UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA QUÍMICA

PROJETO DE CONTROLE DE UM REATOR CERÂMICO PARA PRODUÇÃO DE CLORETO DE ALUMÍNIO ANIDRO UTILIZANDO MICROCONTROLADORES DE HARDWARE LIVRE

Christian de Oliveira Martins

Trabalho de graduação apresentado ao
Departamento de Engenharia Química
da Universidade Federal de São Carlos

Orientador: Prof. Felipe Fernando Furlan

Coorientador: Prof. Adalberto Teógenes Tavares Júnior

São Carlos-SP

2021

BANCA EXAMINADORA

Trabalho de Graduação apresentado no dia 16 de Novembro de 2021 perante a seguinte banca examinadora:

Orientador: Felipe Fernando Furlan, DEQ/UFSCar

Coorientador: Prof. Adalberto Teógenes Tavares Júnior, Parque Tecnológico de Itaipu

Convidado: Andrew Milli Elias, Embrapa

Professor da Disciplina: José Mansur Assaf

AGRADECIMENTOS

Agradeço primeiramente aos integrantes grupo de seminários de reatores, composto durante sua história por Adalberto, Andrew, Francielle, David, Felipe, Henrique e Kenny. Pois todos me auxiliaram do começo ao fim dessa jornada pelo TG.

Agradecimentos especiais para Felipe por me guiar nos caminhos da pesquisa em Engenharia Química durante a graduação, para Adalberto, por me coorientar, para Henrique, por caminhar comigo com os mesmos objetivos profissionais e inspirar o tema deste Trabalho de Graduação e a Andrew, com todo seu suporte de conhecimentos de eletrônica.

Agradeço também a Professora Caterina, por me iniciar no caminho da pesquisa e ser minha “mãe científica”.

Agradeço aos meus amigos pessoais que aturaram horas de monólogos sobre meu Trabalho de Graduação e como eu gosto dele, em especial ao César que perdeu alguns cabelos vendo as gambiarras que fazia.

Por último, mas não menos importante, agradeço a minha família que sempre me apoiou no meu caminho acadêmico mesmo sem saber o que exatamente eu fazia.

RESUMO

O cloreto de alumínio ($AlCl_3$) é um sal frequentemente utilizado na indústria por ser um ácido de Lewis, o que o torna um bom catalisador na manufatura de diversos compostos orgânicos. Entre suas principais aplicações temos a produção de etil benzeno e de tinturas. Várias matérias-primas podem ser usadas na síntese do $AlCl_3$, através da cloração de uma fonte de alumínio. Em uma dessas vias de síntese, reage-se cloreto de hidrogênio gasoso e alumínio metálico a altas temperaturas, produzindo o cloreto de alumínio anidro e gás hidrogênio. Reatores cerâmicos a base de alumina podem ser usados nesse caso, dada a natureza corrosiva da matéria-prima e para evitar a contaminação do produto com outros metais, comumente com ferro proveniente de reatores metálicos. Reatores cerâmicos são sensíveis a variações drásticas de temperatura, podendo ocasionar fraturas por ciclo ou choque térmico. Por isso o controle de temperatura deve ser suave, sem variações bruscas na potência fornecida. Sendo assim, o objetivo deste trabalho é desenvolver um controle de temperatura do tipo feedback para um reator cerâmico tubular para produção de cloreto de alumínio anidro. O modelo utilizado é um controle PID (proporcional-integral-derivativo), tradicional em controle de processos. A atuação do PID na variável manipulada é proporcional ao desvio na variável controlada (diferença entre o valor atual e o desejado), à integral e à derivada deste desvio no tempo. Dois microcontroladores programáveis foram inicialmente triados para serem empregados no controle do reator. O primeiro foi o NodeMCU ESP-8266 com módulo ESP12F, um microcontrolador com disponibilidade de comunicação via Wi-Fi com outros dispositivos, possibilitando inclusive seu uso para IoT, a “internet das coisas”. Este microcontrolador utiliza um processador 32 bits com *clock* de até 80 MHz; uma vantagem com relação ao Arduino Uno, que, por sua vez, possui *clock* de apenas 16 MHz. O segundo controlador utilizado foi o próprio Arduino Uno, que apesar de uma menor capacidade de processamento, está mais consolidado no mercado, conferindo a ele uma grande vantagem com relação a suporte. O controle da temperatura foi realizado pela manipulação do fornecimento de calor ao reator através do controle da potência fornecida à resistência elétrica responsável pelo aquecimento. O método do relé foi empregado para o ajuste dos parâmetros PID no reator sem que a reação estivesse ocorrendo. Com esse controle, diminuiu-se o tempo de aquecimento até a temperatura de operação em aproximadamente duas vezes e as variações de temperatura causadas por perturbações externas.

ABSTRACT

Aluminium chloride ($AlCl_3$) is a salt commonly applied in industry since it is a Lewis acid. Such feature makes aluminiun chloride a good catalyst in the production of several organic compounds. Between its main applications, there is the production of ethyl benzene and dyes in general. Many raw materials can be used in $AlCl_3$ synthesis by the chlorination of a aluminum source. In one of these paths, vapour hydrogen chloride reacts with metallic aluminum in high temperatures, producing aluminum chloride and hydrogen gas. Alumina based ceramic reactors are used in this case, since the raw materials have a corrosive behavior and to avoid possible contamination of the product with other metals, such as iron from metallic alloy reactors. Ceramic reactors are sensible to fast temperature variations, which could cause fractures by cycle or thermal shock. For this reason the temperature control must be smooth, without fast variations of the supplied power. Therefore, the objective of this work is tuning a temperature feedback control system for a tubular ceramic reactor to produce anhydrous aluminum chloride. The model used is a PID (proportional-integral-derivative) control, which is traditionally used in process control. PID control action on the manipulated variable is proportional to the deviation in the controlled variable (difference between its current value and the desired value), to the integral and to the derivative of this deviation. Two programmable microcontrollers were chosen for the reactor control. The first one was the nodeMCU ESP-8266 with a ESP12F module. This microcontroller can communicate by Wi-Fi signal with other devices, which allows its use for IoT, the “Internet of Things”. Furthermore, it uses a 32-bit processor with a maximum clock of 80 MHz, a good advantage in relation to Arduino Uno, which have a clock of 16 MHz. The second controller used was the Arduino Uno itself. Despite its lower processing capacity, it is more consolidated in market, so it has an advantage in terms of support. The temperature control was realized by the manipulation of the reactor heat supply through the control of the power provided to resistance heating alloy. The relay method was applied for the initial tuning of PID parameters in the reactors without reaction. With this control, we reduced the time of heating until the operating temperature of the reactor about 2 times and the temperature variation caused by external perturbations.

SUMÁRIO

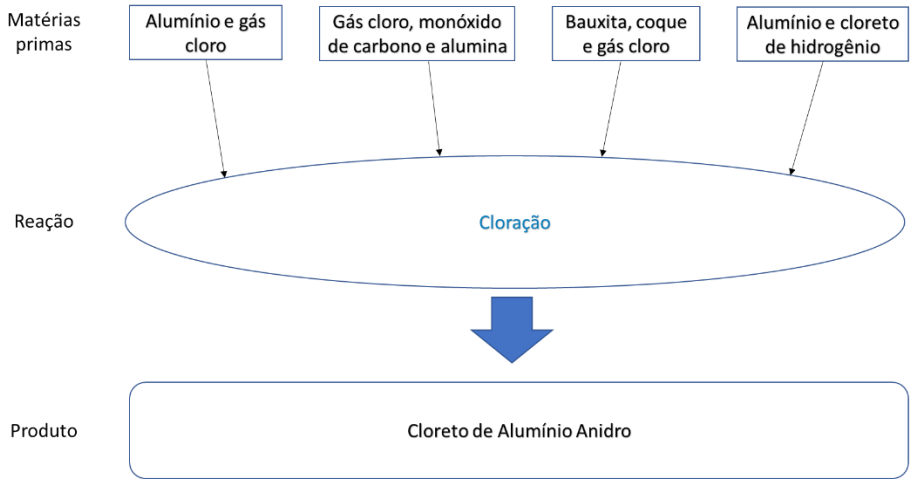
LISTA DE FIGURAS

LISTA DE TABELAS

1. INTRODUÇÃO

O cloreto de alumínio anidro é um sal, cujo principal uso é como catalisador de Friedel-Crafts. Sua produção ocorre por diversas vias, mas podemos generalizá-las em quatro grandes rotas. A Figura 1.1 contém um diagrama que elucida tais rotas, das quais a primeira é a utilização de alumínio metálico e gás cloro através de uma reação de cloração de metal. A segunda é o uso de gás cloro, monóxido de carbono e alumina através de uma reação de cloração catalítica. A terceira é o uso de bauxita, carvão coque e gás cloro através de uma cloração. A última é o uso de alumínio e cloreto de hidrogênio também por cloração (JENKINS, 2016).

Figura 1.1. Várias rotas de produção de $AlCl_3$ existentes.



Fonte: Acervo Pessoal.

Este composto é amplamente usado na indústria como ácido de Lewis e seus usos e mecanismos já são estudados a séculos; como podemos observar no livro de ELBS (1891), livro de síntese orgânica que dedica 45 páginas ao seu uso como catalisador de Friedel-Crafts. Este uso ocorre na produção de diversos compostos orgânicos formados através de mecanismos de reação como isomerização, alquilação e polimerização. Um exemplo é a produção de etilbenzeno, que é normalmente realizada através da alquilação do benzeno.

Esse composto pode ser utilizado posteriormente na produção de estireno e antraquinona, utilizados na produção de poliestireno (isopor®) e tinturas respectivamente. Outra aplicação do cloreto de alumínio é na produção de tetracloroaluminato de sódio ($NaAlCl_4$) que pode ser utilizado como eletrólito (em sua forma fundida) nas baterias de sódio (SUDWORTH, 2001).

A reação de síntese do cloreto de alumínio anidro, mesmo que possua várias rotas, ocorre habitualmente em reatores cerâmicos, devido a natureza corrosiva dos reagentes.

A Fundação Parque Tecnológico de Itaipu possui um reator cerâmico a base de alumina para produção experimental de cloreto de alumínio anidro, utilizando alumínio e cloreto de hidrogênio para a sua produção. A reação ocorre através da adição do cloreto de hidrogênio no reator com alumínio sólido à 600 ºC.

Uma característica marcante destes reatores cerâmicos utilizados é a baixa resistência a choques térmicos, os quais podem causar rachaduras no reator. Visto que o reator opera a altas temperaturas, o aquecimento e o controle de temperatura do reator devem ser feitos de modo a preservar a estrutura física do reator e, ao mesmo tempo, manter a temperatura dentro da especificação para que a reação ocorra de forma eficiente.

Nesse contexto, o objetivo deste trabalho foi implementar um controle de temperatura para o reator que compense as perturbações que podem ocorrer durante a reação sem causar variações bruscas de temperatura que possam danificar a estrutura cerâmica do reator.

Um dos modelos mais comuns de controle de processos é a realimentação (feedback) via controle PID em série, que corrige quaisquer perturbações que ocorram no sistema de forma proporcional ao erro, a integral do erro e a derivada do erro.

Para utilizar tal modelo de controle, é necessário realizar testes e experimentos para extrair parâmetros do comportamento do sistema a fim de ajustar devidamente a forma com que o controlador irá agir. Um dos métodos mais empregados é o método do relé (ÅSTRÖM; HÄGGLUND, 1984) por ser de fácil implementação, seguro e resultar em parâmetros de controle robustos.

E então, o controlador para efetivamente aplicar as leis de controle foi determinado visando um controlador que garantisse a segurança física dos equipamentos, o que motivou

a escolha do hardware livre nodeMCU ESP-8266, pois este permite a aplicação de IoT via Wi-Fi e, portanto, monitoramento e operação remota (ZAIT, 2018), não necessitando do uso de um computador conectado via cabo.

Também foi utilizado um Arduino Uno, outro hardware livre (“Arduino Uno Rev3 — Arduino Online Shop,” [s.d.]), como comparativo uma vez que este é mais bem consolidado no mercado e possui amplo suporte online, servindo de base para verificar comparativamente funcionamento com o NodeMCU ESP8266.

2. REVISÃO BIBLIOGRÁFICA

Para o bom cumprimento da proposta deste trabalho de graduação, é necessário conhecer o histórico e a produção do produto final, o cloreto de alumínio anidro, assim como os principais conceitos de controle de processos e como efetivamente realizar este controle.

2.1. Cloreto de alumínio anidro

O cloreto de alumínio já é estudado a mais de um século, ELBS (1891) dedicou parte de seu livro para falar do uso do cloreto de alumínio como catalisador de reações de Friedel-Crafts na síntese de hidrocarbonetos, cetonas, fenóis e muitos outros.

Na década de 10 do século XX, muito se publicava sobre os possíveis usos do material para indústria petrolífera e da chuva de investimentos no seu uso, assim como da explosão do seu preço na década seguinte (MCAFEE, 1929).

Atualmente é um dos ácidos de Lewis mais usados industrialmente e de uso extremamente amplo, afinal o composto é utilizado como catalisador na produção de inúmeros compostos químicos orgânicos através de diferentes mecanismos, como isomerização, alquilação e polimerização. Mas, as principais aplicações do cloreto de alumínio estão relacionadas a produção de etilbenzeno e de antraquinona para produção de pigmentos e tinturas.

Inicialmente, o cloreto de alumínio era produzido comercialmente utilizando bauxita calcinada e carvão coque ao invés de alumínio. Entretanto posteriormente novos métodos foram surgindo reduzindo a corrosão e aumentando a pureza do produto. Atualmente, as principais matérias primas são alumínio metálico ou óxido de alumínio puro como fontes de alumínio e gás cloro ou ácido clorídrico como fontes de cloro.

Sua reação ocorre tipicamente em um reator cerâmico tubular. Neste trabalho considerou-se a rota de alumínio fundido e adição de cloreto de hidrogênio abaixo da

superfície do alumínio fundido. Essa reação é muito exotérmica (ΔH_R (298 K) = - 611,9 kJ/mol estimado baseado nas entalpias de formação) e quase instantânea, gerando um cloreto de alumínio gasoso. O controle desta reação ocorre a partir da variação da alimentação dos reagentes e do resfriamento do reator.

O processo utilizado pelo Parque Tecnológico de Itaipu possuí caráter experimental apenas e, portanto, segue um caminho próximo, porém com algumas peculiaridades: primeiramente, a fonte de cloro é ácido clorídrico em solução aquosa, gotejado em ácido sulfúrico concentrado, que sequestra a água da solução exaurindo o cloreto de hidrogênio gasoso. O gás entra em uma linha de escoamento com uma coluna de sílica para retirar qualquer água residual e então entra no reator, aquecido previamente até 600 ºC, já com o alumínio previamente inserido a essa temperatura. Com isso a reação ocorre, formando o cloreto de alumínio gasoso e gás hidrogênio, sendo o primeiro coletado finalizando a etapa de reação.

Em todas as rotas, após a formação do cloreto de alumínio vaporizado, o composto é levado por difusão a outro compartimento do reator que se encontra abaixo da temperatura de sublimação do composto (175 ºC, 1 atm). Assim que o composto entra em contato com uma superfície fria, no caso uma parede de condensação, o composto perde calor e ressublima, depositando-se como sólido na superfície do condensador, que deve ser retirado periodicamente.

Vale ressaltar que todos estes processos devem ser feitos sob atmosfera seca, pois a água em contato com o cloreto de alumínio anidro reage o mesmo segundo a reação $nAlCl_3 + mOH^- \rightarrow Al_n(OH)_mCl_{3n-m} + mCl^-$ (KIRK, 1992), sendo impossível de purificá-lo por rotas físicas tais como variações na temperatura e/ou pressão do sistema.

2.2. Controle de Processos

2.2.1. Conceitos principais

As plantas químicas tem se tornado cada vez mais complexas, especialmente devido a exigências com relação a minimização de impactos ambientais, de flexibilidade de produção, aumento na qualidade dos produtos e redução de estoque e segurança. Isso tem levado os

processos cada vez mais aos seus limites operacionais, o que aumenta a demanda por sistemas de controle cada vez mais rigorosos (SEBORG et al., 2017; STEPHANOPOULOS, 1984).

Dentro da linguagem de controle de processos, as variáveis do sistema podem ser divididas em variáveis de entrada e de saída. As variáveis de entrada ainda podem ser divididas em manipuláveis (ou ajustáveis) e perturbações. Já as de saída em variáveis medidas (ou mensuráveis) e não medidas (não mensuráveis) (STEPHANOPOULOS, 1984).

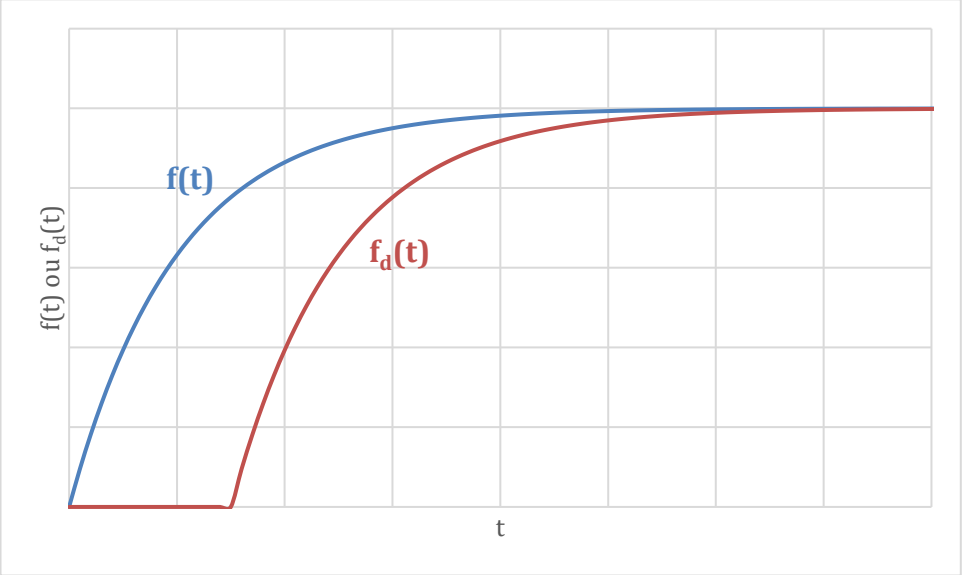
Em um sistema de controle é conveniente que as variáveis que representam os objetivos de controle, ou seja, as variáveis as quais efetivamente queremos controlar, sejam medidas diretamente. Caso isso não seja possível, recomenda-se medir variáveis relacionadas a estas (SEBORG et al., 2017; STEPHANOPOULOS, 1984). Desta forma é possível aplicarmos uma lei de controle. Lei de controle é o roteiro que o controlador irá seguir para manter o processo dentro das restrições impostas.

