apêndice A encontra-se o programa completo. Nele pode-se notar o controlador proporcional (Controlador P) automatizando a bomba de saída do reator, a qual regula o nível de líquido no interior dele. Semelhantemente, a bomba de entrada da jaqueta também está sendo controlada por controlador P, a fim de regular a temperatura interna do reator. Um controlador P tem por principal vantagem a simplicidade de programação, dado que depende somente do erro da variável controlada, além de possibilitar futuros estudos a fim de melhorar o controlador para um proporcional integrativo (PI) ou proporcional integro-derivativo (PID).

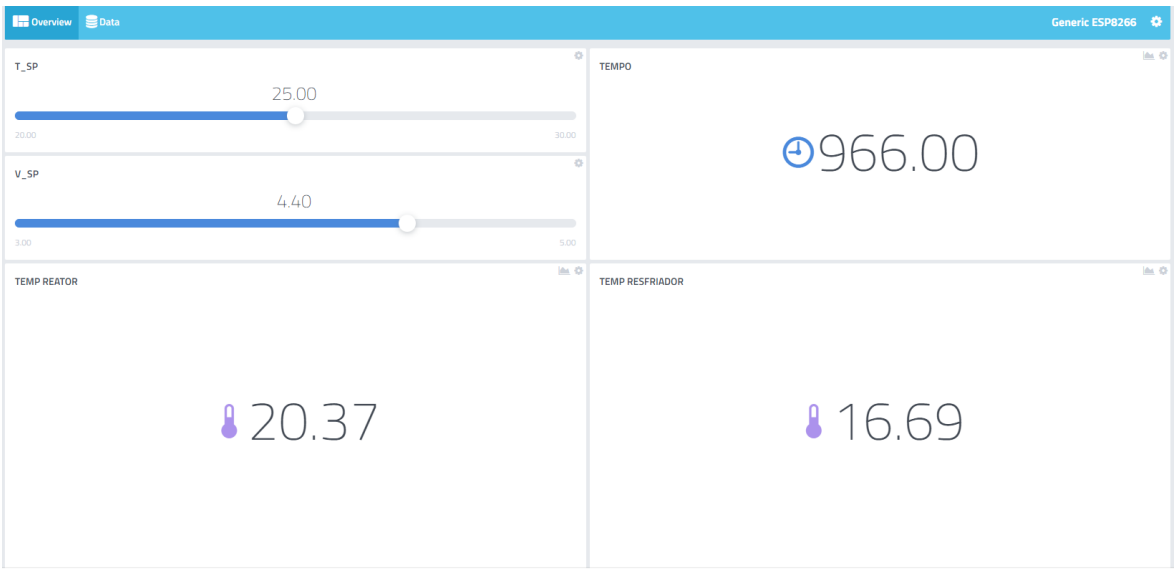
4.4.2 Conexão *online*

Atendendo a um dos escopos do trabalho, a comunicação da placa Arduino® com alguma forma de conexão *online* foi feita a partir de um microprocessador ESP32 DEVKITV1. Essa comunicação foi realizada com o auxílio da plataforma de desenvolvimento *Cayenne*, da empresa *MyDevice.* A conexão com a rede online é feita no *website* criando um projeto, indicando que tipo de placa com conexão online será utilizada (no presente trabalho, placas ESP32) e então recebendo dados de conexão direta com *Cayenne* fornecidos pela própria plataforma. Esses dados são incluídos em um segundo algoritmo que é carregado no microprocessador ESP32, a fim de conectar-lo no servidor da *Cayenne.* Adicionalmente, deve-se informar ao

algoritmo o nome e senha da rede *Wifi* na qual o microprocessador deve se conectar.

No Apêndice B, encontra-se o algoritmo carregado ao microprocessador ESP32. Nele existe a comunicação com o servidor, exportando dados recebidos do Arduino® para a página online e recebendo dados inseridos na página e os enviando para o Arduino®. Essa comunicação de mão dupla com o Arduino® é possível utilizando as portas Seriais presentes em ambos o Arduino® e o ESP32, denominadas como portas TX (Transmissor) e RX (Receptor). É importante denotar que a conexão entre tais portas deve ser cruzada, ou seja, a porta RX do Arduino® deve ser ligada na porta TX do ESP32 e vice-versa. Adicionalmente, como a ESP32 opera numa voltagem de 3,3V e o Arduino®, de 5V, isso impede que ambas sejam ligadas diretamente, portanto requer-se um conversor de tensão. A Figura 9 mostra a tela do aplicativo da *Cayenne* no navegador.

Figura 9 - Tela do aplicativo Cayenne de comunicação remota



Fonte: Autoria Própria

4.5 EXPERIMENTO E COLETA DE DADOS

Inicialmente, utilizou-se um reservatório de água a temperatura ambiente atrelada às bombas do sistema de refrigeração, reduzindo a temperatura da água dentro da caixa de isopor para aproximadamente 6°C, levando em torno de 4 horas. Em seguida, outro reservatório aquece água até a temperatura de 30°C por meio da resistência térmica. Uma vez que a caixa de isopor apresenta a água nas especificações citadas, a jaqueta do reator é abastecida com água à temperatura

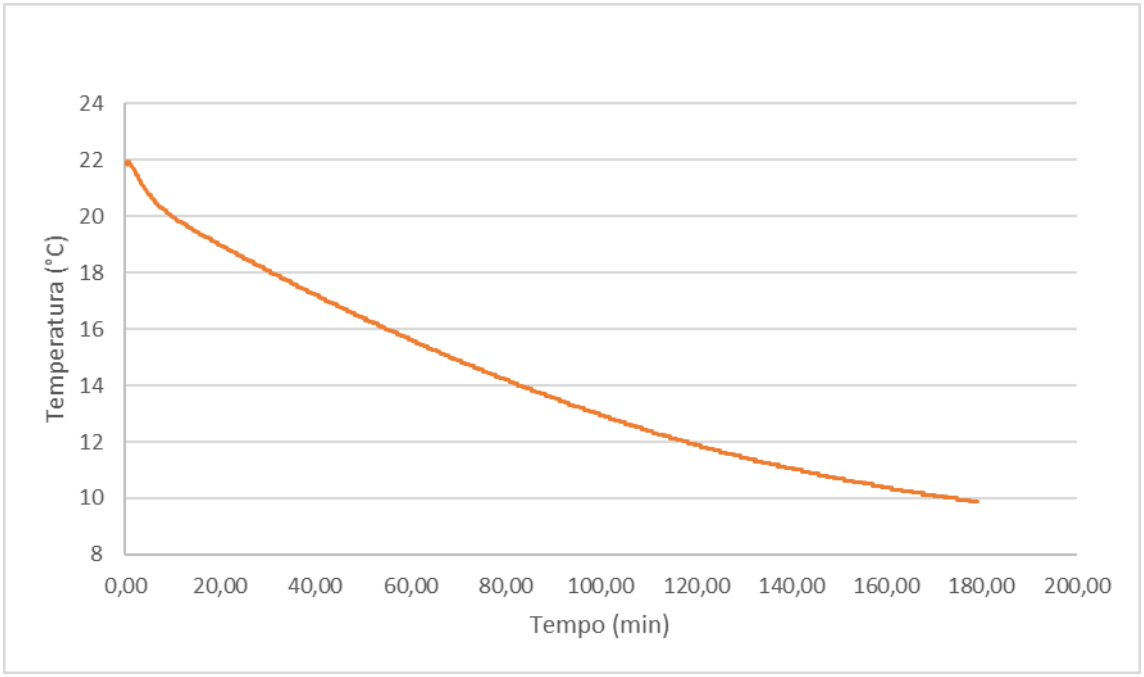
ambiente. A bomba de alimentação na jaqueta é ligada na potência máxima, homogeneizando os dois fluidos até uma temperatura de, aproximadamente 11°C. O experimento é iniciado estipulando uma potência em porcentagem para a bomba de entrada no reator e ativando os controladores P de nível e temperatura do reator.

Os dados coletados foram: Tempo (segundos), Rotação do Motor de Passo (rotações por minuto), Temperatura do Reator (°C), Temperatura do Reservatório Frio (°C), Potência da Bomba de Entrada (%), Potência da Bomba de Saída (%), Potência da Bomba da Jaqueta (%) e Volume do Reator.

5 RESULTADOS E DISCUSSÃO

Inicialmente, testes foram feitos para averiguar o desempenho do sistema de refrigeração, que demonstraram a eficiência das placas de Peltier, chegando a resfriar 6 Litros de água até 10°C sem o uso de *coolers* acoplados nos radiadores. O Gráfico 1 mostra um desses testes.

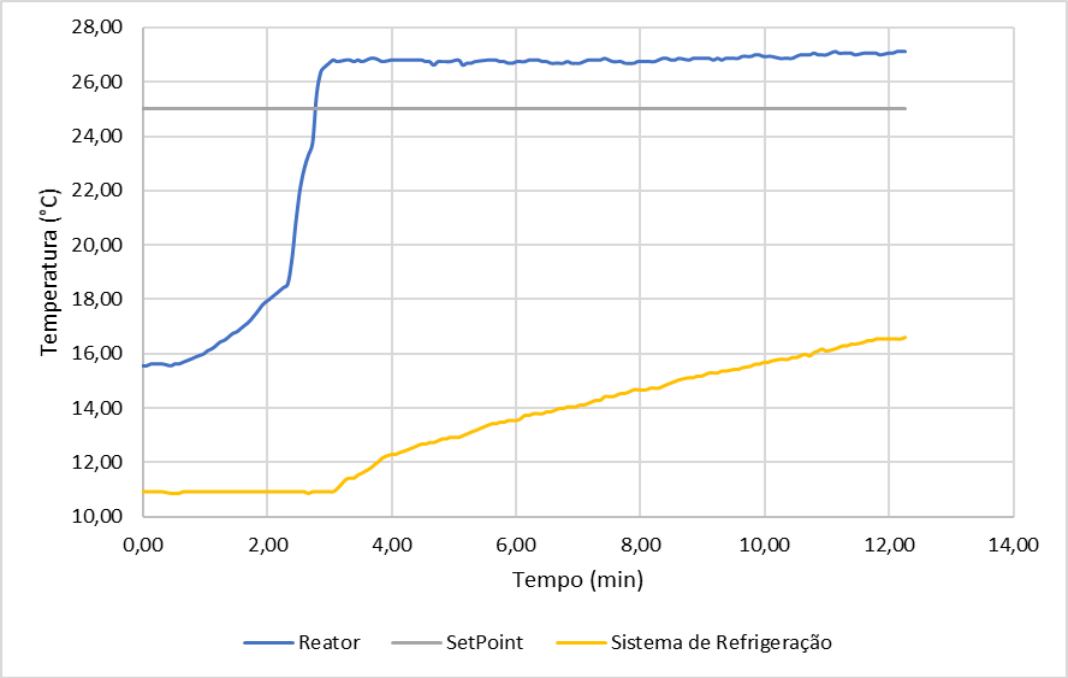
Gráfico 1 - Variação de temperatura dentro do sistema de refrigeração



Fonte: Autoria própria

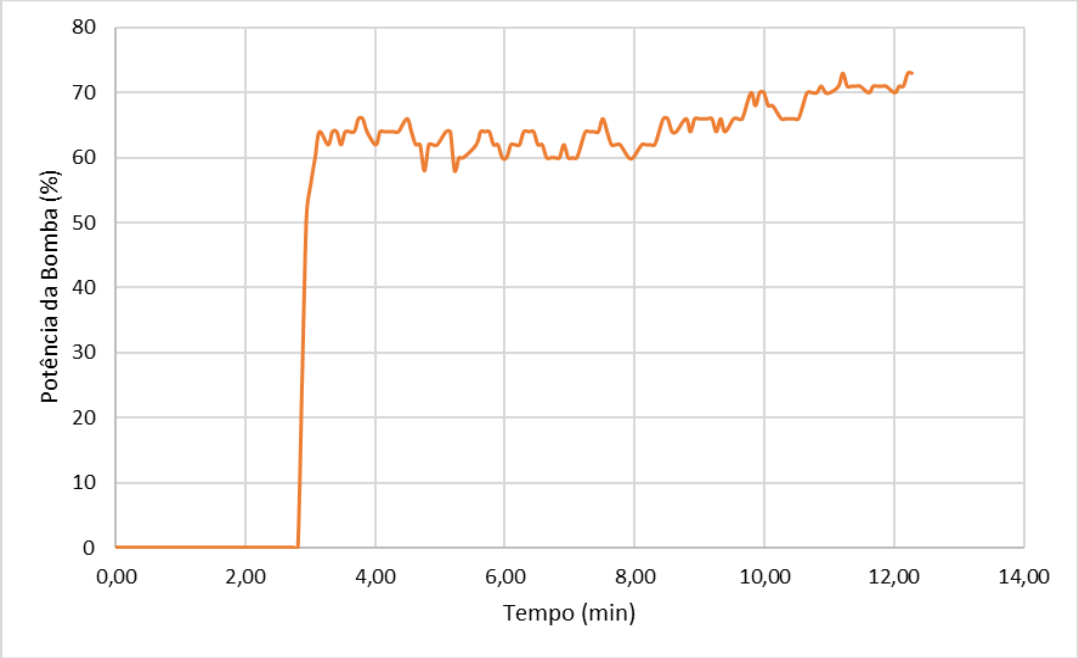
Após a confirmação da efetividade do sistema de refrigeração, fez-se um teste completo, melhorando o sistema de arrefecimento com os *coolers* mencionados. O Gráfico 2 apresenta o controlador P de temperatura atuando sobre o reator e o Gráfico 3 apresenta como o controlador P atua por meio da bomba da jaqueta.