O projeto do controlador ainda depende do conhecimento do comportamento do sistema. Sendo assim, é necessário modelar a dinâmica do processo de alguma forma. Para isso existem dois enfoques principais: o enfoque experimental e o enfoque teórico. O enfoque experimental se baseia em gerar uma perturbação no sistema e observar o comportamento do processo, facilitando a criação do modelo. Entretanto, esse método pode ser dispendioso e limitado. Já o método teórico utiliza os princípios básicos da engenharia química para gerar o modelo matemático, o que faz com que o modelo funcione de forma mais geral. Porém, se o processo for demasiadamente complexo a modelagem pode se tornar dispendiosa, de difícil resolução ou, por outro lado, muito simplificada, se distanciando do comportamento real do processo (SEBORG et al., 2017; STEPHANOPOULOS, 1984).

Tipicamente os modelos de dinâmica do processo são linearizados e categorizados de acordo a ordem da equação diferencial. O mais comumente utilizado consiste em um modelo de equação diferencial de primeira ordem com tempo morto. O tempo morto é o tempo de atraso entre o momento da aplicação da perturbação na variável de entrada e o momento em que a interferência dessa alteração é notada na saída. Essa característica é muito comum em processos que envolvem fenômenos de transporte por dependerem de fenômenos de condução ou escoamento para que alguma perturbação seja notada na variável de saída ((SEBORG et al., 2017).

Um processo de primeira ordem tem um comportamento tipicamente exponencial. Geralmente, é aplicada uma perturbação degrau na entrada para estudar o comportamento e extrair os parâmetros da dinâmica deste processo a partir da resposta (SEBORG et al., 2017; STEPHANOPOULOS, 1984). O comportamento de um sistema de primeira ordem sem e com tempo morto, assim como o equacionamento deste tipo de sistema é mostrado na Figura 2.1 e nas equações: Equação 2.1, na forma diferencial geral de um sistema de primeira ordem; na Equação 2.2, a solução da equação diferencial caso a entrada seja um degrau sem tempo morto e na Equação 2.3, com tempo morto:

Figura 2.1. Uma função dependente do tempo com e sem tempo morto.



Fonte: Acervo Pessoal.

$$\tau_p \cdot \frac{dy}{dt} + y = K_p \cdot u \qquad (2.1)$$

Onde τ_p é a constante de tempo do sistema, K_p é o ganho estacionário, y é a variável de saída, u é a variável de entrada e t o tempo.

$$y = K_p \cdot A \cdot \left(1 - e^{-\frac{t}{\tau_p}}\right) \tag{2.2}$$

Onde A é a amplitude do degrau na entrada.

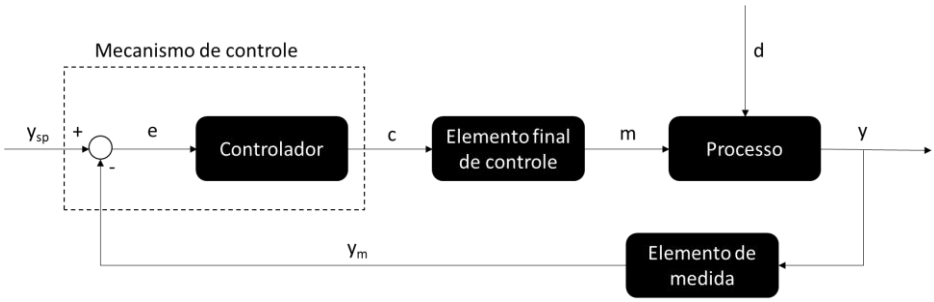
$$\begin{cases} y = 0 \quad para\ 0 \leq t < \tau_d \\ y = K_p \cdot A \cdot \left(1 - e^{-\frac{t-\tau_d}{\tau_p}}\right) \quad para\ t \geq \tau_d \end{cases} \tag{2.3}$$

Onde τ_d é o tempo morto.

2.2.2. Controle por realimentação (feedback)

O controle por realimentação é caracterizado por um modelo de controle baseado na correção de erros após a detecção. Um medidor detecta a medida da variável controlada, este sinal é comparado com uma referência chamada "*set point*" gerando um valor de desvio ou erro. Esse erro, por sua vez, é enviado ao controlador que atua enviando um sinal a ferramenta que efetivamente manipula a variável de entrada (como uma válvula, por exemplo) que por fim modifica a resposta do processo, ou seja, a variável de saída. A Figura 2.2 apresenta um diagrama de blocos típico de um sistema de controle por realimentação (SEBORG et al., 2017; STEPHANOPOULOS, 1984).

Figura 2.2. Diagrama de blocos de um sistema em loop fechado de realimentação.



Fonte: Acervo Pessoal.

Existem alguns elementos essenciais no controle de *feedback* que são listados abaixo:

- Elemento de medida ou sensor/transdutor: este é o conjunto medidor/conversor de sinal do sistema. Por exemplo, um termopar é o sensor que mede a temperatura através da geração de uma diferença de potencial, que é acoplada a um transdutor que retorna o sinal de temperatura em si.
- Mecanismo de controle: é a ferramenta que aplicará a lei de controle. Este elemento fará a comparação do sinal do elemento de medida com o *set point* e enviará o sinal de ação de controle para a regulação do processo.
- Elemento final de controle: ferramenta utilizada para modificar a variável de entrada. Pode ser, por exemplo, uma válvula ou um potenciostato.

Deste modo, o sistema de controle executa certas tarefas de forma a garantir a estabilidade do processo dentro dos critérios estabelecidos, podemos listar estas tarefas da seguinte forma:

- Primeiro mede-se a variável de saída (y) e envia-se o sinal medido (y_m) até o mecanismo de controle.
- No mecanismo de controle, o valor medido é comparado ao valor desejado, *set point* (y_{sp}), e gera um valor de desvio ou erro (e). O erro é enviado ao controlador, o qual atua pela lei de controle calculando um sinal de referência para o elemento final de controle (c).

- O elemento final de controle recebe o sinal do controlador e age modificando a variável manipulada (m), que é uma variável de entrada do processo.
- O processo recebe o novo valor da variável manipulada, podendo também sofrer impacto de interferências externas que não são manipuladas, que são chamadas de carga (d) e então gerando um valor de saída, a variável controlada (y), reiniciando o ciclo.

Existem três ações de controle por realimentação tradicionais que podem ser combinadas entre si. São elas a ação proporcional, a ação integral e a ação derivativa. O controlador é o equipamento encarregado de executar essas ações de modo a corrigir o erro observado (SEBORG et al., 2017; STEPHANOPOULOS, 1984)..

A ação proporcional implica que o sinal de controle será proporcional ao erro, podendo ou não ser acrescentado um viés. Esta ação pode ser aplicada sozinha ao controlador, gerando assim um controle proporcional, que segue os moldes da equação observada na Equação 2.4. A ação proporcional aumenta a velocidade de reação do sistema, mas ela por conta própria não elimina o erro do processo, ou seja, apresenta um desvio permanente (SEBORG et al., 2017; STEPHANOPOULOS, 1984).

$$c(t) = c_s + K_c \cdot e(t) \tag{2.4}$$

Onde c_s é o viés, que define o valor do sinal de controle quando o erro é nulo e K_c é o ganho proporcional, definido pelo operador, define a sensibilidade da ação de controle. Um valor alto dessa constante implica que o mínimo valor de erro resultará em uma ação rápida e intensa de controle. Se o ganho proporcional for excessivamente alto, o sistema ganha aspectos de um controle *on/off* (SEBORG et al., 2017; STEPHANOPOULOS, 1984).

A ação integral é uma ação proporcional à integral do erro. Ela existe para que seja eliminado o erro que pode permanecer após o uso da ação proporcional. Como ela corrige o processo sendo proporcional a integral do erro, quanto mais tempo existir desvio entre o valor medido e o *set point*, maior será seu efeito sobre o sistema. Ela é associada a ação proporcional, tendo seu controle chamada de controle proporcional-integral e possui o

equacionamento evidenciado na Equação 2.5 (SEBORG et al., 2017; STEPHANOPOULOS, 1984).

$$c(t) = c_s + K_c \cdot e(t) + \frac{K_c}{\tau_I} \cdot \int_o^t e(t)dt \quad (2.6)$$

Onde τ_I é o tempo integral, valor definido pelo operador junto com o ganho proporcional (K_c). O valor de $1/\ \tau_I$ representa a taxa de ajuste, pois caso haja um erro constante no tempo, este valor representa o tempo para que a ação integral gere um efeito equivalente à ação proporcional.

Já a ação derivativa acrescenta um termo proporcional a derivada do erro. Este termo não elimina o desvio permanente como o termo integral, mas seu uso serve para amortecer as ações de controle dos outros dois tipos (SEBORG et al., 2017; STEPHANOPOULOS, 1984). Alguns processos podem apresentar dinâmicas de difícil controle. Deste modo, o controle proporcional-integral pode gerar mudanças abruptas no processo, podendo apresentar riscos. Sendo assim, a ação derivativa reduz, até certo nível, essas possíveis mudanças drásticas de ação de controle. Quando ela está associada às ações proporcional e integral, ela ganha características mostradas na Equação 2.7. Este tipo de controle é chamado de controle proporcional-integral-derivativo, ou, simplesmente, controle PID, um modelo de controle bem tradicional e amplamente utilizado.

$$c(t) = c_s + K_c \cdot e(t) + \frac{K_c}{\tau_I} \cdot \int_o^t e(t)dt + K_c \cdot \tau_D \cdot \frac{de(t)}{dt} \quad (2.7)$$

Onde τ_D é o tempo derivativo, selecionado pelo operador em conjunto com o ganho proporcional (K_C) e o tempo integral (τ_I).

Essencialmente, cada uma destas ações deve ser ponderada de acordo com as características do sistema como um todo. Desta forma, existem diversos métodos para encontrar-se parâmetros de base para aplicar no controlador, este processo é chamado de projeto ou sintonia de controladores. Os projetos de controladores são testados quanto aos

objetivos de controle, sendo testados em operação servo, quando se varia o *set point* do sistema, ou em operação carga, quando aplica-se uma perturbação externa ao sistema.

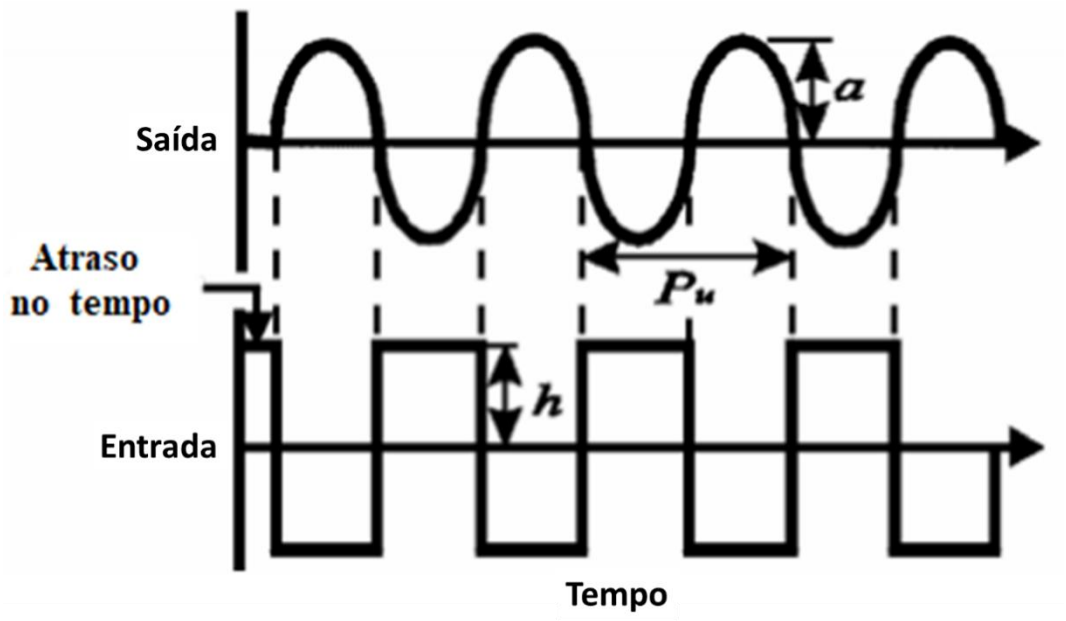
2.2.3. Projeto de controladores de realimentação pelo método do relé

Um método para obter tais parâmetros é o método de relé, método criado por ÅSTRÖM & HÄGGLUND (1984) e popularizado por LUYBEN (2002).Este método consiste em fechar uma malha de controle via feedback e aplicar um controle *on/off* em volta do ponto de operação do sistema temporariamente até que seja observada uma oscilação sustentada e desta oscilação se retira os devidos parâmetros. Esta etapa não necessariamente é composta por um controlador que desliga e liga o sistema totalmente, recomenda-se que a variação na variável de entrada seja entre 3 e 10% (YU, 2006) e então, assim que a variável medida cruze o *set point* o sinal de controle se inverte.

O procedimento geral do método consiste em:

1. Levar o sistema ao estado estacionário.
2. Fazer uma pequena perturbação na variável manipulada (recomendado entre 3 e 10% do valor no *set point*).
3. Assim que o valor da variável de saída atravessar o *set point* a variável manipulada é perturbada no sentido contrário (-3 a -10% do valor no *set point*).
4. Voltar ao passo 2 assim que a variável da saída cruzar novamente o *set point* até a oscilação se estabilizar como na Figura 2.3.
5. Por fim, ler o valor do período final (P_U) e calcular o ganho final (K_U) pela Equação 2.8.
6. Calcular os parâmetros de controle segundo as Equações 2.9 a 2.10.

Figura 2.3. Comportamento do teste de realimentação por relé para um sistema com ganho estacionário positivo após a estabilização da oscilação.



Fonte: Adaptado sob permissão de *Springer Nature*: Springer. Springer eBook. Relay Feedback, YU, C. C. Copyright Clearance Center. 2006.

O cálculo do ganho final é:

$$K_U = \frac{4 \cdot h}{\pi \cdot a} \tag{2.8}$$

Já os parâmetros do projeto PID são calculados pelas equações 2.9-2.11(ZIEGLER; NICHOLS, 1942).

$$K_C = 0{,}6 \cdot K_U \tag{2.9}$$

$$\tau_I = \frac{P_U}{2} \tag{2.10}$$

$$\tau_D = \frac{P_U}{8} \tag{2.11}$$

Dessa forma, tais características fazem do método do relé um ótimo método de projeto de controladores. Afinal permite o cálculo dos parâmetros do controlador de forma prática e segura, sem a necessidade de levar o sistema ao limite da estabilidade, como é o caso do método de Ziegler-Nichols em malha fechada (SEBORG, 2017).

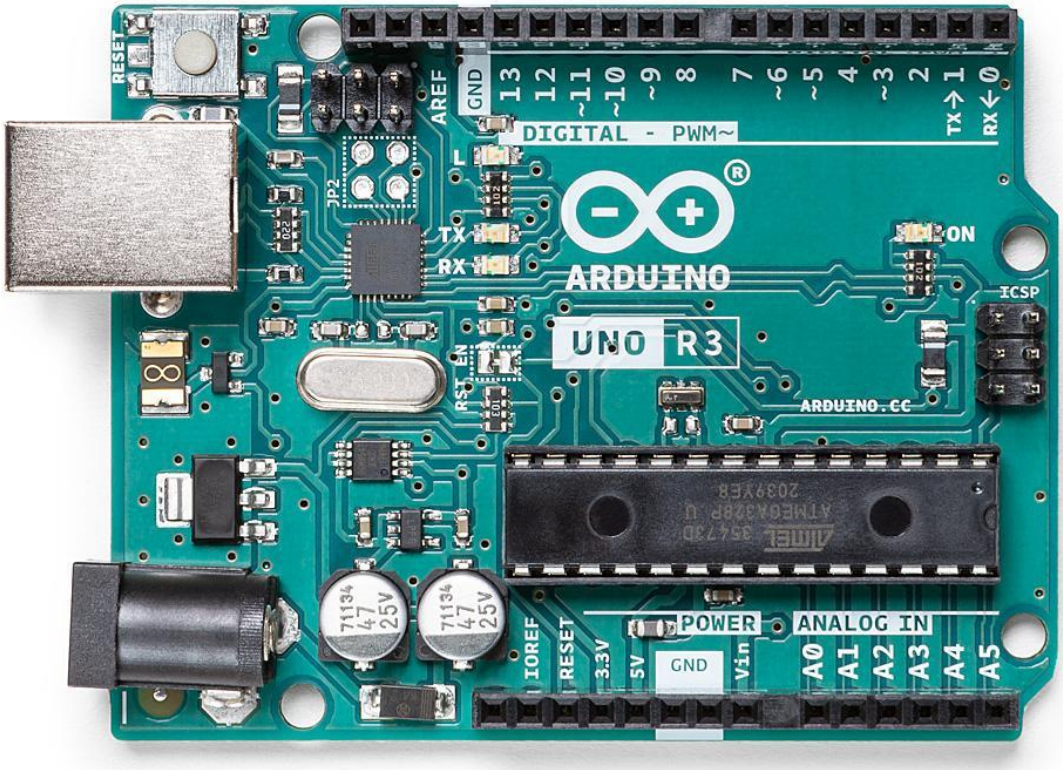
2.3. Microcontroladores de *Hardware* Livre

2.3.1. Arduino Uno

Arduino é a plataforma de criação de protótipos eletrônicos de *hardware* e *software livre*. Os Arduinos são arquitetados para que recebam informações do ambiente e emitam uma resposta baseada nessa informação adquirida, conferindo às placas deste programa uma gama de aplicações de acordo com inúmeros tipos de sensores que podem ser utilizados. Deste modo o Arduino pode ser implementado em uma diversidade imensa de campos, como a engenharia, a robótica, música, moda, entre outras (MULTILÓGICA-SHOP, [s. d.]).

Os microcontroladores das placas Arduino são programados por uma linguagem de programação de Arduino em si. Esta linguagem é baseada em C e C# e pode ser programada em um ambiente de desenvolvimento próprio dos Arduino, chamado Arduino IDE. Sendo assim, uma das placas de Arduino mais comuns é o Arduino Uno R3, que pode ser visto na Figura 2.4. Essa placa emprega um circuito integrado ATmega328P, desenvolvido pela Arduino. Esta placa possui 14 pinos digitais que servem tanto de entrada, quanto de saída e 6 entradas analógicas, sendo bastante utilizada na construção de protótipos. Isso se deve ao seu custo acessível de US$ 23,00 (cotação em outubro de 2021) no site oficial e por possuir tutoriais, suporte e comunidades voltadas ao seu uso amplamente divulgadas pela internet (ARDUINO, [s. d]; MANUAL DO MUNDO, 2019).

Figura 2.4. Placa de desenvolvimento do microcontrolador Arduino Uno R3



Fonte: Arduino, [s. d.].

2.1.1. NodeMCU ESP-8266 (ESP-12F)

NodeMCU é uma plataforma aberta de baixo custo para IoT. Ela possui um *firmware* adaptado para rodar em módulos como o ESP8266 e o ESP32, sendo o preço do NodeMCU ESP8266 por volta de US$ 6,99 (cotação de outubro de 2021) (ESP8266 SHOP, [s. d.]). Seu nome vem da combinação da palavra inglesa "*node"* com o termo "MCU" que quer dizer "*microcontroller unit" (YUAN, 2017).*

Figura 2.5. Placa de desenvolvimento NodeMCU ESP8266

Fonte: ELECROW, [s. d.]**.**

Assim como o Arduino, o NodeMCU utiliza de placas de desenvolvimento de *hardware* livre, mas este se baseia em chips da *Espressif*, que produz diversos microchips com suporte com conexão Wi-Fi através de *softwares* de rede TCP/IP. Outra semelhança desta com as placas do Arduino é a existência de uma comunidade que discute sobre o tema e desenvolve os mais diversos projetos através desses dispositivos (REDDIT, [s. d.], ESP8266 [s. d.])

A programação nestas placas é baseada em Lua. Entretanto, existe suporte da placa em ambientes de desenvolvimento de Arduino como o Arduino IDE permitindo a programação destes microcontroladores através da mesma programação utilizada para as placas do Arduino.

3. MATERIAIS E MÉTODOS

3.1. Materiais

3.1.1. Lâmpada Halógena

Lâmpada halógena clara H100 de 70W para rede elétrica de 127V e 50/60Hz, base E27, luz de cor branca morna 2700K

3.1.2. Microcontroladores de Hardware Livre

3.1.2.1. Arduino Uno R3

Placa microcontroladora baseado no chip ATmega328P, com 14 pinos digitais que servem tanto de entrada quanto saída de sinal, sendo 6 deles pinos PWM, 6 entradas analógicas, processador de 16 MHz de *clock*, tensão de operação de 5V, com alimentação recomendada de 7-12 V.

3.1.1.1. NodeMCU ESP8266 v3

Placa microcontroladora baseada em um chip ESP12-F, com 16 pinos digitais que também servem tanto para entrada quanto saída de sinal, sendo 4 deles pinos PWM, 1 entrada analógica, processador de 80 MHz de *clock*, tensão de operação de 3,3V, com alimentação de 7-12V e um módulo wi-fi.

3.1.2. Módulo dimmer com zero-cross

Circuito de modulo de dimmer AC para rede elétrica de 50/60Hz com 1 canal de entrada lógica de 3,3V/5V da RobotDyn.

3.1.3. Medidores de temperatura

3.1.3.1. LM35DZ

Sensor de temperatura de faixa de 0ºC a 100ºC. Precisão de 0,5 ºC. Tensão de operação: 4 a 30 V. Consumo de corrente: até 60 µA.

3.1.3.2. Termopar tipo K

Termopar de 1m. Faixa de medida de 0 a 800 ºC. Isolamento interno: Fibra de Vidro, Terminal forquilha.

3.1.4. Conversor analógico digital para termopar (MAX6675)

Tensão de Operação: 3 V a 5,5V. Temperatura de operação: -20 ºC a 86 ºC. Junta fria inclusa. Interface SPI.

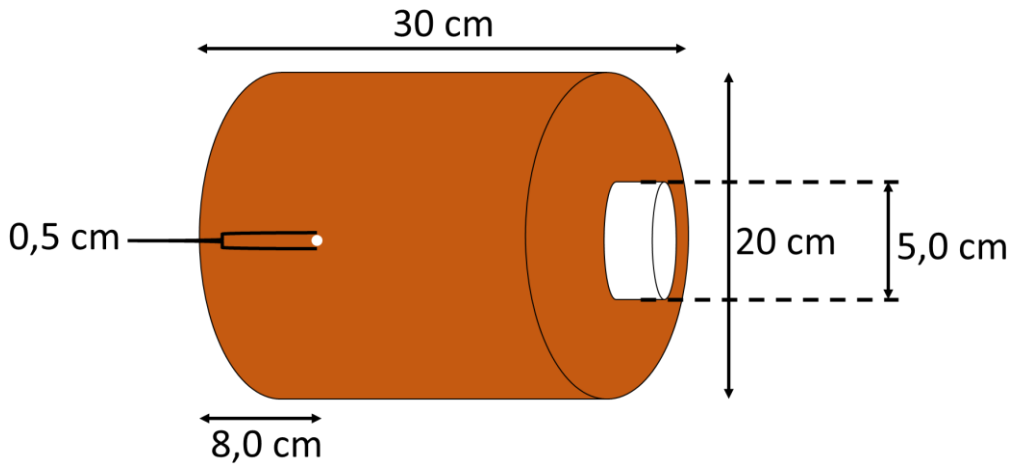
3.1.5. Regulador de voltagem

Regulador de voltagem JNG TDGC2 – Capacidade: 2000VA. Corrente máxima: 8 A. Entrada: 220V/50-60Hz. Saída: 0-250V.

3.1.6. Reator cerâmico de alumina

Reator cilíndrico: 36,0 cm de comprimento e 20,0 cm de diâmetro externo. Tubo de alimentação com diâmetro de 0,5 cm localizado a 8,0 cm do início do reator. Diâmetro de tudo de saída: 5,0 cm.

Figura 3.1. Croqui do reator.



Fonte: Acervo Pessoal.

3.1.7. Multímetros

Multímetro digital Minipa ET-1002 (Medida de tensão elétrica após o regulador de voltagem)

Multímetro digital DT-830B (Testes de continuidade e tensão no módulo dimmer)

Multímetro digital ET-2076A (Eventuais medidas de corrente elétrica)

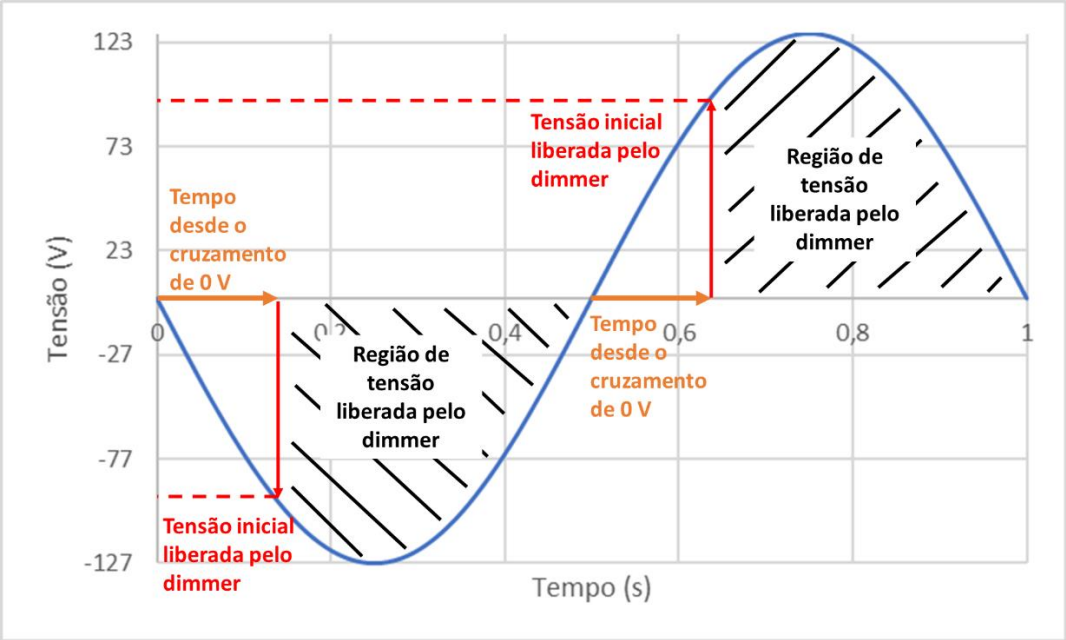
3.2. Procedimentos experimentais

3.2.1. Regulação da potência via dimerização

O método de regulação da potência se baseou na dimerização da tensão de alimentação dos sistemas via corte de fase. O módulo de dimmer tem como objetivo fazer com que o microcontrolador detecte sempre que a rede elétrica de tensão alternada cruza a faixa de 0 V e conte um determinado tempo antes de fechar o circuito, permitindo a passagem de corrente

até o elemento final. Como consequência o microcontrolador modula a região da tensão alternada que será aplicada ao equipamento, consequentemente modulando a potência alimentada ao equipamento. Um esquema geral pode ser visualizado na Figura 3.1.

Figura 3.2. Modulação de potência via dimerização.



Fonte: Acervo Pessoal.

3.2.2. Validação do Node MCU ESP-8266 em comparação com o Arduino Uno R3

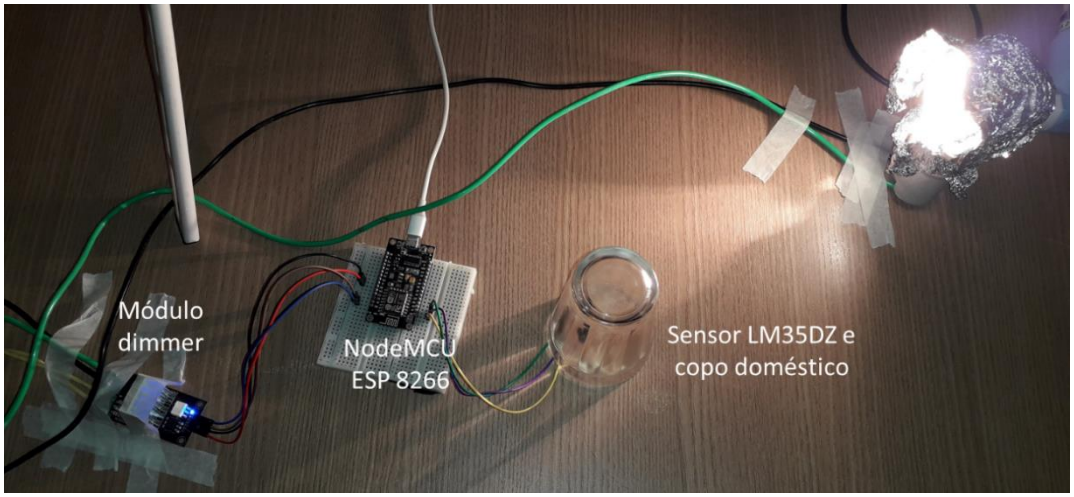
A ideia uso da modulação da onda da rede elétrica surgiu do projeto de TWOMEY ([201-]) para criação de uma lâmpada dimerizada. Entretanto o método utilizado neste material não se aplica perfeitamente ao NodeMCU ESP8266, uma vez que a base da dimerização foi o uso da biblioteca de Arduino “TimerOne” (STOFFREGEN et al. 2015) e este não é compatível com os chips da Espressif.

Sendo assim, este método se baseou em buscar uma alternativa para o uso da biblioteca “TimerOne” que tivesse funcionalidade para o ESP8266 e adaptar o projeto de TWOMEY ([201-]) de forma que o NodeMCU ESP8266 pudesse realizar as mesmas operações que o Arduino Uno R3 realiza utilizando a “TimerOne”, gerando no final uma lâmpada dimerizada com o ESP8266.

3.2.3. Experimento preliminar

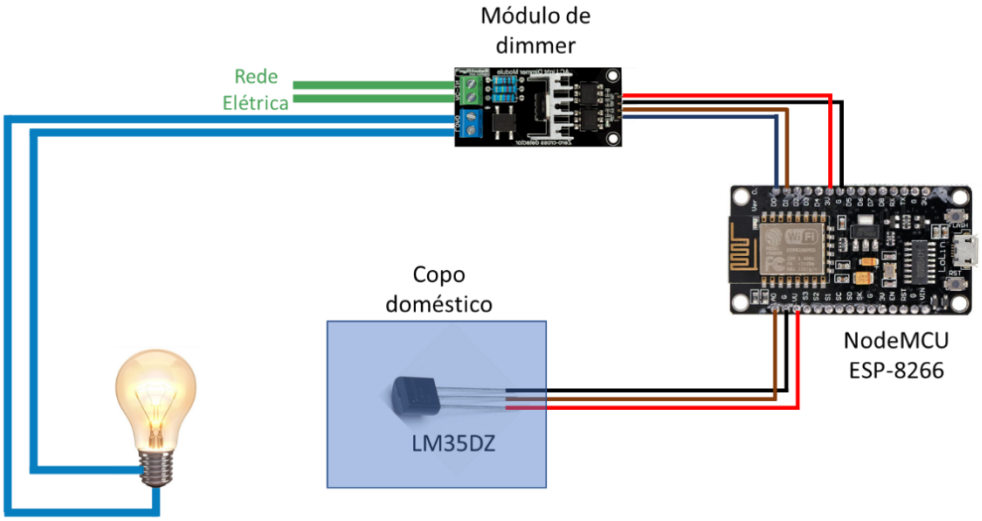
Primeiramente, as técnicas de ajuste de controle e o controlador obtido foram testadas em um sistema simplificado. A Figura 3.2 apresenta o aparato experimental que consiste em uma lâmpada halógena de 70W, o módulo dimmer descrito anteriormente, o controlador NodeMCU ESP 8266 e um sensor de temperatura LM35DZ. O sensor de temperatura foi fechado em vidro a uma distância de 21cm da lâmpada, que foi usada como fonte de calor. O ESP8266 juntamente com o módulo dimmer foram usados para controlar a potência dissipada pela lâmpada, utilizando o circuito descrito na Figura 3.3. O copo colocado sobre o sensor de temperatura teve o intuito de minimizar as distorções na medida de temperatura causadas por convecção.

Figura 3.3. Lâmpada envolvida com papel alumínio e sensor LM35DZ.



Fonte: Acervo Pessoal.

Figura 3.4. Diagrama esquematizando o sistema Lâmpada-LM35DZ.



Fonte: Acervo Pessoal

Inicialmente, analisou-se a dinâmica do sistema através da aplicação de um degrau de 100% da potência alimentada a lâmpada. A temperatura medida pelo sensor foi acompanhada até atingir o estado estacionário. O comportamento dinâmico do sistema foi caracterizado segundo um modelo linear em variável desvio de acordo com a ordem aparente do sistema. O ajuste do modelo aos dados experimentais foi feito através do *software* Origin, utilizando o método de otimização de Levenberg-Marquardt.

Na sequência, aplicou-se o método do relé para projetar o modelo de controle. Para isso utilizou-se o módulo dimmer de modo que este modulasse a potência fornecida para a lâmpada. Devido a problemas na sensibilidade do sensor de temperatura frente a perturbações na potência dissipada na lâmpada, escolheu-se trabalhar no estado estacionário obtido com uma potência correspondente a 50% da potência máxima. A variação escolhida foi de 50 pontos percentuais para cima e para baixo.

Uma vez que as oscilações do sistema se estabilizaram, os parâmetros característicos do sistema (Pu e Ku) foram obtidos e utilizados para calcular os parâmetros de controle para este sistema. Antes do teste do controle no aparato experimental, o modelo do sistema e o controle foram implementados no complemento Xcos do Scilab, para analisar o comportamento esperado e comparar com os resultados experimentais para uma operação servo com degrau de 5 ºC no set point. Por fim, o sistema de controle obtido foi ao aparato

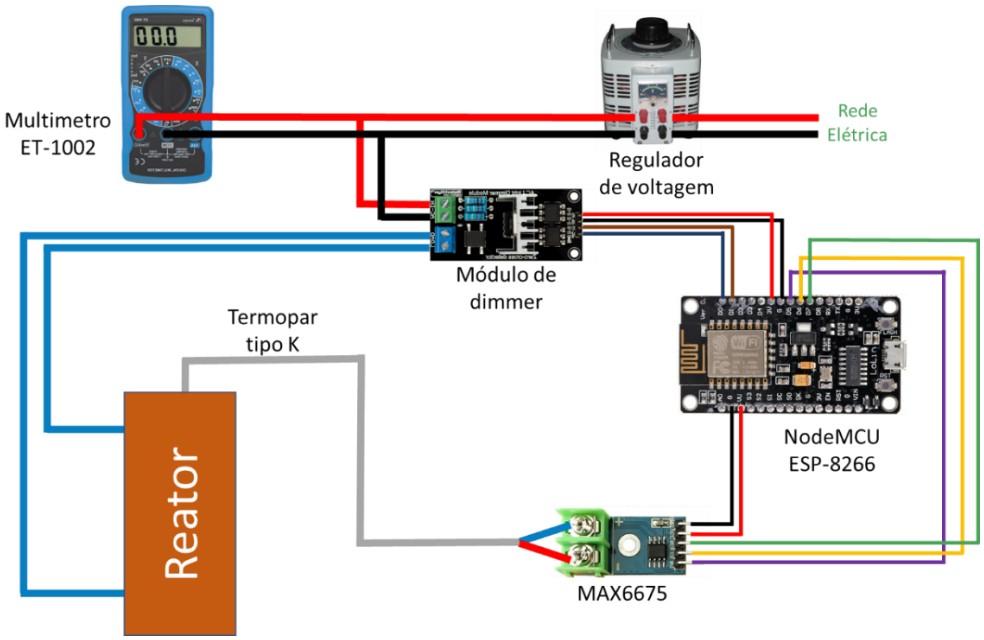
experimental (Figura 3.2), utilizando média móvel de três pontos para minimizar interferências de ruídos nas ações de controle. A qualidade da resposta foi analisada graficamente, comparando-a com a resposta simulada.

3.1.4. Experimentos no reator

3.1.4.1. Circuito

O circuito aplicado consistiu em um regulador de voltagem ligado a alimentação da rede elétrica 220V e alimentando o módulo de *dimmer*, com um multímetro em paralelo medindo a tensão alternada saindo do regulador de voltagem. O módulo de *dimmer* alimentado pela potência fornecida pelo regulador de voltagem era controlado pelo microcontrolador ESP8266 que regulava o quanto de potência iria para a resistência que aquecia o reator. A Figura 3.4 mostra um diagrama explicando visualmente a organização do circuito e do sistema em geral.

Figura 3.5. Diagrama da disposição do circuito e dos dispositivos nos experimentos do Reator.