Gráfico 2 - Temperaturas do reator, do sistema de refrigeração e set point durante teste



Fonte: Autoria própria

Gráfico 3 - Potência da bomba da jaqueta, em porcentagem, sob atuação do controlador



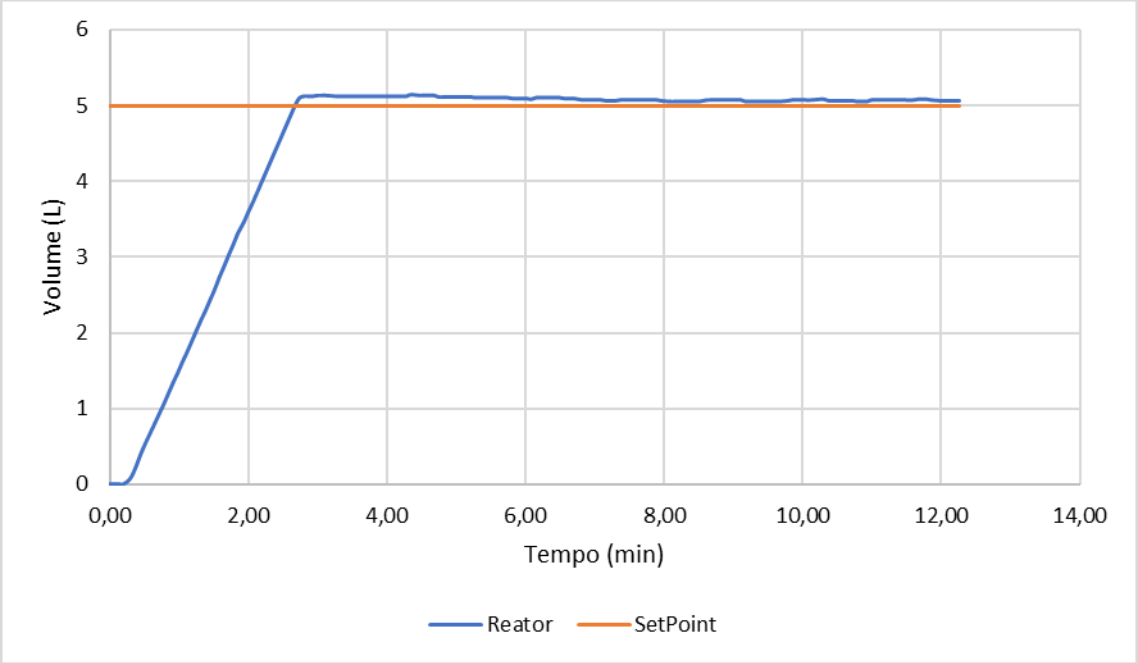
Fonte: Autoria própria

Como o procedimento proposto inicia o reator vazio, o sensor de temperatura fica em contato com o fundo do reator. Este, por sua vez, devido à proximidade com a jaqueta, a cerca de 11°C, faz uma leitura inicial muito abaixo da temperatura ambiente. À medida que o fluido quente a 30°C entra no reator, a leitura do sensor gradualmente sobe até a temperatura real do meio, propriedade característica do

sensor utilizado. Observa-se que, aproximadamente em 3 minutos, o controlador P da temperatura começa a responder ativando a bomba da jaqueta quando a temperatura do reator ultrapassa da temperatura estipulada como *Set Point.* Observa-se também, que a temperatura da caixa de isopor também aumenta à medida que o controlador atua. O acréscimo de temperatura no reservatório implica que o calor retirado do reator por meio da vazão de fluido frio na jaqueta é maior do que o calor retirado pelas quatro placas de Peltier.

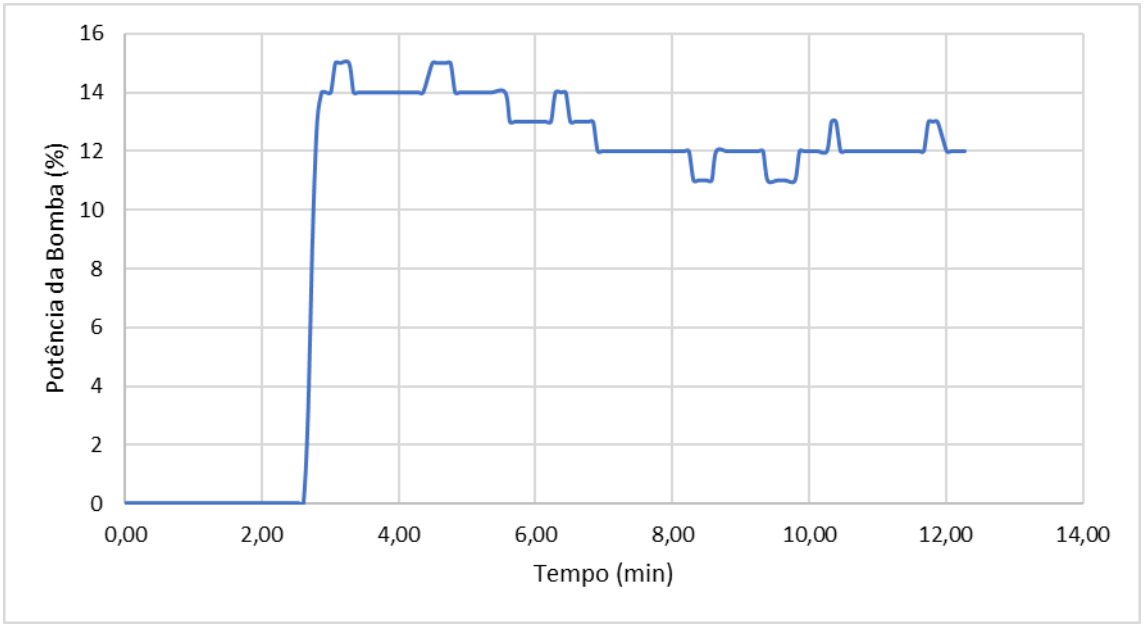
O Gráfico 4 mostra o volume do reator afetado pelo controlador P de nível em um dos testes e o Gráfico 5 mostra a potência da bomba de saída nesse mesmo teste. Uma análise de ambos evidencia a atuação do controlador de volume e demonstra sua ação condizente com o esperado para a potencia da bomba de saída.