Fonte: Acervo Pessoal.

3.1.4.2. Análise da dinâmica do reator

Dado a fragilidade do material cerâmico a variações abruptas de temperatura, o aquecimento até uma temperatura próxima do *set point* foi de aproximadamente 3ºC por minuto. Este aquecimento foi feito manualmente, através do aumento da tensão estipulada no regulador de voltagem, com o módulo de *dimmer* desacoplado em um primeiro momento. A potência final alimentada no reator foi de aproximadamente 188 W.

A temperatura do reator foi medida o tempo todo através de um termopar tipo K em conexão com os microcontroladores através de um módulo de conversão analógico-digital para leitura de termopares chamado MAX6675.

O reator foi deixado pela noite para estabilização de temperatura. Posteriormente, foi aplicado um degrau médio de 7,3 W na alimentação, para análise de dinâmica do sistema. Este degrau de potência foi escolhido baseado em uma extrapolação linear da relação potência dissipada para o reator – temperatura do reator. Deste modo, foram coletados os dados de temperatura em função do tempo até a estabilização. Este processo foi feito duas vezes e os dados foram modelados como um processo de segunda ordem com tempo morto. O valor de tempo morto foi definido por método gráfico e os parâmetros de ganho proporcional e tempo de processos foram obtidos via regressão pelo software *Origin*.

3.1.4.3. Aplicação do método de relé

Após o conhecimento do comportamento geral do sistema, o sistema foi novamente levado até a próximo do *set point*. Porém, desta vez o regulador de voltagem estava operando a uma tensão 1,38 vezes maior que a utilizada nas operações de degrau, sendo possível atingir a potência máxima, de aproximadamente 260 W. Mas deve-se considerar que o módulo de *dimmer* possui impedâncias, resultando em uma queda na tensão do sistema. Sendo assim, desta vez o controlador limitou a passagem da potência advinda do regulador de voltagem a 85%, ainda conferindo ao sistema uma temperatura próxima dos 600ºC.

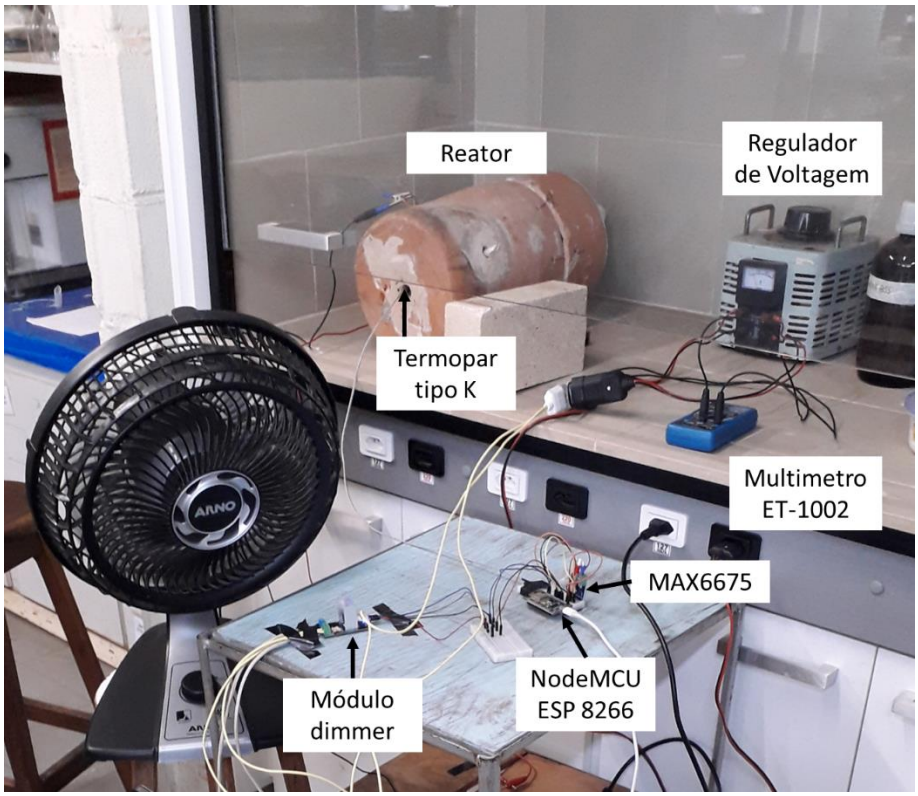
Em sequência, o método do relé foi aplicado com perturbações de +/- 5% com relação ao valor inicial da variável manipulada, correspondendo a uma variação na potência dissipada na resistência de 80% a 90% durante o processo. Assim que uma consistência nas oscilações causadas pelo método foi observada, pôde-se tirar os parâmetros e projetar o controle.

3.1.4.4. Verificação do projeto de controle

Para verificar a qualidade do projeto de controle, assim como no experimento com a lâmpada halógena, foi simulado o comportamento teórico do reator submetido ao controle PID na extensão Xcos do software Scilab. Esse teste serviu tanto para analisar o comportamento esperado quanto o tempo que o processo demoraria para eliminar o erro com uma perturbação servo.

Uma vez simulado, o reator foi submetido a um degrau no *set point* de 10 ºC já em malha fechada, para observar a estabilidade e qualidade do controle na operação servo através da observação de parâmetros como a taxa de decaimento e o tempo de resposta. O sistema implementado pode ser visualizado na Figura 3.5.

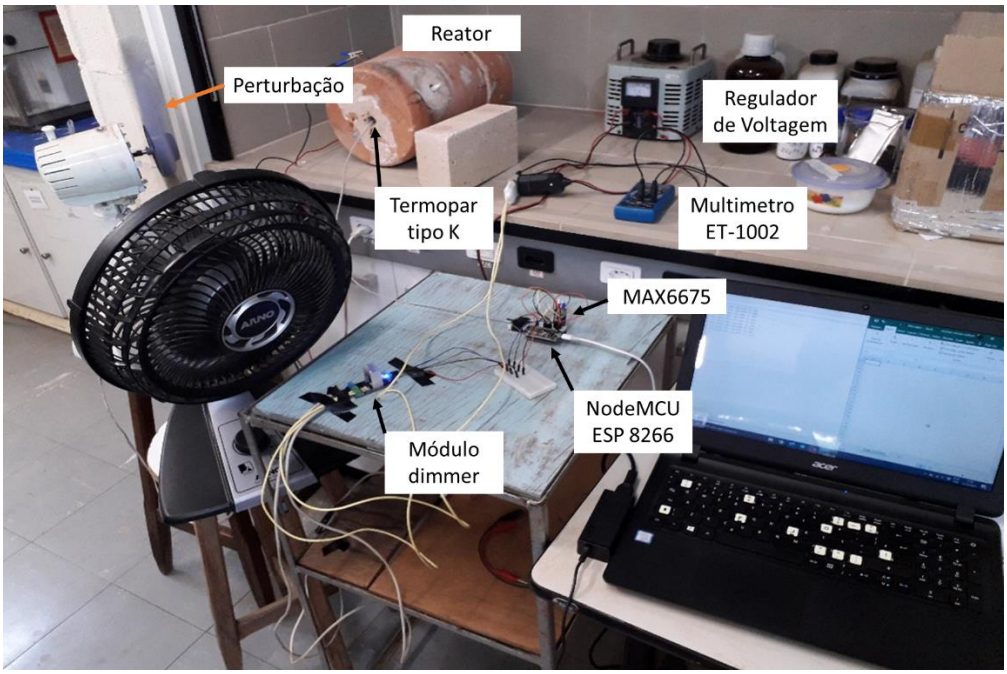
Figura 3.6. Sistema experimental de controle de temperatura do reator.



Fonte: Acervo Pessoal.

Por fim, uma última perturbação foi causada, desta vez na operação de carga, a perturbação ocorreu através da ativação da exaustão da capela, uma vez que efetivamente o reator operará em uma capela, em conjunto com um ventilador doméstico direcionado para o reator, para intensificar a perturbação. Deste modo foi possível avaliar se o controle seria efetivo em operação carga, verificando perturbações externas ao sistema em si e que são cotidianas do laboratório. O arranjo dos equipamentos neste experimento pode ser visualizado na Figura 3.6.

Figura 3.7. Sistema experimental de controle do reator com operação carga.



Fonte: Acervo Pessoal.

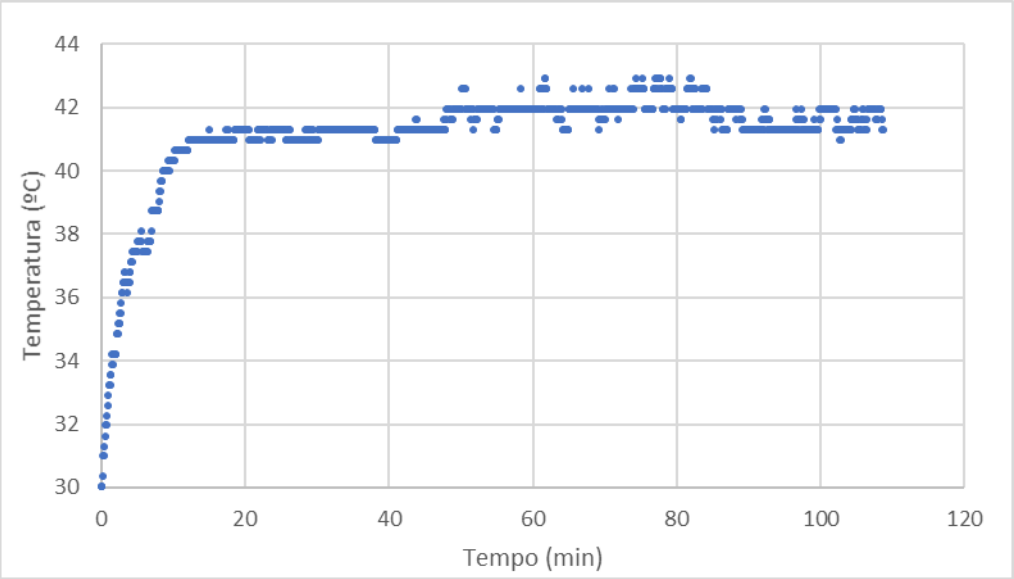
4. RESULTADOS E DISCUSSÃO

4.1. Experimentos preliminares

A validação do NodeMCU ESP8266 como microcontrolador foi bem sucedida. O controlador foi capaz de efetuar a mesma operação que o Arduino Uno R3 com a mudança consistindo basicamente em utilizar a biblioteca “Ticker” (STAUB et al., 2021) ao invés da “TimerOne” (STOFFREGEN et al., 2015).

Deste modo, o próximo passo foi avaliar a dinâmica do sensor frente ao calor fornecido pela lâmpada. Sendo assim o sensor foi posicionado em um primeiro momento em um ambiente aberto sem o isolamento com vidro, o que resultou em um comportamento cheio de ruídos e oscilações como pode ser visto na Figura 4.1.

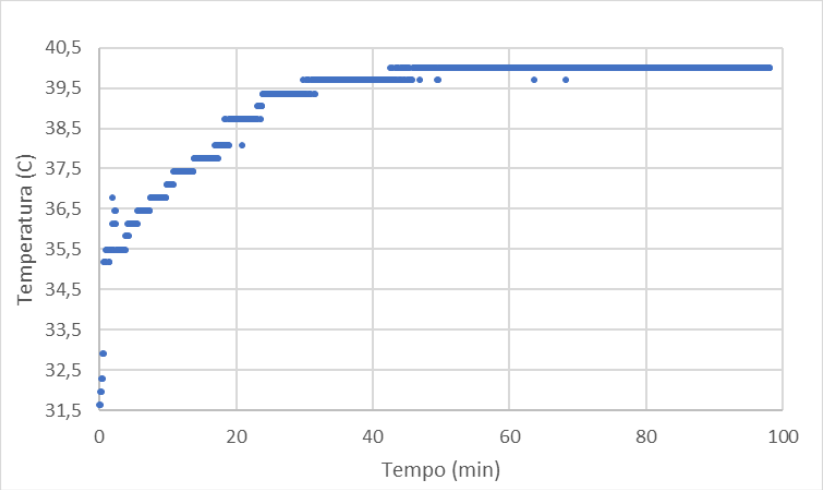
Figura 4.1. Degrau na potência da lâmpada com sensor em ambiente aberto.



Fonte: Acervo Pessoal.

Deste modo, foram feitos quatro testes de degrau no sistema com inibição da convecção, como pode ser observado nas figuras Figura 4.2, Figura 4.3, Figura 4.4 e Figura 4.5, afim de avaliar o comportamento do sistema mesmo com variações de temperatura ambiente entre os testes e durante o teste. O critério de parada dos testes foi baseado na percepção do operador de que o sistema havia chegado no estado estacionário.

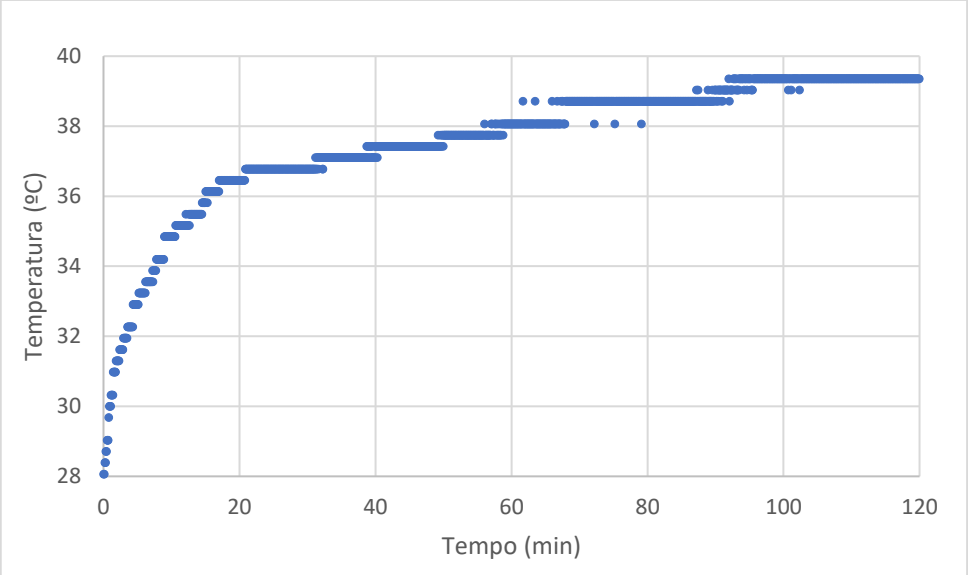
Figura 4.2. Primeiro teste de aplicação de degrau no sistema Lâmpada-LM35DZ após inibição da convecção.



Fonte: Acervo Pessoal

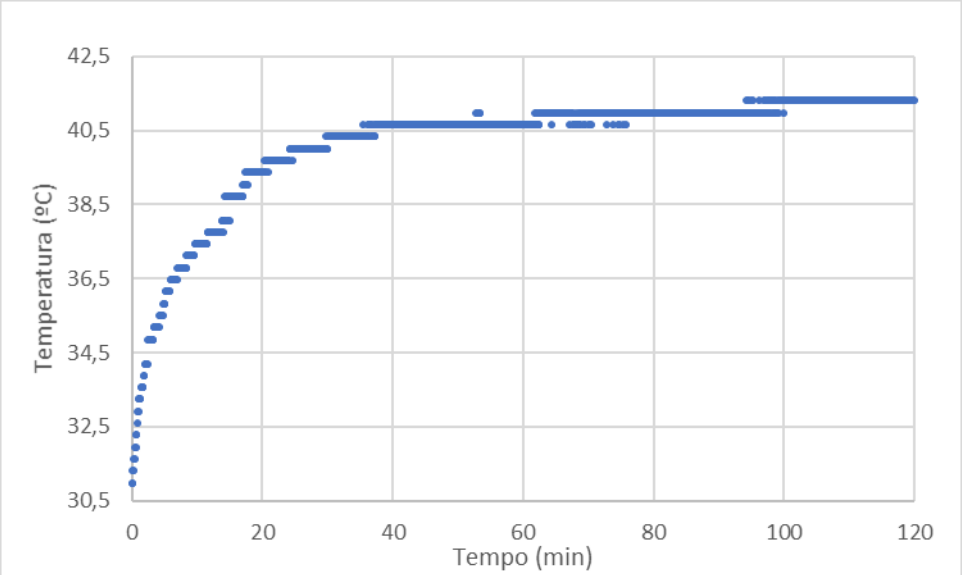
Na Figura 4.2, já podemos observar um claro comportamento de primeira ordem, entretanto houve alguns pontos que se destoaram do comportamento padrão logo no início do degrau. A hipótese é a de que isso foi causado por algum mau contato na conexão do sensor de temperatura, pois podemos observar abaixo, nas figuras Figura 4.3, Figura 4.4 e Figura 4.5, que estes desvios não se repetiram.

Figura 4.3. Segundo teste de aplicação de degrau no sistema Lâmpada-LM35DZ após inibição da convecção.



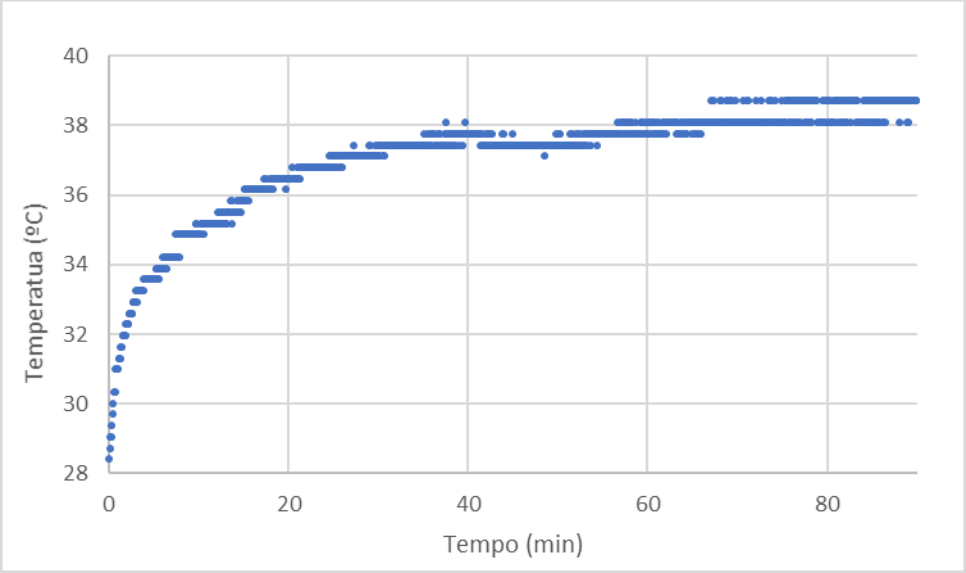
Fonte: Acervo Pessoal

Figura 4.4. Terceiro teste de aplicação de degrau no sistema Lâmpada-LM35DZ após inibição da convecção.



Fonte: Acervo Pessoal

Figura 4.5. Quarto teste de aplicação de degrau no sistema Lâmpada-LM35DZ após inibição da convecção.



Fonte: Acervo Pessoal

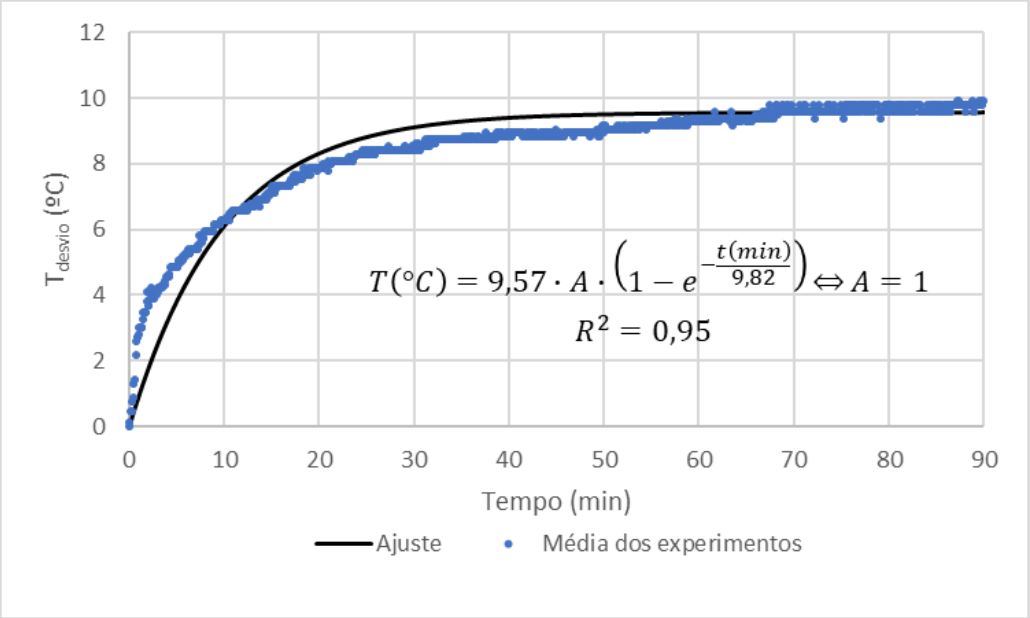
Uma vez que os experimentos foram bem caracterizados, foram estimados os parâmetros de sistema de primeira ordem para cada curva e então feito a média destes parâmetros, que podem ser encontrados na Tabela 4.1. Podemos ver na Figura 4.6 o resultado da aplicação deste ajuste, que compara a média dos pontos experimentais no decorrer do tempo com a curva de ajuste.

Tabela 4.1. Parâmetros estimados para processo de primeira ordem.

Parâmetro	1º Teste	2º Teste	3º Teste	4º Teste	Média ± desv. Pad.
K_P (ºC)	8,30	10,52	9,97	9,48	9,57 ± 0,94
τ_p (min)	9,09	11,94	9,92	8,33	9,82 ± 1,56

Fonte: Acervo Pessoal

Figura 4.6. Gráfico de comparação da média de pontos experimentais com o ajuste.

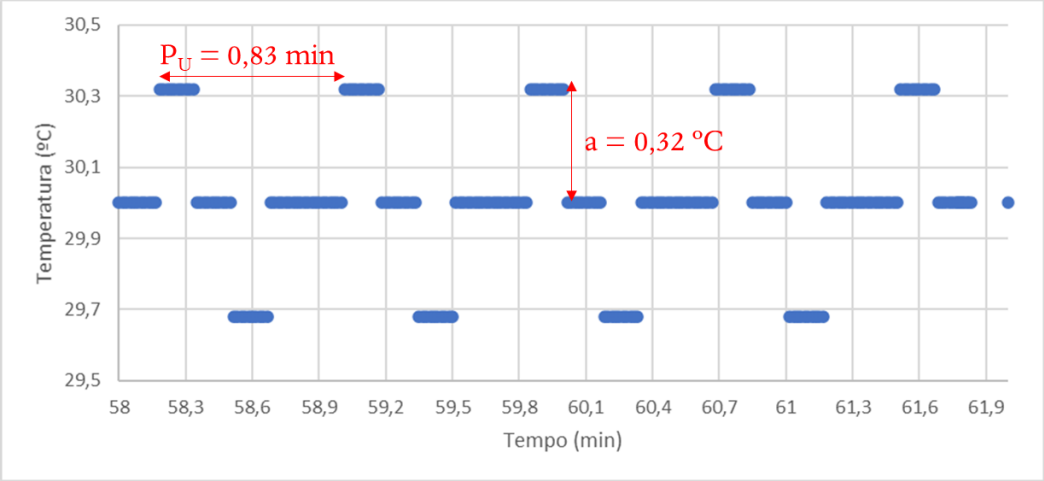


Fonte: Acervo Pessoal

Podemos concluir que o ajuste representa bem o sistema no geral, uma vez que a curva se aproximou muito bem da média dos pontos experimentais, além de que o coeficiente de determinação (R^2) do ajuste foi de 0,95, reforçando a qualidade do ajuste. Os pontos medidos após 90 minutos de experimento foram considerados no ajuste dos parâmetros, mas não na comparação com a média dos pontos experimentais explicitada na Figura 4.6, uma vez que o intervalo de tempo que contém dados de todos os testes foi apenas de 0 a 90 min.

Considerado a modelagem do processo concluída, aplicou-se o método do relé ao sistema para o projeto dos parâmetros de controle de realimentação PID. Deste modo, com o uso do módulo de *dimmer* realizou-se o procedimento até a oscilação se apresentar sustentada como evidenciado na Figura 4.7.

Figura 4.7. Recorte do comportamento do sistema lâmpada-LM35DZ durante o experimento do método do relé.



Fonte: Acervo Pessoal

A Figura 4.7 representa um recorte da região onde a oscilação causada durante o experimento do método do relé demonstrou maior estabilidade. Sendo assim, como demonstrado na imagem, o período final foi de 0,83 minutos enquanto a amplitude da oscilação foi de 0,32 ºC. Tendo em mente que a perturbação na variável de entrada foi de 50% (ou 0,50 em fração), foi calculado o ganho último pela equação 2.8 e os parâmetros do controle PID pelas equações 2.9, 2.10 e 2.11. Os resultados destes calculados são mostrados na Tabela 4.2.

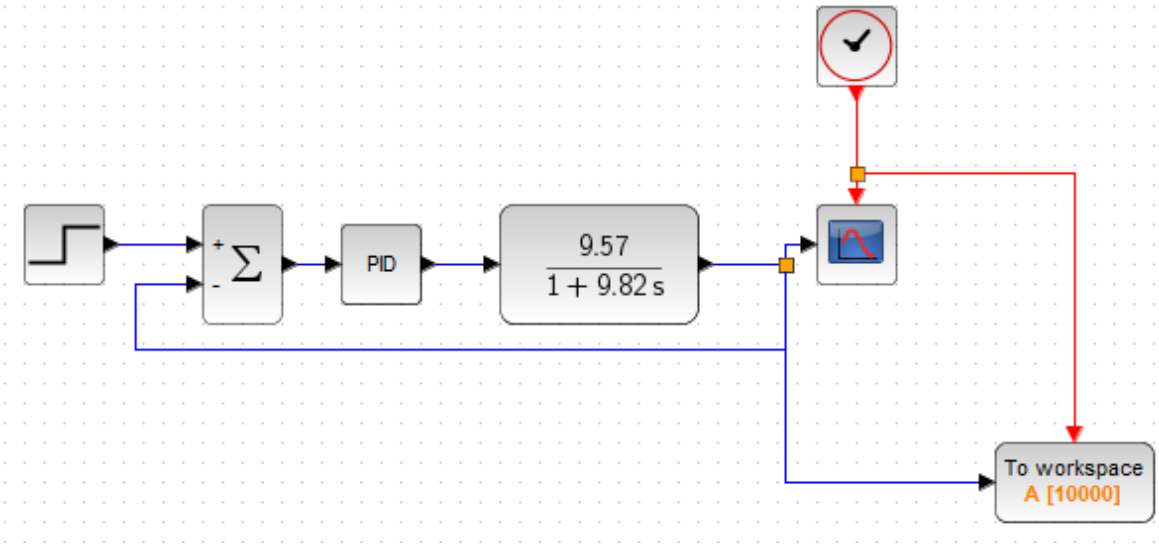
Tabela 4.2. Calculo do ganho último e dos parâmetros do controle PID do sistema Lâmpada-LM35DZ.

K_U ($ºC^{-1}$)	K_C ($ºC^{-1}$)	τ_I (min)	τ_D (min)
2,0	1,2	0,42	0,10

Fonte: Acervo Pessoal.

Em sequência, os parâmetros de projeto e do ajuste da curva foram inseridos no *software* Xcos para simulação do comportamento experimental. O digrama de blocos gerado pode ser observado na Figura 4.8.

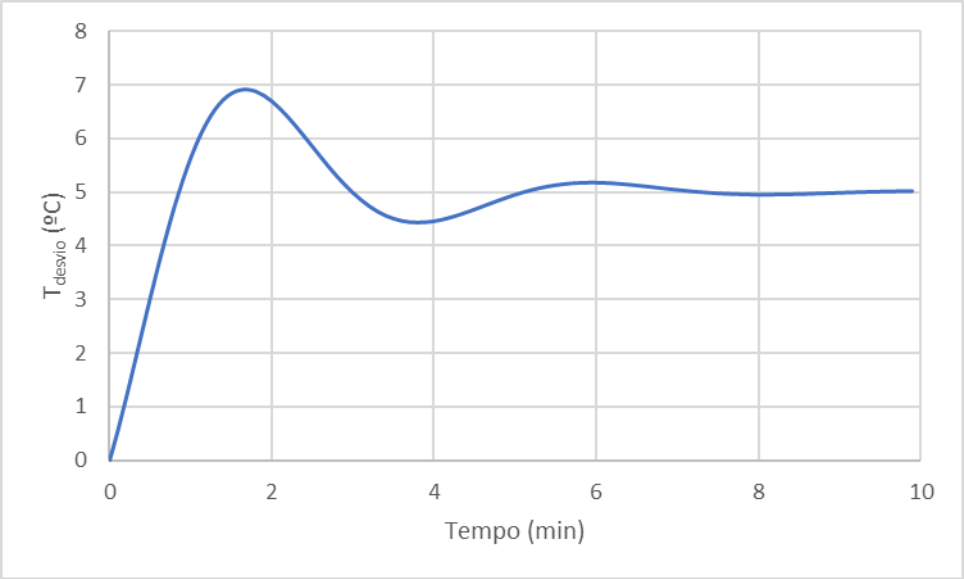
Figura 4.8. Diagrama de blocos da simulação do controle do sistema Lâmpada-LM35DZ no *software* Xcos.



Fonte: Acervo Pessoal.

O resultado da simulação pode ser visto na Figura 4.9. Podemos notar graficamente que o resultado simulado exibe uma variação de temperatura no decorrer do tempo de aproximadamente 5 ºC/min. Por outro lado, os testes de degrau de 100% (ou seja, potência máxima de entrada) mostraram um aumento máximo de aproximadamente 1 ºC/min. Portanto a aplicação deste controle exigiria um controle excessivo que superaria o limite do elemento final de controle, o chamado "*wind-up*" (SEBORG, 2017).

Figura 4.9. Resultado da simulação do controle PID pelo Xcos.



Fonte: Acervo Pessoal.

Por outro lado, SEBORG (2017) apresenta o parâmetro “a” do cálculo do ganho final no método do relé como a variação entre um pico e um vale, o que nos fornece um ganho final que é metade do valor calculado a partir do método em YU (2006) e ÅSTRÖM (1984). Apesar da divergência, essa discrepância permitiu a aplicação de um controle PID neste sistema. Uma vez que os objetivos destes primeiros experimentos foram testar o circuito, a programação dos controladores e o aprendizado, foi aplicado este novo parâmetro de ganho final para o experimento preliminar, resultado nos novos parâmetros evidenciados na Tabela 4.3.

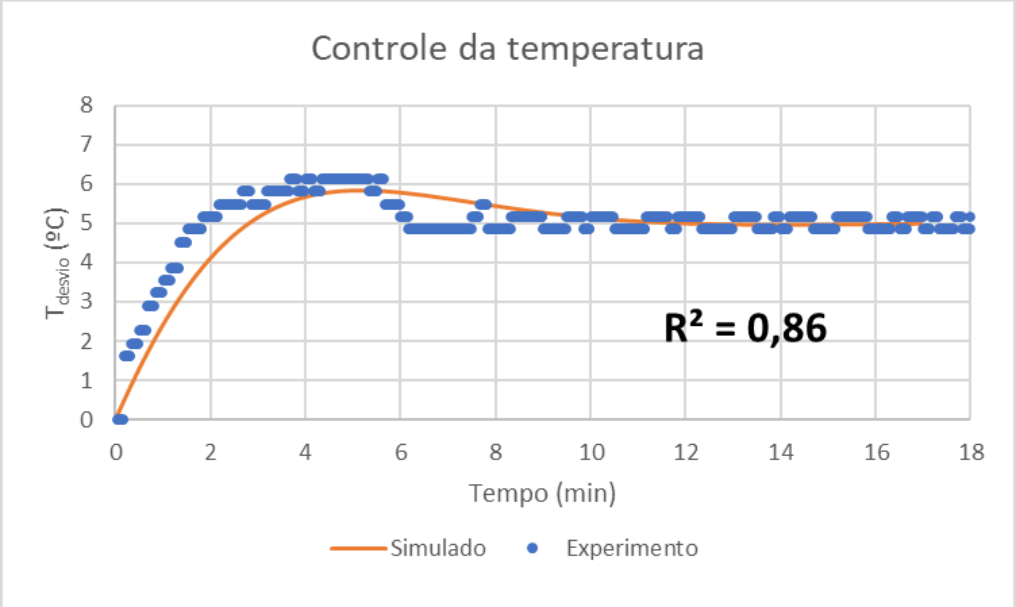
Tabela 4.3. Novos dados para a aplicação de controle PID no sistema Lâmpada-LM34DZ.

K_U (ºC^{-1})	K_C (ºC^{-1})	τ_I (min)	τ_D (min)
0,99	0,60	0,42	0,10

Fonte: Acervo Pessoal.

Com estes novos dados e usando o mesmo diagrama da Figura 4.8, foi possível simular um novo comportamento do sistema e também realizar o experimento de forma devida, sendo assim, a Figura 4.10 mostra o comportamento experimental e o comportamento simulado do controle PID com estes parâmetros.

Figura 4.10. Comparação entre simulação e dados experimentais para o controle do sistema Lâmpada-LM35DZ.



Fonte: Acervo Pessoal.

De forma qualitativa, podemos notar que o comportamento simulado e o experimental se aproximaram bem, afinal visualmente eles se acompanham, além do coeficiente de determinação ser de 0,86. Ainda temos que o maior desvio entre os dados do modelo e o teórico nesta avaliação foi de 1,1 ºC.

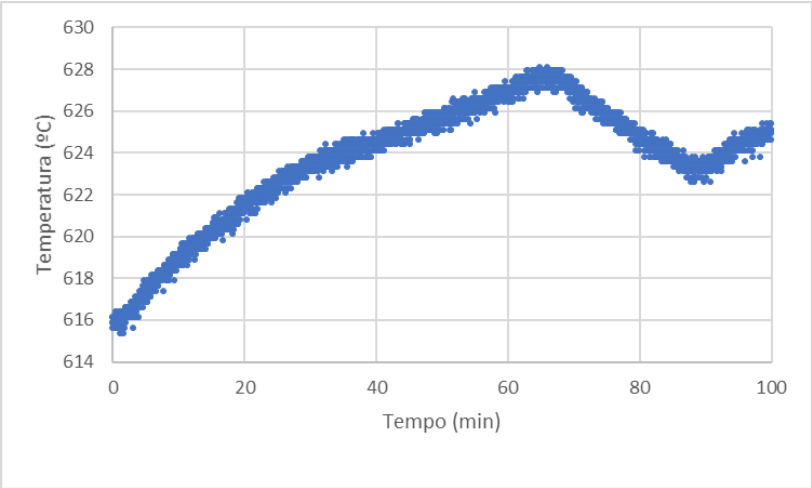
Isso indica que a modelagem ocorreu de forma coerente e ainda indica também que a programação do microcontrolador, assim como toda a metodologia utilizada, está realmente sendo efetiva.

Com esta experiência e os aprendizados das observações discutidas, tivemos segurança para avançar para a próxima etapa e lidar efetivamente com o reator.

4.2. Experimentos no reator

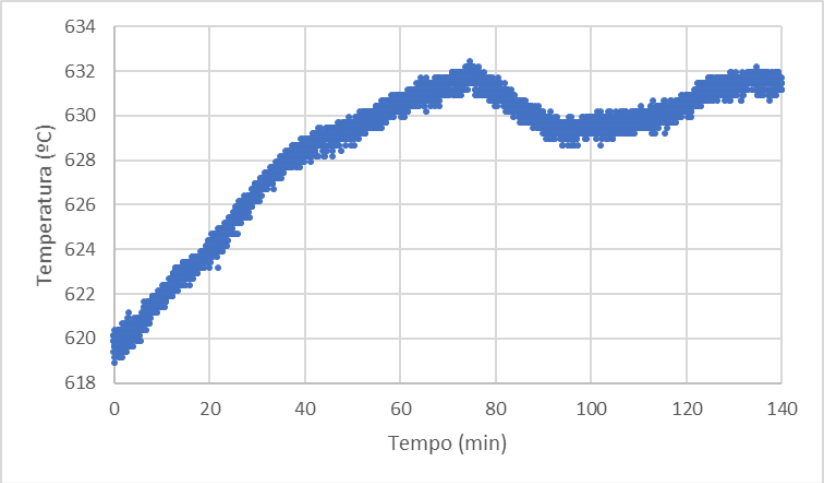
Ao trabalhar no reator, o experimento seguiu basicamente os mesmos passos do experimento preliminar. Primeiramente, verificou-se o comportamento do sistema através de perturbações degrau na variável de entrada como informado na seção de métodos. Os resultados podem ser observados nas Figuras 4.11 e 4.12.

Figura 4.11. Primeiro teste de aplicação de degrau no reator.