Gráfico 4 - Volume do reator sob atuação do controlador



Fonte: Autoria própria

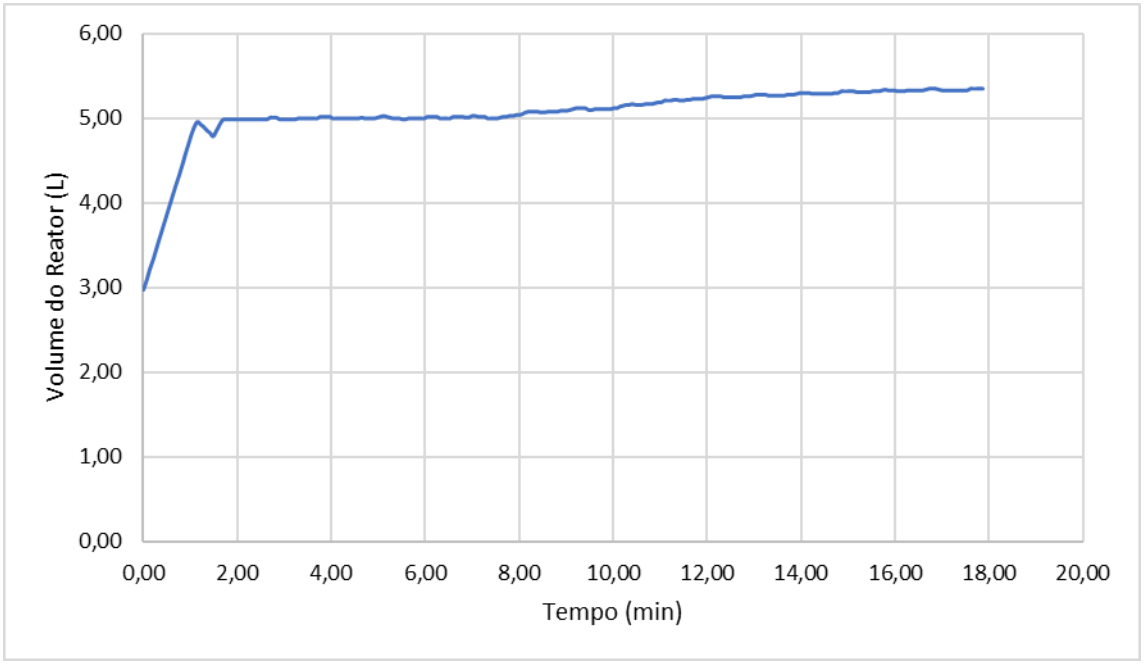
Gráfico 5 - Potência da bomba de saída do reator, em porcentagem, sob atuação do controlador



Fonte: Autoria própria

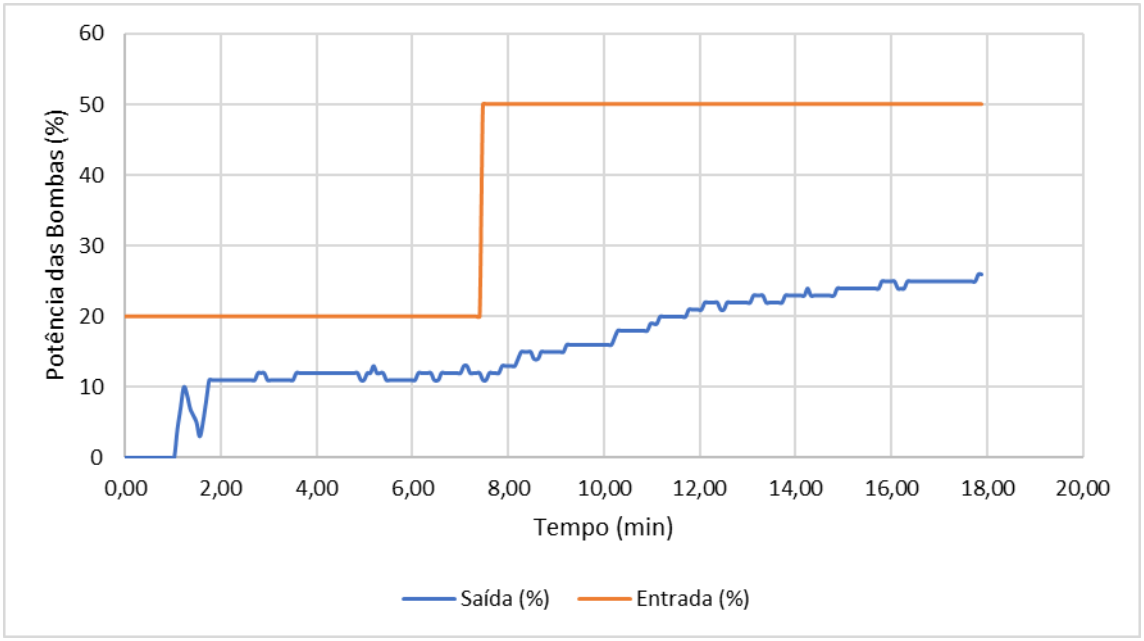
Semelhantemente ao controlador de temperatura, ao redor dos três minutos de teste o controlador começa a responder aumentando a vazão de saída do reator. Em um teste adicional apresentado no Gráfico 6 e 7, pode-se observar a atuação do controlador de nível no volume e na bomba de saída do reator uma vez que a potência da bomba de entrada é aumentada. Essa resposta demonstra a efetividade do controlador P, mesmo com a limitação de um sistema de controle simples.

Gráfico 6 - Resposta do controlador sobre o volume



Fonte: Autoria própria

Gráfico 7 - Resposta do controlador sobre a potência da bomba de saída



Fonte: Autoria própria

6 CONCLUSÃO

Como os resultados indicaram, foi possível implementar um controle de nível muito eficiente apresentando valor final de volume muito próximo do volume de *Set Point,* característica não comum a controladores do tipo P. Também foi possível observar a implementação de um controle de temperatura aceitável, porém muito inferior se comparado com o controle de nível. Isso provavelmente é atribuído à uma troca térmica ocorrida na jaqueta do reator muito superior ao potencial de refrigeração do pequeno de placas de Peltier. Adicionalmente, outra possível razão pelo desempenho inferior do controle de temperatura é o grande reservatório de água utilizada nos dissipadores de fluxo esquentar consideravelmente durante a refrigeração. Apesar de todos os empecilhos atribuídos ao sistema de refrigeração, o método de refrigeração por placas de Peltier mostrou-se como uma alternativa viável, fugindo dos clássicos sistemas com fluidos refrigerantes muitas vezes poluentes.