Fonte: Acervo Pessoal.

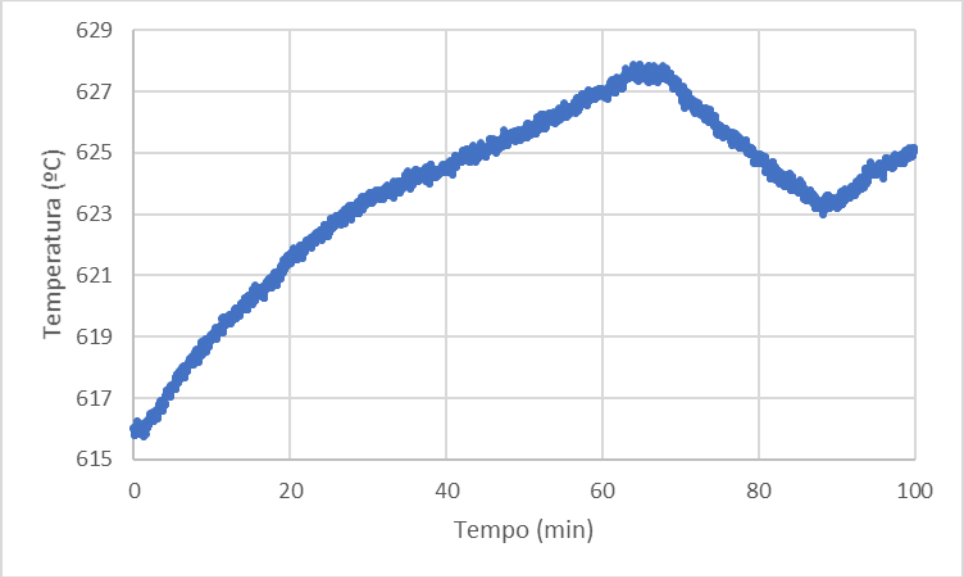
Figura 4.12. Segundo teste de aplicação de degrau no reator.



Fonte: Acervo Pessoal.

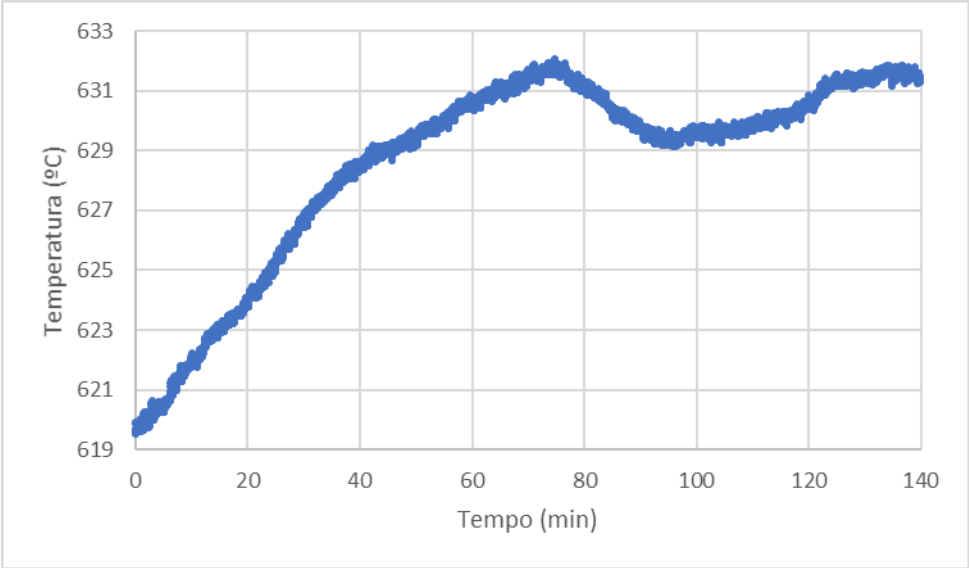
Nestes gráficos podemos observar alguns pontos de destaque dos experimentos: primeiramente, alcançar o *set point* de 600ºC como valor inicial para os degraus não foi uma tarefa trivial, devido a dificuldade de operar manualmente a potência fornecida ao reator. Resultando que os dois testes de análise de dinâmica foram efetuados em temperaturas acima do idealizado. Outro ponto notado é que após certo período o sistema teve uma queda de temperatura que posteriormente foi sendo recuperada, foi notado durante os experimentos uma grande variação na tensão de alimentação do sistema, provavelmente devido a interferências de outros equipamentos ligados na rede elétrica do departamento. Por fim, verifica-se que a medida do termopar possui um ruido da ordem de 2°C. Por esse motivo utilizou-se um filtro de média móvel considerando 5 pontos para minimizar os efeitos dos ruídos no ajuste da curva, como pode ser visto na Figura 4.13 e Figura 4.14.

Figura 4.13. Primeiro teste de aplicação de degrau no reato com média móvel de cinco pontos.



Fonte: Acervo Pessoal.

Figura 4.14. Segundo teste de aplicação de degrau no reator com média móvel de cinco pontos.

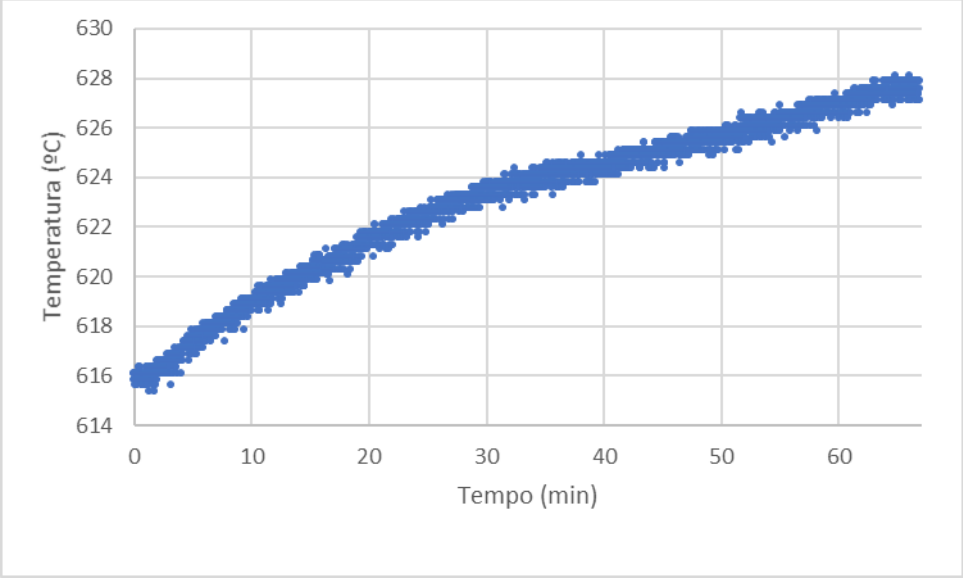


Fonte: Acervo Pessoal.

Com os gráficos da média móvel, deduziu-se que o sistema também poderia ser descrito de forma aproximada de um sistema de primeira ordem sem tempo morto. Realmente foi notada uma leve inércia inicial para o aumento da temperatura do sistema, o que é esperado em um sistema como esse, em que a variação de temperatura depende de fenômenos condutivos de transferência de calor. Porém, esse atraso é relativamente pequeno quando comparada com a dinâmica do sistema.

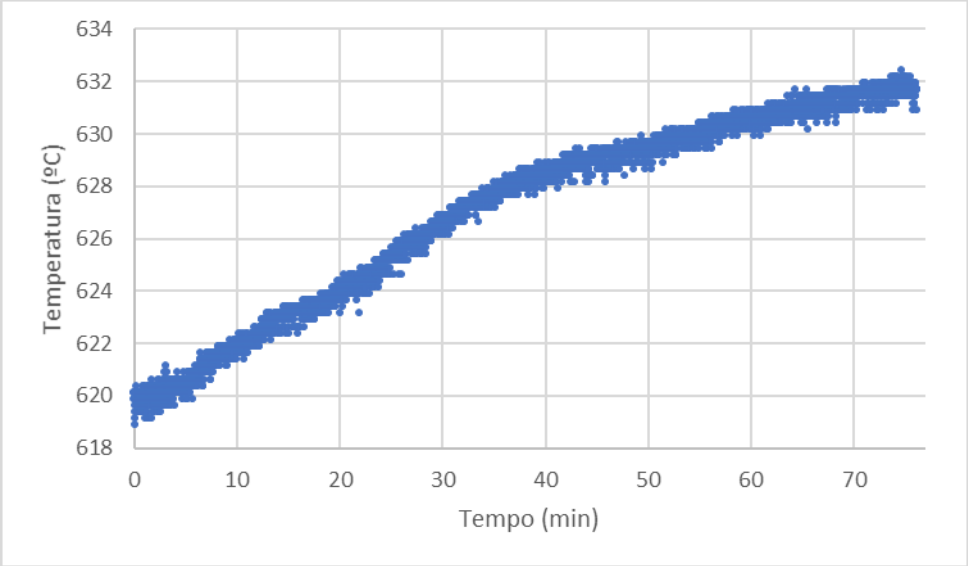
Sendo assim, da Figura 4.11 (primeira curva mostrada) foram extraídos os pontos até 66,87 minutos como podemos ver na Figura 4.15, logo antes da queda de temperatura ocorrer e feito um ajuste de uma curva de primeira ordem no *software* Origin. Da Figura 4.12 (segunda curva mostrada) foram feitos dois ajustes, um com todos os pontos medidos, os quais estão presentes na Figura 4.16 e um apenas dos pontos até 76,22 minutos e depois de 127,62 minutos a 141,25 minutos como podemos ver na Figura 4.17. Esse segundo conjunto de pontos foi selecionado visualmente e utilizado, pois, indicam o novo estado estacionário do sistema após a perturbação.

Figura 4.15. Pontos extraídos para o primeiro ajuste de parâmetros do degrau no reator.



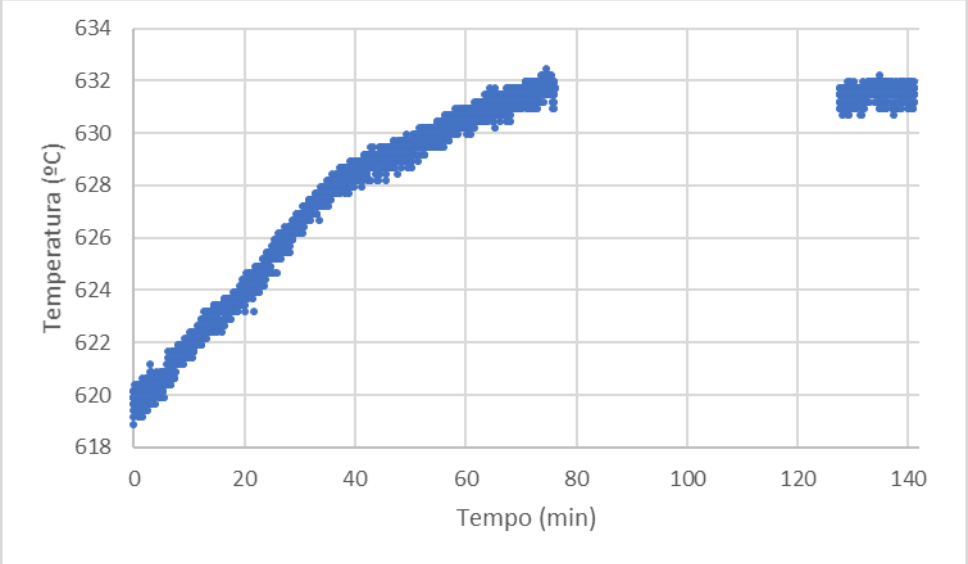
Fonte: Acervo Pessoal.

Figura 4.16. Pontos extraídos para o segundo ajuste de parâmetros do degrau no reator.



Fonte: Acervo Pessoal.

Figura 4.17. Pontos extraídos para o terceiro ajuste de parâmetros do degrau no reator.



Fonte: Acervo Pessoal.

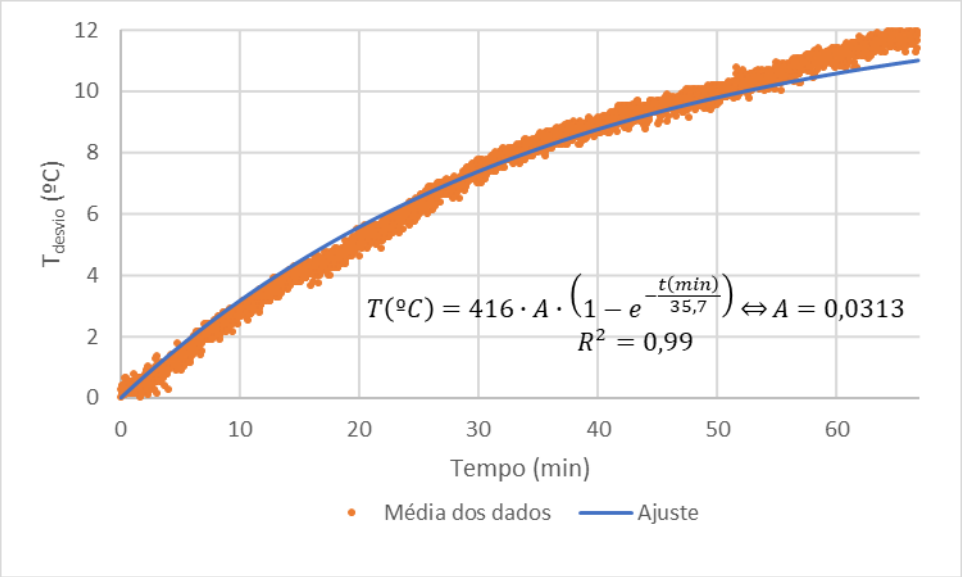
Os resultados destes ajustes assim como a média dos parâmetros são explicitados na Tabela 4.4. Logo após, podemos ver a comparação entre a curva de ajuste e a média dos dados experimentais na Figura 4.18, tendo sido comparado apenas as médias até 66,87 min, pois após este tempo, não há pontos da primeira curva que foram ajustados.

Tabela 4.4. Parâmetros estimados para processo de primeira ordem para curva do reator.

	1º Ajuste	2º Ajuste	3º Ajuste	Média ± Dev. Pad.
K_P (ºC)	459	373	416	416 ± 43
τ_p (min)	40,1	30,4	36,6	35,7 ± 4,9

Fonte: Acervo pessoal.

Figura 4.18. Comparação do ajuste das curvas de degrau no reator com a média dos pontos experimentais.



Fonte: Acervo Pessoal.

Uma vez que o comportamento do sistema foi caracterizado a partir dos pontos escolhidos, o próximo passo foi aplicar o método do relé no sistema. Na metodologia havia sido informado que, considerando a resistência utilizada no sistema, foi escolhida uma tensão que permitisse uma potência máxima por volta de 260 W. Nessa situação, a temperatura de 600°C foi alcançada com o módulo dimmer trabalhando com 85% da potência. Isso resulta em 221 W para o reator, sendo que em experimentos de aplicação de degrau, a potência base era de cerca de 188 W. O que ajuda a justificar essa discrepância é que foi observada uma queda de tensão de aproximadamente 10% entre a alimentação do dimmer e sua saída causada pelas próprias impedâncias do módulo quando este permite a passagem completa da corrente. Dessa forma, a potência máxima fornecida pelo dimmer (trabalhando com 100% da onda) foi de 233 W. Com isso, estabelecendo 85% de potência fornecida, idealmente o reator seria alimentado com 198 W, o que ainda resultaria em uma temperatura acima da inicial nos degraus o que não ocorreu.

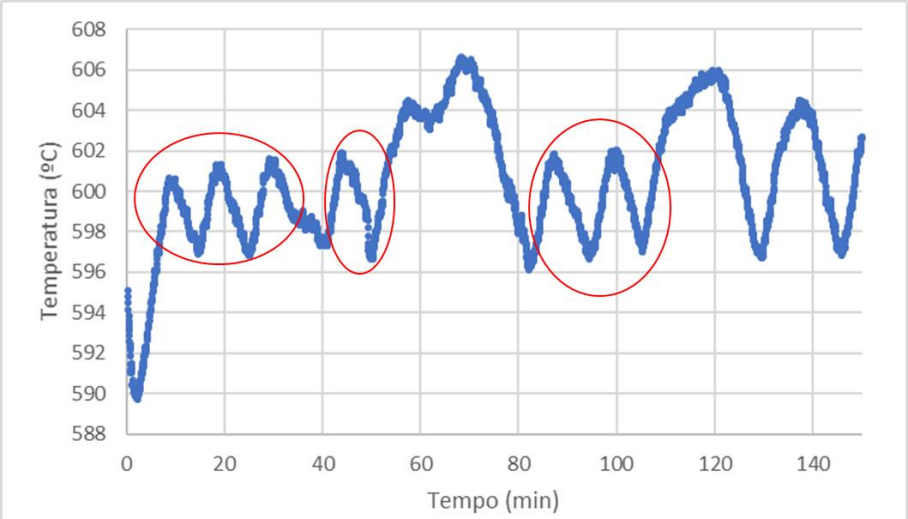
Uma possível explicação para a diferença observada é que a resistência responsável pelo aquecimento do reator sofre drásticas alterações ôhmicas, pois à temperatura ambiente ela possui um valor de 8,8 Ω e na temperatura de operação essa resistência sobe para aproximadamente 13 Ω.

Outra possibilidade para explicar este fenômeno é que isto tenha ocorrido devido a uma parte da tensão que é dissipada no triac do módulo de dimmer durante o processo de dimerização em si. Tal dissipação de tensão observada entre a entrada e saída do módulo dimmer foi responsável por um aquecimento indesejado neste, que poderia danificar alguns de seus componentes. Assim, a temperatura desta placa foi supervisionada. Experimentalmente utilizou-se um arrefecimento convectivo por fluxo de ar a temperatura ambiente, garantindo a faixa de operação dos componentes dentro de suas especificações recomendadas.

Uma vez que o sistema apresenta bastante ruído, foi aplicado para o método do relé uma amostragem baseada na média móvel das últimas cinco medidas de temperatura e aplicado uma histerese na variável de entrada, que consistia em ao invés da inversão da perturbação de entrada ocorrer quando a temperatura cruzava o *set point*, ela ocorria quando a temperatura cruzava o *set point* acrescido de 1ºC, no caso de a perturbação estar positiva, ou decrescido de 1ºC, no caso de a perturbação estar negativa. O resultado do teste do relé é

apresentado na Figura 4.19. Como pode ser observado, houveram diversas distorções no comportamento do sistema destoando do ideal. Desta forma, os parâmetros do método do relé foram calculados utilizando somente as zonas destacadas em vermelho.