Apesar do sistema eletrônico funcionar apropriadamente com os comandos inseridos pelo usuário, alguns problemas foram flagrados durante os experimentos. Primeiramente, a relação entre tensão enviada para as bombas hidráulicas e suas vazões, apesar de perfeitamente lineares, não são nem de perto proporcionais. Isso quer dizer que, por exemplo, na tensão máxima, a bomba exerce uma vazão de 1,25 L/min, com 50% da tensão máxima, exerce 1 L/min e sem tensão alguma, ela também não liga. Apesar de ser conhecido que quanto maior a diferença de altura, menor é a eficiência da bomba, os equipamentos utilizados apresentavam perdas muito consideráveis. Notou-se também que quando dois ou mais sensores de temperatura são ligados simultaneamente, a leitura destes é ocasionalmente afetada. Por fim, a plataforma de desenvolvimento *Cayenne,* na sua versão gratuita, não se mostrou adequada para um envio de dados contínuos de dados em tempo real, o período de envio dos dados era de 15 segundos.

6.1 TRABALHOS FUTUROS

Essa seção apresenta sugestões de melhorias que podem ser implementadas visando tornar o sistema mais robusto. As principais sugestões são a

melhoria do sistema de refrigeração, a implementação de controladores PID para nível e temperatura e desenvolvimento de um painel de controle juntamente com uma interface de controle de usuário. Também é possível realizar estudos de reações químicas.

O sistema de refrigeração construído provou a possibilidade do uso de placas de Peltier para controle de sistemas, porém para atender as necessidades de estudos futuros faz-se necessário um aumento de sua capacidade de refrigeração. Para tanto, pode-se aumentar o número de placas de Peltier, o volume de seu reservatório de fluido frio e também de água para arrefecimento.

Para melhor funcionamento e controle do sistema, recomenda-se a instalação de um medidor de pH e a implementação de controladores mais eficientes, como por exemplo o PID, para a manutenção de nível e temperatura. Pensando no usuário final, é recomendável implementar também um painel de controle visando a operação manual do sistema e também o desenvolvimento de interface para simplificar a comunicação e a leitura dos dados coletados.

Por fim, as melhorias acima apresentas, quando concluídas, tornam o sistema ideal para a realização de estudos cinéticos, como determinação do tempo residência em relação a vazão de entrada no reator e de parâmetros cinéticos de reações específicas.

REFERÊNCIAS

AJEEV, A. **Principles of Cyber-Physical Systems**. Cambridge, Massachusetts: Mit Press, 2015. 459 p.

COELHO, P. **Rumo à Indústria 4.0**. Dissertação de Mestrado –FCTU Universidade de Coimbra, Portugal, 2016.

DELOITTE. **Industry 4.0: challenges and solutions for the digital transformation and use of exponential technologies**. Zurique, Suiça, 2015a.

DIBABA, H. **IoT Implementation with Cayenne Platform**. Metropolia University of Applied Sciences. 2018.

FÁBREGA, F. M. **Cálculo de Reatores I.** JundiaÍ: Faculdade Pitágoras de Jundiaí, 2012. Notas de Aula. Disponível em: < http://bizuando.com/material-apoio/reat-quim/reat-quim-apostila.pdf >. Acesso em: 19 ago. 2019.

FOGLER, H. S. **Elementos de Engenharia das Reações Químicas**. 4.ed. Rio de Janeiro: Editora LTC, 2009.853 p.

IAMARINO, J. F. L. **Obtenção dos parâmetros cinéticos e parâmetros de Arrhenius em reações de fase líquida através da construção de um reator batelada**. 2017. 47 p. Trabalho de Conclusão de Curso (Graduação em Engenharia Química) - Universidade Tecnológica Federal do Paraná, Ponta Grossa, 2017.

MCROBERTS, M. **Arduíno Básico**. Novatec Editora. São Paulo: Novatec, 2011.

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

OLIVEIRA, S. **Internet das Coisas com ESP 8266, Arduino e Raspeberry PI**. 1.ed. São Paulo: Novatec, 2017.

SANTOS, B. P. et al. **Internet das Coisas: da Teoria à Prática**. Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuıdos., Belo Horizonte, 2016.

SILVERIO, L. B., **Análise de um condicionador de ar automotivo utilizando o efeito termoelétrico**. Dissertação de mestrado da Universidade de Taubaté –SP. 2012.

SLEVIN, J. **The Internet and Society**. Polity Press, Cambridge, U.K. 2000

SOUNDMAEKER, H et al. **Vision and challenges for realising the internet of things**. European Comission, v. 20. 2010.

APÊNDICE A - Algoritmo utilizado no Arduino® para controle e comunicação com a placa ESP32

```
#include <DallasTemperature.h> // ler a biblioteca
#include <OneWire.h> // ler a biblioteca
#include <HX711.h>

#define PRESSAO_DOUT_PIN  A0
#define PRESSAO_SCK_PIN  A1
#define NIVEL A2
#define SENSOR_TEMPERATURA 22
#define RESISTENCIA 23
#define DIRECTION 2
#define STEP 3
#define SLEEP 4
#define DIRECTION2 6
#define STEP2 7
#define SLEEP2 8
#define BOMBA_R1 9
#define BOMBA_R2 11
#define BOMBA_R3 12
#define BOMBA_J 13

OneWire oneWire(SENSOR_TEMPERATURA);
DallasTemperature sensors(&oneWire);
HX711 sensor_pressao;

int motorSpeed = 100, PWM_BOMBA_R1 = 0, PWM_BOMBA_R2 = 0,
PWM_BOMBA_R3 = 0, PWM_BOMBA_J = 0, RESIS = 0,
VELOCIDADE_AGITADOR = 0, VELOCIDADE_AGITADOR_ANTERIOR= 0;
int VELOCIDADE_AGITADOR2 = 0, VELOCIDADE_AGITADOR_ANTERIOR2 =
100, a;
int PID =0;
float TEMPERATURA_R1 = 0, TEMPERATURA_R2 = 0, TEMPERATURA_J = 0,
PERIODO_INTERVALO_AGITADOR = 0, PERIODO_DESLIGADO = 0,
PERIODO_AGITADOR, VOLUME = 0, Vi;
float PERIODO_INTERVALO_AGITADOR2 = 0, PERIODO_DESLIGADO2 = 0,
PERIODO_AGITADOR2, T_SP = 30;
long PERIODO_DESLIGADO_TEMPO = 0;
long PERIODO_DESLIGADO_TEMPO2 = 0;

float K2 = 40, PWM_ERROZERO2 = 12, VOLUME_SP = 4, ERRO2 = 0, TIV =
500000;
float KJ = 30, PWM_ERROZEROJ = 60, ERROJ = 0;

boolean ATIVA_AGITACAO = 1;
```

```
boolean ATIVA_AGITACAO2 = 0;
boolean ATIVA_TEMPERATURA = 1;
boolean ATIVA_RESISTENCIA = 1;
boolean ATIVA_BOMBA_R1 = 1;
boolean ATIVA_BOMBA_R2 = 1;
boolean ATIVA_BOMBA_R3 = 1;
boolean ATIVA_BOMBA_J = 1;
boolean ATIVA_NIVEL = 1;
int TESTE = 1;
boolean PERIODO= false;
boolean PERIODO2= false;
int i=0;
boolean LEITURA_NIVEL = 0;
const long PRESSAO_OFFSET = 50682624;
const long PRESSAO_DIVIDER = 5895655;

void setup() {
  Serial.begin(9600);
  Serial1.begin(9600);
   pinMode(BOMBA_R1, OUTPUT);
   digitalWrite(BOMBA_R1, LOW);
   pinMode(BOMBA_R2, OUTPUT);
   digitalWrite(BOMBA_R2, LOW);
   pinMode(BOMBA_R3, OUTPUT);
   digitalWrite(BOMBA_R3, LOW);
   pinMode(BOMBA_J, OUTPUT);
   digitalWrite(BOMBA_J, LOW);
   pinMode(STEP, OUTPUT);
   digitalWrite(STEP, HIGH);
   pinMode(DIRECTION, OUTPUT);
   digitalWrite(DIRECTION, LOW);
   pinMode(SLEEP, OUTPUT);
   digitalWrite(SLEEP, LOW);
   pinMode(STEP2, OUTPUT);
   digitalWrite(STEP2, HIGH);
   pinMode(DIRECTION2, OUTPUT);
   digitalWrite(DIRECTION2, LOW);
   pinMode(SLEEP2, OUTPUT);
   digitalWrite(SLEEP2, LOW);
   pinMode(RESISTENCIA, OUTPUT);
   digitalWrite(RESISTENCIA, LOW);
```

```
  pinMode(NIVEL, INPUT);

  sensors.begin();
  sensor_pressao.begin(PRESSAO_DOUT_PIN, PRESSAO_SCK_PIN);
  sensor_pressao.set_scale(PRESSAO_DIVIDER);
  sensor_pressao.set_offset(PRESSAO_OFFSET);
  sensor_pressao.tare();
}

void loop() {
 LE_DADOS();
 ENVIA_DADOS();
 if(ATIVA_NIVEL){
   LEITURA_NIVEL = (analogRead(NIVEL) < 100) ? 1 : 0;
 }
if(TESTE==1){
 EXIBE_DADOS2();
}else{
 EXIBE_DADOS1();
}
 delay(2000);
}

void LE_DADOS() {
   if(Serial.available() > 0){
   // Lê primeiro byte digitado pelo usuário e atua no sistema
        switch (Serial.read()){
             case 'A':
               delay(2);
               switch(Serial.read()){
                 case '1':
                    VELOCIDADE_AGITADOR = Serial.parseInt(); //faixa recomendada
limite 156
                    if((Serial.available() > 0) && (Serial.read() == 'P'))
                       PERIODO_INTERVALO_AGITADOR = Serial.parseFloat();
                    if((Serial.available() > 0) && (Serial.read() == 'D'))
                       PERIODO_DESLIGADO = Serial.parseFloat();
                    if(VELOCIDADE_AGITADOR > 130) //156
                       VELOCIDADE_AGITADOR = 130;
                    else if(VELOCIDADE_AGITADOR < 30 &&
VELOCIDADE_AGITADOR != 0)
                    {
```

```
                    VELOCIDADE_AGITADOR = 30;
                    Serial.println("Velocidade mínima: 30rpm");
                   }
                   break;
                  case '2':
                   VELOCIDADE_AGITADOR2 = Serial.parseInt(); //faixa
recomendada limite 156
                   if((Serial.available() > 0) && (Serial.read() == 'P'))
                      PERIODO_INTERVALO_AGITADOR2 = Serial.parseFloat();
                   if((Serial.available() > 0) && (Serial.read() == 'D'))
                      PERIODO_DESLIGADO2 = Serial.parseFloat();
                   if(VELOCIDADE_AGITADOR2 > 130) //156
                      VELOCIDADE_AGITADOR2 = 130;
                   else if(VELOCIDADE_AGITADOR2 < 30 &&
VELOCIDADE_AGITADOR2 != 0)
                   {
                    VELOCIDADE_AGITADOR2 = 30;
                    Serial.println("Velocidade mínima: 30rpm");
                   }
                   break;
                }
        case 'B':
                delay(2);
                switch(Serial.read()){
                 case '1':
                    PWM_BOMBA_R1 = Serial.parseInt();
                 break;
                 case '2':
                    PWM_BOMBA_R2 = Serial.parseInt();
                 break;
                 case '3':
                    PWM_BOMBA_R3 = Serial.parseInt();
                 break;
                 case 'J':
                    PWM_BOMBA_J = Serial.parseInt();
                 break;
                }
                break;
         case 'V':
                Vi = Vi + VOLUME;
                Serial.println("Volume Zerado");
                break;
```

```
      case 'T':
            TESTE = Serial.parseInt();
            if(TESTE != 0){
              TESTE = 1;
            }else{
              TESTE = 0;
            }
            break;
      case 'P':
            if(PID == 0){
              PID = 1;
            }else{
              PID = 0;
            }
            Serial.println("PID");
            break;
       case 'Z':
            a = Serial.parseInt();
            break;
   }
}
if(Serial1.available() > 0){
// Lê primeiro byte digitado pelo usuário e atua no sistema
    switch (Serial1.read()){
      case 'T':
            T_SP = Serial1.parseInt();
            break;
      case 'V':
            VOLUME_SP = Serial1.parseFloat();
            break;
   }
}
if(ATIVA_BOMBA_R1) // Aplica as alterações
  LIGA_BOMBA_R1();

if(ATIVA_BOMBA_R2)
  LIGA_BOMBA_R2();

if(ATIVA_BOMBA_R3)
  LIGA_BOMBA_R3();
```

```
    if(ATIVA_BOMBA_J)
      LIGA_BOMBA_J();

    if(ATIVA_TEMPERATURA){
        LE_TEMPERATURA();
    }
    if(ATIVA_RESISTENCIA)
    LIGA_RESISTENCIA();

    if(ATIVA_AGITACAO)
    LIGA_AGITACAO();

    if(ATIVA_AGITACAO2)
    LIGA_AGITACAO2();

    if(ATIVA_NIVEL)
    LE_NIVEL();

    delay(100);
}

void ENVIA_DADOS(){
  //Serial.print("A");
  //Serial.println(a);
  Serial1.print(TEMPERATURA_R1);
  Serial1.println(TEMPERATURA_J);
  Serial1.print("A");
  Serial1.println(TEMPERATURA_R1, 2);
  delay(2);
  Serial1.print("B");
  Serial1.println(TEMPERATURA_J, 2);
}

void LE_NIVEL()
{
    if(sensor_pressao.is_ready())