Figura 4.19. Experimento do relé no reator.



Fonte: Acervo Pessoal.

Durante o experimento foram notados picos de tensão na rede elétrica do departamento onde ocorria os experimentos, o Departamento de Química, que alimentava o reator, próximo de 50 minutos e de 110 minutos, o que se discute que seja a razão da oscilação se distorcer tanto nessas regiões. Com isso, foram extraídos manualmente os valores de temperatura e tempo de experimento dos picos e vales presentes nessas regiões e calculados os parâmetros do método do relé e de controle, como pode ser visto nas tabelas Tabela 4.5, Tabela 4.6 e Tabela 4.7. É possível observar que apesar dos problemas encontrados, os valores calculados apresentaram boa reprodutibilidade, com desvio padrão inferior a 1% para as amplitudes e 4% para o período de oscilação.

Tabela 4.5. Valores de temperatura dos picos e vales e amplitude final calculada.

	1º Medida (ºC)	2ª Medida (ºC)	3ª Medida (ºC)	4ª Medida (ºC)	5ª Medida (ºC)	6ª Medida (ºC)	Média ± Desv. Pad. (ºC)	Amplitude (ºC)
Pico	600,7	601,3	601,6	601,9	601,9	602,05	601,6 ± 0,5	4,7
Vale	596,9	596,8	596,6	596,7	587	-	596,8 ±0,2	

Fonte: Acervo Pessoal.

Tabela 4.6. Valores de período final extraídos dos dados experimentais e média.

1º Medida (min)	2ª Medida (min)	3ª Medida (min)	4ª Medida (min)	Média ± Desv. Pad. (min)
10,8	10,1	10,3	11	10,5 ± 0,4

Fonte: Acervo Pessoal.

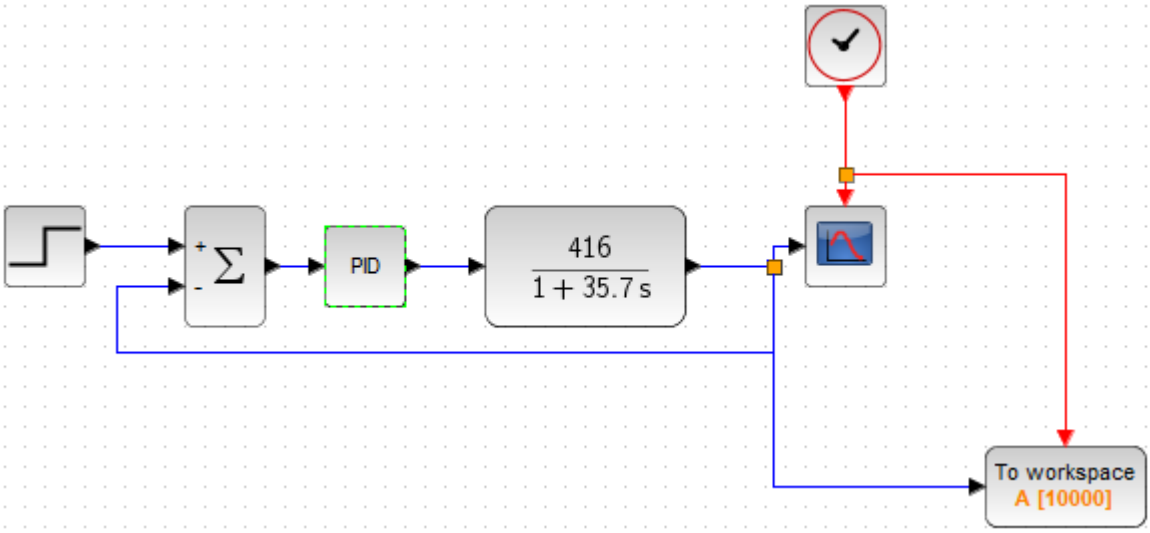
Tabela 4.7. Calculo do ganho final e dos parâmetros de controle.

K_U ($^oC^{-1}$)	K_C ($^oC^{-1}$)	τ_I (min)	τ_D (min)
0,027	0,016	5,3	1,3

Fonte: Acervo Pessoal.

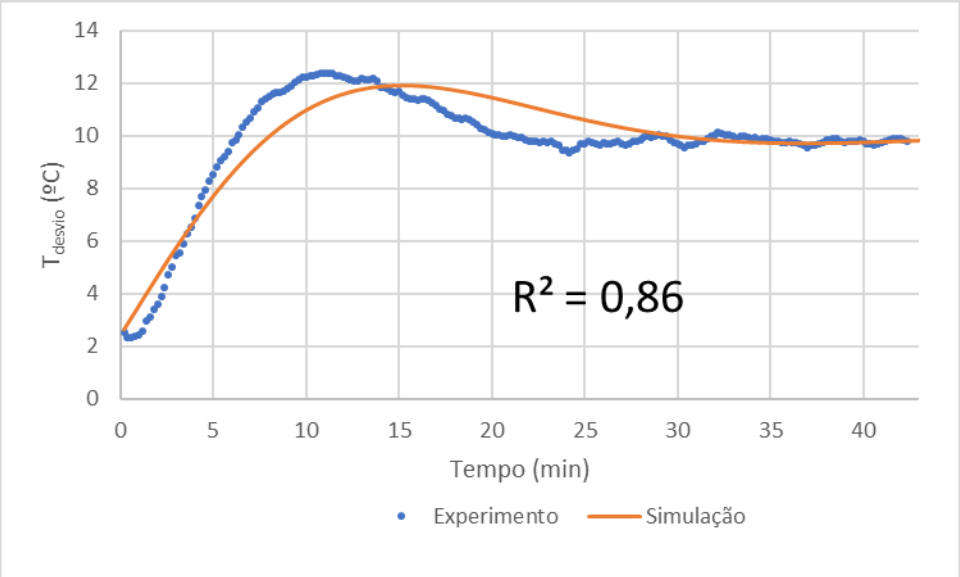
Com isto foi possível simular o controle no software Xcos através do diagrama evidenciado na Figura 4.20. Após isso, foi comparado o comportamento simulado com o experimental para avaliação da qualidade do controle, cujo resultado é mostrado na Figura 4.21.

Figura 4.20. Diagrama da simulação do sistema de controle do reator no Xcos.



Fonte: Acervo Pessoal.

Figura 4.21. Gráfico de comparação do resultado experimental com o simulado.

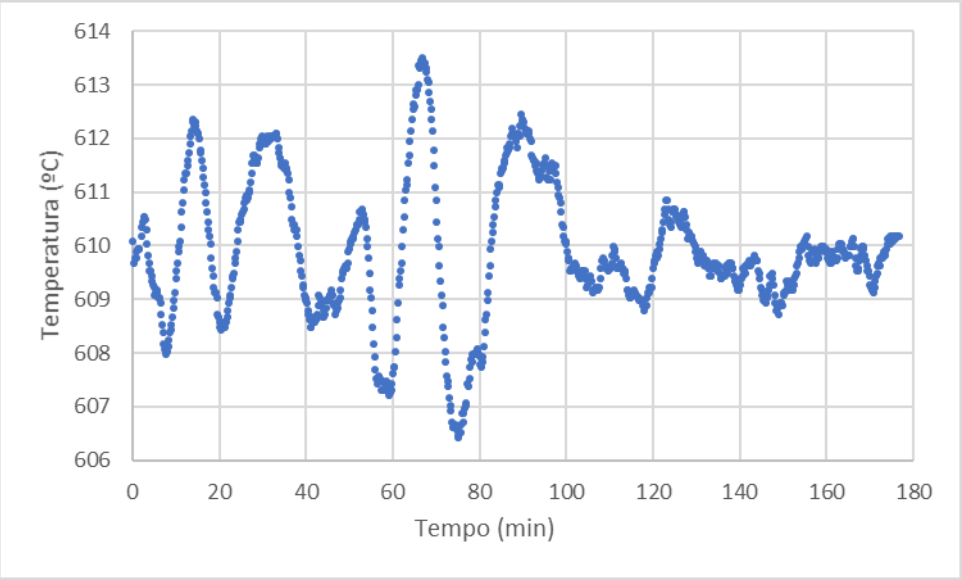


Fonte: Acervo Pessoal

Podemos notar que para a operação servo o controle se ajustou muito bem ao modelo, com um coeficiente de determinação de 0,86 e um desvio máximo de 1,6 ºC entre os dados teórico e o experimental.Além de ter sido aplicado com alta eficácia, havendo apenas um pico acima do *set point* e mais nenhuma oscilação considerável antes do sistema atingir o estado estacionário. Com isso, também podemos dizer que a metodologia aplicada foi efetiva no projeto do controle do reator.

O último teste consistiu em avaliar a capacidade do sistema de controlar a operação carga, que uma vez aplicada, mostrou o comportamento visualizado na Figura 4.21.

Figura 4.21. Dinâmica da aplicação de perturbação carga no sistema.



Fonte: Acervo Pessoal.

Neste teste pudemos notar que o sistema é robusto o suficiente para corrigir mesmo perturbações externas. Entretanto, o sistema demorou aproximadamente duas horas para que o desvio da temperatura estivesse entre +/- 1°C do *set point*, além de haver sérias variações durante o processo. Como já mencionado, a rede elétrica do departamento apresentou várias oscilações, uma vez que o ventilador estava ligado na mesma rede elétrica que o reator e que foram notadas diversas variações de tensão medidas pelo multímetro durante o experimento. Tais variações possivelmente são causadas devido a outros equipamentos usados no departamento que podem gerar interferências na rede elétrica. Supõe-se que boa parte desta demora e das oscilações mais intensas ocorreram devido a essas oscilações na tensão da rede elétrica, que inevitavelmente afetará a potência de aquecimento do reator e a intensidade de giro do ventilador.

5. CONCLUSÃO E SUGESTÕES

5.1. Conclusão

Foi possível implementar um controle PID em loop fechado de realimentação utilizando hardware e software livres num reator cerâmico desenvolvido para a produção laboratorial de cloreto de alumínio anidro, que era controlado manualmente. O projeto de controle satisfaz as necessidades de correção na operação servo e operação carga, tendo um ótimo desempenho na operação servo. Já a operação carga apresentou diversas oscilações, retornando ao set point somente após duas horas. A grande maioria dos problemas ocorridos até então foram devidos a variações na tensão alimentada, o que é um ponto delicado uma vez que a falta de estabilidade na alimentação de eletricidade pode debilitar a estabilidade do sistema como um todo.

5.2. Sugestões

Sugere-se uma avaliação do controle enquanto ocorre a reação em si, dado que a mesma é exotérmica, para futura avaliação, além de aplicar um controle da alimentação de reagente em conjunto. Uma possível solução para o problema da instabilidade da rede elétrica é a aplicação de um controle em cascata para ajustar o controle da potência de aquecimento da resistência de forma que o sistema não oscile tanto.

Em outro aspecto, como o NodeMCU ESP-8266, o controlador utilizado nos diversos testes, possui capacidade de integração de Internet das Coisas via Wi-Fi, uma melhoria interessante é aplicar esta possível integração para monitoramento e controle do sistema como um todo remotamente.

REFERÊNCIAS BIBLIOGRÁFICAS

ARDUINO. Site principal de comunidade de Arduino. Disponível em: <https://www.arduino.cc/>. Acesso em: 21 out. 2021.

ARDUINO Trademark & Copyright. Disponível em: <https://www.arduino.cc/en/trademark>. Acesso em: 17 out. 2021.

ÅSTRÖM, K. J.; HÄGGLUND, T. Automatic tuning of simple regulators with specifications on phase and amplitude margins. Automatica, v. 20, n. 5, p. 645–651, 1 set. 1984.

ELBS, Karl. Die synthetischen Darstellungsmethoden der Kohlenstoff-Verbindungen. 1 ed. Leipzig, Alemanha: J. A. Barth, 1891. Volume 2.

ESP8266. ESP8266 Community Forum. Disponível em: < https://www.esp8266.com/>; Acesso em: 28 out. 2021

KIRK, R. E. Encyclopedia of Chemical Technology: Alkanolamines to Antibiotics. 4ed. Hoboken, EUA: Wiley-Blackwell, 1992. Volume 2.

LUYBEN, W. L. Derivation of transfer functions for highly nonlinear distillation columns. Industrial and Engineering Chemistry Research, v. 26, n. 12, p. 2490–2495, 1 dez. 2002.

MANUAL DO MUNDO. O que é Arduino, afinal de contas? #ManualMaker Aula 4, Vídeo 1. Disponível em: <https://www.youtube.com/watch?v=sv9dDtYnE1g>. Acesso em: 17 out. 2021.

MCAFEE, A. M. The Manufacture of Commercial Anhydrous Aluminum Chloride. Industrial & Engineering Chemistry, v. 21, n. 7, p. 670–673, 1 jul. 1929.

MULTILÓGICA-SHOP. Arduino Guia Iniciante. Disponível em: <https://multilogica-shop.com/download_guia_arduino>. Acesso em: 19 nov. 2021.

ZAIT, A. NodeMCU - A Perfect Board for IoT. Circuito.io. 2018. Disponível em: <https://www.circuito.io/blog/nodemcu-esp8266/>. Acesso em: 10 out. 2021.

REDDIT. ESP8266. Disponível em: < https://www.reddit.com/r/esp8266/>. Acesso em: 28 out. 2021

SEBORG, D. E. et al. Process Dynamics and Control. 4 ed. Hoboken: John Wiley & Sons, Inc., 2017. 515p.

STAUB, S. et al. Ticker. 2021. Plataforma de hospedagem de código-fonte e arquivos usando o Git. Disponível em: <https://github.com/sstaub/Ticker>. Acesso em: 24 out. 2021.

STEPHANOPOULOS, G. Chemical Process Control: An Introduction to Theory and Practice. 1 Ed. Englewood Cliffs, USA: Prentice Hall, 1984.

STOFFREGEN, P. et al. TimerOne. 2015. Plataforma de hospedagem de código-fonte e arquivos usando o Git. Disponível em: <https://github.com/PaulStoffregen/TimerOne>. Acesso em: 24 out. 2021.

YU, C. C. Autotuning of PID controllers: A Relay Feedback Approach. 2 Ed. Germany: Springer-Verlag London, 2006.

YUAN, M. Getting to know NodeMCU and its DEVKIT board. Developer. 07 ago. 2017. Disponível em: < https://developer.ibm.com/tutorials/iot-nodemcu-open-why-use/>. Acesso em: 28 out. 2021

ZIEGLER, J. G.; NICHOLS, N. Optimum Settings for Automatic Controllers. Journal of Dynamic Systems Measurement and Control-transactions of The Asme, v. 115, p. 220–222, 1942.

BIBLIOGRAFIA

ARDUINO Store. Disponível em: <https://store-usa.arduino.cc/collections/most-popular/products/arduino-uno-rev3>. Acesso em: 17 out. 2021.

BENCHOFF, B. *A DEV BOARD FOR THE ESP LUA INTERPRETER*. HACKADAY. 01 jan. 2015. Disponível em: <https://hackaday.com/2015/01/01/a-dev-board-for-the-esp-lua-interpreter/>. Acesso em: 24 out. 2021.

ELECROW. Disponível em: <https://www.elecrow.com/nodemcu-v2-esp8266-development-board-p-1583.html>. Acesso em: 24 out. 2021.

ESP8266*: A cost-effective and highly integrated Wi-Fi MCU for IoT applications*. In: ESPRESSIF site. Página da Web da Espressif sobre o chiop ESP8266. Disponível em: <https://www.espressif.com/en/products/socs/esp8266>. Acesso em: 24 out. 2021.

ESP8266 Shop. Disponível em: <https://esp8266-shop.com/product/nodemcu-esp8266-esp-12e/>. Acesso em: 19 nov. 2021.

ESPRESSIF. Empresa produtora de módulos e chips focados em IoT. Disponível em: <http://www.espressif.com/>. Acesso em: 24 out. 2021.

ESPRESSIF. ESP8266EX: Datasheet. Versão 6.6: Espressif Systems. 2020. Disponível em: < https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf>. Acesso em: 14 out. 2021.

JENKINS, S. *Aluminum Chloride Production. Chemical Engineering*, Nova Iorque: Intratec Solutions, 2016.

PAUL FLOWERS; KLAUS THOPOLD; RICHARD LANGLEY. *5.3: Enthalpy - Chemistry LibreTexts*. Disponível em: <https://chem.libretexts.org/Bookshelves/General_Chemistry/Chemistry_(OpenSTAX)/05%3A_Thermochemistry/5.3%3A_Enthalpy>. Acesso em: 24 out. 2021.

RUMBLE. J. et al. *CRC HANDBOOK OF CHEMISTRY AND PHYSICS*. 100 ed. Boca Raton, EUA: CRC Press, 2019, 1554p.

STÖR, M. et al. nodemcu-firmware. 3.0.0. 2021. Plataforma de hospedagem de código-fonte e arquivos usando o Git. Disponível em: <https://github.com/nodemcu/nodemcu-firmware>. Acesso em: 24 out. 2021.

TWOMEY, R. *AC-Dimmer-Timer Driven*. Ambiente de desenvolvimento integrado em nuvem de plataforma cruzada. [201-]. Disponível em: <https://codebender.cc/sketch:171819#AC-Dimmer-Timer%20Driven.ino>. Acesso em: 24 out. 2021.

WIGUNA, H. NodeMCU LUA *Firmware*. HACKADAY.IO. 29 nov. 2014. Disponível em: <https://hackaday.io/project/3465-playing-with-esp8266/log/11449-nodemcu-lua-firmware>. Acesso em: 24 out. 2021.