    VOLUME = sensor_pressao.read_average(15)*0.0050086/1000. - 1.2495 +8.21 -
Vi;
}

void LE_TEMPERATURA(){
```

```
        sensors.requestTemperatures();
        TEMPERATURA_R1 = sensors.getTempCByIndex(0);
        TEMPERATURA_R2 = sensors.getTempCByIndex(1);
        TEMPERATURA_J = sensors.getTempCByIndex(2);
}

void LIGA_AGITACAO()
{
  int step = 30;
  // Implementa uma variação suave de velocidade
  if(abs(VELOCIDADE_AGITADOR - VELOCIDADE_AGITADOR_ANTERIOR) <=
step)
  {
     VELOCIDADE_AGITADOR_ANTERIOR = VELOCIDADE_AGITADOR;
  }
  else if((VELOCIDADE_AGITADOR - VELOCIDADE_AGITADOR_ANTERIOR) >
step)
     VELOCIDADE_AGITADOR_ANTERIOR+= step;

  else if((VELOCIDADE_AGITADOR - VELOCIDADE_AGITADOR_ANTERIOR) <
step)
     VELOCIDADE_AGITADOR_ANTERIOR-= step;

  define_periodo();
  if((VELOCIDADE_AGITADOR == 0) || not(PERIODO))
  {
     noTone(STEP);
     digitalWrite(SLEEP, LOW);
  }
  else if((VELOCIDADE_AGITADOR != 0) && PERIODO)
  {
     digitalWrite(SLEEP, HIGH);
     tone(STEP, VELOCIDADE_AGITADOR_ANTERIOR*200/60); // Manda uma
onda quadrada para o driver do motor com a velocidade em rpm desejada.
  }
}

void define_periodo()
{
    if(not(PERIODO))  // Permite usar o acionamento suave no motor de passo
usando os períodos de intervalo.
     VELOCIDADE_AGITADOR_ANTERIOR = 0;
```

```
   if(PERIODO_INTERVALO_AGITADOR != 0)
   {
     if((millis() - PERIODO_AGITADOR)/1000 >=
(PERIODO_INTERVALO_AGITADOR+PERIODO_DESLIGADO))
     {
       PERIODO = false;
       PERIODO_DESLIGADO_TEMPO = millis();
       PERIODO_AGITADOR = millis();
     }
     else if((millis() - PERIODO_DESLIGADO_TEMPO)/1000 >=
PERIODO_DESLIGADO)
     {
          PERIODO = true;
     }
    }
   else
    PERIODO = true;
}

void LIGA_AGITACAO2()
{
  int step = 30;
  // Implementa uma variação suave de velocidade
  if(abs(VELOCIDADE_AGITADOR2 - VELOCIDADE_AGITADOR_ANTERIOR2) <=
step)
  {
     VELOCIDADE_AGITADOR_ANTERIOR2 = VELOCIDADE_AGITADOR2;
  }
  else if((VELOCIDADE_AGITADOR2 - VELOCIDADE_AGITADOR_ANTERIOR2) >
step)
     VELOCIDADE_AGITADOR_ANTERIOR2+= step;

  else if((VELOCIDADE_AGITADOR2 - VELOCIDADE_AGITADOR_ANTERIOR2) <
step)
     VELOCIDADE_AGITADOR_ANTERIOR2-= step;

  define_periodo2();

  if((VELOCIDADE_AGITADOR2 == 0) || not(PERIODO2))
  {
     noTone(STEP2);
```

```
    digitalWrite(SLEEP2, LOW);
  }
  else if((VELOCIDADE_AGITADOR2 != 0) && PERIODO2)
  {
    digitalWrite(SLEEP2, HIGH);
    tone(STEP2, VELOCIDADE_AGITADOR_ANTERIOR2*200/60); // Manda uma
onda quadrada para o driver do motor com a velocidade em rpm desejada.
  }
}
void define_periodo2()
{
   if(not(PERIODO2))  // Permite usar o acionamento suave no motor de passo
usando os períodos de intervalo.
    VELOCIDADE_AGITADOR_ANTERIOR2 = 0;

   if(PERIODO_INTERVALO_AGITADOR2 != 0)
   {
    if((millis() - PERIODO_AGITADOR2)/1000 >=
(PERIODO_INTERVALO_AGITADOR2+PERIODO_DESLIGADO2))
    {
      PERIODO2 = false;
      PERIODO_DESLIGADO_TEMPO2 = millis();
      PERIODO_AGITADOR2 = millis();
    }
    else if((millis() - PERIODO_DESLIGADO_TEMPO2)/1000 >=
PERIODO_DESLIGADO2)
    {
        PERIODO2 = true;
    }
   }
   else
    PERIODO2 = true;
}

void LIGA_BOMBA_R1(){
      analogWrite(BOMBA_R1, map(PWM_BOMBA_R1,0,100,0,253));
}
double INT_ERRO = 0;
int A= 1, TEMPO_ANTES = 0;
void LIGA_BOMBA_R2(){
  if(PID == 1){
    if(A == 1){
```

```
      TEMPO_ANTES = millis();
      A = 0;
    }
        ERRO2 = VOLUME - VOLUME_SP;
        INT_ERRO = INT_ERRO + ERRO2*(millis()-TEMPO_ANTES);
        PWM_BOMBA_R2 = PWM_ERROZERO2 + K2*ERRO2+ K2*INT_ERRO/TIV;
        if(PWM_BOMBA_R2 < 0){
           PWM_BOMBA_R2 = 0;
        }
        if(PWM_BOMBA_R2 > 100){
           PWM_BOMBA_R2 = 100;
        }
  }
        TEMPO_ANTES = millis();
        analogWrite(BOMBA_R2, map(PWM_BOMBA_R2,0,100,0,253));
}

void LIGA_BOMBA_R3(){

        analogWrite(BOMBA_R3, map(PWM_BOMBA_R3,0,100,0,253));
}

void LIGA_BOMBA_J(){
    if(PID == 1){
        ERROJ = TEMPERATURA_R1 - 25;
        PWM_BOMBA_J = PWM_ERROZEROJ + KJ*ERROJ;
        if(PWM_BOMBA_J < 0){
           PWM_BOMBA_J = 0;
        }
        if(PWM_BOMBA_J > 100){
           PWM_BOMBA_J = 100;
        }
  }
        analogWrite(BOMBA_J, map(PWM_BOMBA_J,0,100,0,253));
}
void LIGA_RESISTENCIA(){
  if(TEMPERATURA_R2 < T_SP){
        digitalWrite(RESISTENCIA, HIGH);
        RESIS =1;
  }else{
    digitalWrite(RESISTENCIA, LOW);
```

```
    RESIS = 0;
  }
}
void EXIBE_DADOS2(){
  Serial.print(millis()/1000);
    if(ATIVA_AGITACAO){
        Serial.print(", ");
        Serial.print(VELOCIDADE_AGITADOR_ANTERIOR);

    }if(ATIVA_AGITACAO2){
        Serial.print(", ");
        Serial.print(VELOCIDADE_AGITADOR_ANTERIOR2);

    }
    if(ATIVA_TEMPERATURA){
        Serial.print(", ");
        Serial.print(TEMPERATURA_R1);
        Serial.print(", ");
        Serial.print(TEMPERATURA_R2);
        Serial.print(", ");
        Serial.print(TEMPERATURA_J);
    }
    if(ATIVA_BOMBA_R1){
        Serial.print(", ");
        Serial.print(PWM_BOMBA_R1);
    }
    if(ATIVA_BOMBA_R2){
        Serial.print(", ");
        Serial.print(PWM_BOMBA_R2);
    }
    if(ATIVA_BOMBA_R3){
        Serial.print(", ");
        Serial.print(PWM_BOMBA_R3);
    }
    if(ATIVA_BOMBA_J){
        Serial.print(", ");
        Serial.print(PWM_BOMBA_J);
    }
    if(ATIVA_RESISTENCIA){
        Serial.print(", ");
        Serial.print(RESIS);
```

```
    }
    if(ATIVA_NIVEL){
        Serial.print(", ");
        Serial.print(VOLUME, 2);
    }

  Serial.print(", ");
  Serial.print(T_SP);
  Serial.print(", ");
  Serial.println(VOLUME_SP);

}


 void EXIBE_DADOS1(){
  Serial.print("\n*************************************************************************\n");
    if(ATIVA_AGITACAO){
        Serial.print("Pot. Agitacao: ");
        Serial.println(VELOCIDADE_AGITADOR_ANTERIOR);
    }if(ATIVA_AGITACAO2){
        Serial.print("Pot. Agitacao 2: ");
        Serial.println(VELOCIDADE_AGITADOR_ANTERIOR2);
    }
    if(ATIVA_TEMPERATURA){
        Serial.print("Temp R1: ");
        Serial.println(TEMPERATURA_R1);
        Serial.print("Temp R2: ");
        Serial.println(TEMPERATURA_R2);
        Serial.print("Temp Jaq: ");
        Serial.println(TEMPERATURA_J);
    }
    if(ATIVA_BOMBA_R1){
        Serial.print("BR1: ");
        Serial.println(PWM_BOMBA_R1);
    }
    if(ATIVA_BOMBA_R2){
        Serial.print("BR2: ");
        Serial.println(PWM_BOMBA_R2);
    }
       if(ATIVA_BOMBA_R3){
        Serial.print("BR2: ");
```

```
        Serial.println(PWM_BOMBA_R3);
    }
    if(ATIVA_BOMBA_J){
        Serial.print("BJq: ");
        Serial.println(PWM_BOMBA_J);
    }
    if(ATIVA_RESISTENCIA){
        Serial.print("RESISTENCIA: ");
        if(RESIS == 1){
          Serial.println("LIGADA");
        }else{
         Serial.println("DESLIGADA");
        }
    }
    if(ATIVA_NIVEL){
        Serial.print("VOLUME: ");
        Serial.println(VOLUME);
    }
}
```

APÊNDICE B - Algoritmo utilizado no ESP32 para comunicação Arduino® -internet

```
#define CAYENNE_PRINT Serial
#include <CayenneMQTTESP32.h>
#include <WiFi.h>

// WiFi network info.
char ssid[] = "NOME_DA_REDE";
char wifiPassword[] = "SENHA_DA_REDE";

// Cayenne authentication info. This should be obtained from the Cayenne Dashboard.
char username[] = "6f64a310-bdd9-11e9-b01f-db5cf74e7fb7";
char password[] = "4a58ff3cd31ed7e3e8de5af97283d987ed9199bc";
char clientID[] = "b357f1e0-bf68-11e9-b6c9-25dbdbf93e02";

#define VIRTUAL_CHANNEL 5
#define ACTUATOR_PIN 3

double TEMP1, Temp_SP, V_SP;
double TEMPJ;

//HardwareSerial Serial1(2);

void setup() {
  Serial.begin(9600);
  Serial2.begin(9600);
  Cayenne.begin(username, password, clientID, ssid, wifiPassword);
}

void loop()
{
    if(Serial2.available() > 0){
        switch (Serial2.read()){
          case 'A':
                TEMP1 = 100*Serial2.parseFloat();
                Serial.print("A");
                Serial.println(TEMP1);
                break;
          case 'B':
                TEMPJ = 100*Serial2.parseFloat();
                Serial.print("B");
                Serial.println(TEMPJ);
                break;
```

```
      }
    }
  Cayenne.loop();
}
CAYENNE_IN(5)
{
  int value = getValue.asInt();
  Temp_SP = value;
  Serial.print("T");
  Serial.println(Temp_SP);
  Serial2.print("T");
  Serial2.println(Temp_SP);
}

CAYENNE_IN(6)
{
  double value = getValue.asDouble();
  V_SP = value;
  Serial.print("V");
  Serial.println(V_SP);
  Serial2.print("V");
  Serial2.println(V_SP);
}

CAYENNE_OUT_DEFAULT()
{
  Cayenne.virtualWrite(0, millis()/1000);
  Cayenne.virtualWrite(1, TEMP1/100);
  Cayenne.virtualWrite(2, TEMPJ/100);
}
```