APÊNDICE A – Código de leitura da temperatura pelo LM235DZ

```
void setup() {
  // put your setup code here, to run once:
  pinMode(A0, INPUT);
  pinMode(D5, OUTPUT);
  Serial.begin(115200);
}

unsigned long tempo[2];

void loop() {
  int leitura = analogRead(A0);
    tempo[0] = millis();
  float temperatura = leitura/3.1;
  if ((tempo[0] - tempo[1]) >= 5000) {
    Serial.println(leitura);
    Serial.println(String(temperatura)+"ºC; "+String(tempo[0])+"ms");
    tempo[1] = tempo[0];
  }
    // put your main code here, to run repeatedly:
}
```

APÊNDICE B – Código de leitura de temperatura pelo termopar tipo K

```
#include <max6675.h>

/* Definições: GPIOs do Arduino utilizado na comunicação com o
   MAX6675 */
#define GPIO_SO       D5
#define GPIO_CS       D6
#define GPIO_CLK      D7

/* Definição: baudrate da comunicação com Serial Monitor */
#define BAUDRATE_SERIAL_MONITOR 9600

/* Definição: tempo entre leituras do MAX6675 */
#define TEMPO_ENTRE_LEITURAS            1000 //ms

/* Criação de objeto para comunicação com termopar */
MAX6675 termopar(GPIO_CLK, GPIO_CS, GPIO_SO);

void setup()
{
   Serial.begin(BAUDRATE_SERIAL_MONITOR);
}
```

```
/* Programa principal */
void loop(){
    int tempo = millis()/1000;
    Serial.print("t: ");
    Serial.print(tempo);
    Serial.print("s; ");
    Serial.print("Temperatura: ");
    float T = 1.004233129*termopar.readCelsius()-1.743315757;
    Serial.println(T);

    delay(TEMPO_ENTRE_LEITURAS);
}
```

APÊNDICE C – Código de contagem de cruzamentos de zero na corrente de rede alternada.

```
volatile int i = 0;

unsigned long current;

unsigned long old = 0;

unsigned long delta = 1000;


ICACHE_RAM_ATTR void zero_cross_detect() {

i++;

}


void setup() {

  pinMode(D1, INPUT);

  attachInterrupt(D1, zero_cross_detect, RISING);

  Serial.begin(9600);

}


void loop() {

  current = millis();

  if(current - old > delta) {

    Serial.println("interrupções: " + String(i));

    i = 0;

    old = current;
```

```
    }
}
```

APÊNDICE D – Código de definição manual de potência do reator e leitura da temperatura pelo termopar tipo K.

```
#include <Ticker.h>

#include <max6675.h>


//função que aproxima linearmente a potência da rede elétrica de corrente alternada

int potencia(float pot) {

  int resp;

  if (pot <= 28.968) { //9

    if (pot <= 4.049) { //5

      if (pot <= 1.248) { //3

        if (pot <= 0.413) { //2

          resp = 26.634*pot;

          }

        else { //2

          resp = 5.988*pot+85269;

          }

        }

        else { //3

          if (pot <= 2.394) { //4

            resp = 3.4904*pot+11.644;

            }

          else { //4
```

```
      resp = 2.4169*pot+14.214;
      }
    }
  }
  else { //5
    if (pot <= 12.518) { //7
      if (pot <= 6.268) { // 6
        resp = 1.8026*pot+16.701;
        }
      else { //6
        resp = 1.28*pot+19.997;
        }
      }
    else { //7
      if (pot <= 19.945) { //8
        resp = 0.9425*pot+24.202;
        }
      else { //8
        resp = 0.7758*pot+27.527;
        }
      }
    }
  }
else  { //9
```

```
if (pot <= 90.912) { //13
  if (pot <= 75.071) { //11
    if (pot <= 63.835) { //10
      resp = 0.6596*pot+30.891;
      }
    else { //10
      resp = 0.712*pot+27.55;
      }
    }
  else { //11
    if (pot <= 84.513) { //12
      resp = 0.8473*pot+17.394;
      }
    else { // 12
      resp = 1.0939*pot-3.4505;
      }
    }
  }
else { //13
  if (pot <= 98.238) { //15
    if (pot <= 95.451) { //14
      resp = 1.5422*pot-44.204;
      }
    else { //14
```

```
            resp = 2.5117*pot-136.74;
            }
          }
        else { //15
          if (pot <= 99.688) { //16
            resp = 5.5172*pot-432;
            }
          else { //16
            resp = 32.051*pot-3077.1;
            }
          }
        }
       }
    return resp;
}

//Definição de variáveis diversas
volatile int ativador = D0;    // saida para o pino do Opto Triac
int pot=85;  //potencia em porcentagem
volatile int dim = 0;  // nível de dimerização (0-128)  0 = ligado, 128 = desligado
volatile boolean zero_cross = 0; //estado do cruzamento de zero
volatile int y = 0; //contador da intensidade da lampada
float atualizacao = 1e6/(120*128); //unidade: us
float T[5] = {0, 0, 0, 0, 0}; //unidade: ºC (temperatura para filtro)
```

```
float Tmed; ////unidade: ºC (média das ultimas 3 temperaturas)
float tempo; //unidade: s
float tempo_ant = 0; // unidade: s
int i = 0;

/* Definições: GPIOs do ESP utilizado na comunicação com o
   MAX6675 */
#define GPIO_SO       D5
#define GPIO_CS       D6
#define GPIO_CLK      D7

/* Criação de objeto para comunicação com termopar */
MAX6675 termopar(GPIO_CLK, GPIO_CS, GPIO_SO);

//contador de tempo após cruzamento do zero
void checagem_dimmer() {
  if(zero_cross == true) {
    if(y >= dim) {
      digitalWrite(ativador, HIGH); // liga a luz
      y=0;  // reseta o contador
      zero_cross = false; //reseta o zero cross detection
    }
    else {
      y++; // incremento do passo de contador de tempo
```

```
    }
  }
}

//função do cruzamento de zero
void IRAM_ATTR zero_cross_detector() {
  digitalWrite(ativador, LOW);// desliga o triac e a corrente alternada)
  y=0;
  zero_cross = true; // troca o estado para verdadeiro avisando a função de dimmerização que ocorreu o cruzamento de zero
}

//Chamada da função dependente do tempo
Ticker timer1(checagem_dimmer, atualizacao, 0, MICROS_MICROS);

void setup() {
  Serial.begin(115200);
  pinMode(ativador, OUTPUT); // Coloca o pino que liga a corrente alternada como saida
  pinMode(D1, INPUT); //pino de interrupção
  timer1.start();
  attachInterrupt(digitalPinToInterrupt(D1), zero_cross_detector, RISING); //interrupção do cruzamento de zero
  //Definição da temperatura inicial
  T[0] = 1.004233129*termopar.readCelsius()-1.743315757;
```

```
  for (i=1;i<4;i++)
  T[i] = T[0];
}

void loop() {
  // put your main code here, to run repeatedly:
  timer1.update();
  tempo = millis()/1000;
  //Função de escrita da potência no serial em parâmetros ASCII
  if (Serial.available()>0) {
    int x = Serial.read();
    if (x > 10){
      pot = x-32;
    }
  }
  dim = 128-potencia(pot);
  //Propagação do valor de temperatura e impressão dos valores no serial
  if (tempo-tempo_ant>=2){
    T[i] = 1.004233129*termopar.readCelsius()-1.743315757;
    i++;
    if (i==5){
      i=0;
    }
    Tmed = (T[0]+T[1]+T[2]+T[3]+T[4])/5;
```

```
  Serial.print("Temperatura: " + String(Tmed) + "ºC; ");
  Serial.print("Tempo decorrido: " + String(tempo) + "s; ");
  Serial.println("Potencia: " + String(pot) + " %");
  tempo_ant = tempo;
}
```

APÊNDICE E - Código do dimmer na lâmpada

```
#include <Ticker.h>

volatile int activate = D0;    // Saida para o pino  do Opto Triac
volatile int dim = 0;  // Dimming level (0-128)  0 = ON, 128 = OFF
volatile boolean zero_cross = 0;
volatile int y=0;
//int i0=0;
volatile int i=0;
int check_refresh = 1e6/(120*128);
int dim_refresh = 50;
volatile int inc=-1;

/* Devido a problemas de temporização, os extremos podem fazer a lâmpada piscar*/
//Função de checa se o dimmer deve ligar
void dim_check() {
  if(zero_cross == true) {
    if(y>dim) {
      digitalWrite(activate, HIGH); // liga a luz
      y=0;  // reseta o contador de tempo
      zero_cross = false; //reseta o contador de cruzamento de zeros
    }
```

```
    else {
      y++; // incremento no contador de tempo
    }
  }
}

//função que da a forma de comportamento do dimmer
void dimmerize() {
  //modo de operação 1
  /*if (i<=128){
    i++;
  }
  else {
    i=i0;
  }
  dim = i;*/
  //modo de operação 2
if (i<128 && i>0) {
  i+=inc;
}
else{
  inc*=-1;
  i+=inc;
}
```

```
dim = i;
}

//Chamada das funções dependentes do tempo
Ticker timer1(dim_check, check_refresh, 0, MICROS_MICROS);
Ticker timer2(dimmerize, dim_refresh, 0,MILLIS);



//Função que detecta o cruzamento de zero
void IRAM_ATTR zero_cross_detect() {
  digitalWrite(activate, LOW);// turn off TRIAC (and AC)
  y=0;
  zero_cross = true; // set the boolean to true to tell our dimming function that a zero cross has occured
}

void setup()
{
  Serial.begin(115200);
  pinMode(activate, OUTPUT);// Coloca o pino de controla a corrente alteranada como saida
  pinMode(D1, INPUT);
  timer2.start();
  timer1.start();
  attachInterrupt(digitalPinToInterrupt(D1), zero_cross_detect, RISING);
```

```
}



void loop()  {
  timer1.update();
  timer2.update();
}
```

APÊNDICE F - Código aplicado no controle do sistema Lâmpada-LM35DZ

```
#include <Ticker.h>

//função que aproxima linearmente a potência da rede elétrica de corrente alternada
int potencia(float pot) {
  int resp;
  if (pot <= 28.968) { //9
    if (pot <= 4.049) { //5
      if (pot <= 1.248) { //3
        if (pot <= 0.413) { //2
          resp = 26.634*pot;
          }
        else { //2
          resp = 5.988*pot+85269;
          }
        }
        else { //3
          if (pot <= 2.394) { //4
            resp = 3.4904*pot+11.644;
            }
          else { //4
            resp = 2.4169*pot+14.214;
```

```
    }
   }
  }
  else { //5
   if (pot <= 12.518) { //7
    if (pot <= 6.268) { // 6
     resp = 1.8026*pot+16.701;
     }
    else { //6
     resp = 1.28*pot+19.997;
     }
    }
   else { //7
    if (pot <= 19.945) { //8
     resp = 0.9425*pot+24.202;
     }
    else { //8
     resp = 0.7758*pot+27.527;
     }
    }
   }
  }
 else  { //9
  if (pot <= 90.912) { //13
```

```
  if (pot <= 75.071) { //11
    if (pot <= 63.835) { //10
      resp = 0.6596*pot+30.891;
      }
    else { //10
      resp = 0.712*pot+27.55;
      }
    }
  else { //11
    if (pot <= 84.513) { //12
      resp = 0.8473*pot+17.394;
      }
    else { // 12
      resp = 1.0939*pot-3.4505;
      }
    }
  }
else { //13
  if (pot <= 98.238) { //15
    if (pot <= 95.451) { //14
      resp = 1.5422*pot-44.204;
      }
    else { //14
      resp = 2.5117*pot-136.74;
```

```
            }
          }
        else { //15
          if (pot <= 99.688) { //16
            resp = 5.5172*pot-432;
            }
          else { //16
            resp = 32.051*pot-3077.1;
            }
          }
        }
       }
    return resp;
}


volatile int ativador = D0;    // saida para o pino do Opto Triac
float pot; //Variável de potência
volatile int dim = 0;  // nível de dimerização (0-128)  0 = ligado, 128 = desligado
volatile boolean zero_cross = 0; //estado do cruzamento de zero
volatile int y = 0; //contador da intensidade da lampada
float atualizacao = 1e6/(120*128); //unidade: us
float t = 10000;  //unidade: ms periodo de amostragem
float Kc = 0.596831037; //unidade: 1/ºC
int talI = 25; //unidade: s
```

```
int talD = 6.25; //unidade: s
volatile float erro[3] = {0, 0, 0}; //0 = atual; 1 = ultimo; 2 = penultimo;
volatile float sinal_c[2] = {0, 0}; //sinal de controle
float Tsp = 32.74; //unidade: ºC (temperatura do set point)
float T[3] = {0, 0, 0}; //unidade: ºC (temperatura para filtro)
float Tmed; ////unidade: ºC (média das ultimas 3 temperaturas)
int tempo; //unidade: s

//contador de tempo após
void checagem_dimmer() {
  if(zero_cross == true) {
    if(y >= dim) {
      digitalWrite(ativador, HIGH); // liga a luz
      y=0;  // reseta o contador
      zero_cross = false; //reseta o contador de cruzamento de zero
    }
    else {
      y++; // incremento do passo de contador de tempo
    }
  }
}

//função do controle PID
void PID() {
```

```
  //reinicio do ciclo
  //filtro de temperatura
  T[2] = T[1];
  T[1] = T[0];
  T[0] = analogRead(A0)/3.1;
  Tmed = (T[2]+T[1]+T[0])/3;
  //propagação do erro;
  erro[2] = erro[1];
  erro[1] = erro[0];
  erro[0] = Tsp-Tmed;
  //propagação do sinal de controle
  sinal_c[1] = sinal_c[0];
  //equação do controle discretizada na forma de velocidade
  //100*Kc para que o sinal_c saia em porcentagem e não fração
  sinal_c[0] = sinal_c[1] + 100*Kc*((erro[0]-
erro[1])+(t/1000)/talI*erro[0]+talD/(t/1000)*(erro[0]-2*erro[1]+erro[2]));
  //"função de transferência do elemento final de controle"
  dim = 128-potencia(sinal_c[0]);
  //limite de segurança a ser emitido para a lâmpada
  if (dim>120) {
    dim = 120;
  }
}
```

```
//função de escrita dos dados no serial
void Print() {
    Serial.println("Temperatura: " + String(T[0]) + "ºC; " + "tempo: " + String(tempo) + "s");
}

//Definição do set point automática
void SP() {
  Tsp = analogRead(A0)/3.1 + 5;
  T[0] =  analogRead(A0)/3.1;
  T[1] = T[0];
  T[2] = T[1];
}

//chamada das funções que utilizam a bilioteca Ticker
Ticker timer1(checagem_dimmer, atualizacao, 0, MICROS_MICROS);
Ticker timer2(PID, t, 0,MILLIS);
Ticker timer3(Print, 1000, 0, MILLIS);
Ticker timer4(SP, 1000, 1, MILLIS);

//função do cruzamento de zero
void ICACHE_RAM_ATTR zero_cross_detector() {
  digitalWrite(ativador, LOW);// desliga o triac
  //y=0;
  zero_cross = true; //avisa o chrcagem_dimmer que o zero foi cruzado
```

```
}


void setup(){
  Serial.begin(115200);
  pinMode(ativador, OUTPUT); // coloca o pino que ativa o triac como saida
  pinMode(D1, INPUT); // pino de detecção do cruzamento de zero
  timer2.start();
  timer1.start();
  timer3.start();
  timer4.start();
  attachInterrupt(digitalPinToInterrupt(D1), zero_cross_detector, RISING);
}


void loop()  {
  timer1.update();
  timer2.update();
  timer3.update();
  timer4.update();
  tempo = millis()/1000;
 }
```

APÊNDICE G – Código aplicado no controle de temperatura do reator.

```
#include <Ticker.h>
#include <max6675.h>

int potencia(float pot) {
  int resp;
  if (pot <= 28.968) { //9
    if (pot <= 4.049) { //5
      if (pot <= 1.248) { //3
        if (pot <= 0.413) { //2
          resp = 26.634*pot;
          }
        else { //2
          resp = 5.988*pot+85269;
          }
        }
        else { //3
          if (pot <= 2.394) { //4
            resp = 3.4904*pot+11.644;
            }
          else { //4
            resp = 2.4169*pot+14.214;
```

```
    }
   }
  }
  else { //5
   if (pot <= 12.518) { //7
    if (pot <= 6.268) { // 6
     resp = 1.8026*pot+16.701;
     }
    else { //6
     resp = 1.28*pot+19.997;
     }
    }
   else { //7
    if (pot <= 19.945) { //8
     resp = 0.9425*pot+24.202;
     }
    else { //8
     resp = 0.7758*pot+27.527;
     }
    }
   }
  }
 else  { //9
  if (pot <= 90.912) { //13
```

```
  if (pot <= 75.071) { //11
    if (pot <= 63.835) { //10
      resp = 0.6596*pot+30.891;
      }
    else { //10
      resp = 0.712*pot+27.55;
      }
    }
  else { //11
    if (pot <= 84.513) { //12
      resp = 0.8473*pot+17.394;
      }
    else { // 12
      resp = 1.0939*pot-3.4505;
      }
    }
  }
else { //13
  if (pot <= 98.238) { //15
    if (pot <= 95.451) { //14
      resp = 1.5422*pot-44.204;
      }
    else { //14
      resp = 2.5117*pot-136.74;
```

```
            }
           }
          else { //15
           if (pot <= 99.688) { //16
            resp = 5.5172*pot-432;
            }
           else { //16
            resp = 32.051*pot-3077.1;
            }
           }
          }
         }
     return resp;
}


volatile int ativador = D0;    // saida para o pino do Opto Triac
float pot = 81;
volatile int dim = 0;  // nível de dimerização (0-128)  0 = ligado, 128 = desligado
volatile boolean zero_cross = 0; //estado do cruzamento de zero
volatile int y = 0; //contador da intensidade da lampada
float atualizacao = 1e6/(120*128); //unidade: us
float t = 12000;  //unidade: s periodo de amostragem
float Kc = 0.01577320684; //unidade: 1/ºC
int talI = 316.25; //unidade: s
```

```
int talD = 79.0625; //unidade: s
volatile float erro[3] = {0, 0, 0}; //0 = atual; 1 = ultimo; 2 = penultimo;
volatile float sinal_c[2] = {81, 81};
float Tsp = 610; //unidade: ºC (temperatura do set point)
float T[5] = {0, 0, 0, 0, 0}; //unidade: ºC (temperatura para filtro)
float Tmed; ////unidade: ºC (média das ultimas 5 temperaturas)
int tempo; //unidade: s
int tempo_ant = 0; // unidade: s
boolean iTsp = 0; //inicializador da temperatura inicial
int i=0;

/* Definições: GPIOs do Arduino utilizado na comunicação com o
   MAX6675 */
#define GPIO_SO       D5
#define GPIO_CS       D6
#define GPIO_CLK      D7

/* Criação de objeto para comunicação com termopar */
MAX6675 termopar(GPIO_CLK, GPIO_CS, GPIO_SO);

void checagem_dimmer() {
  if(zero_cross == true) {
    if(y >= dim) {
      digitalWrite(ativador, HIGH); // liga a luz
```

```
      y=0;  // reseta o contador
      zero_cross = false; //reseta o zero cross detection
    }
    else {
      y++; // incremento do passo de contador de tempo
    }
  }
}

void PID() {
  //reinicio do ciclo
  //filtro de temperatura
  T[4] = T[3];
  T[3] = T[2];
  T[2] = T[1];
  T[1] = T[0];
  T[0] = 1.004233129*termopar.readCelsius()-1.743315757;
  Tmed = (T[0]+T[1]+T[2]+T[3]+T[4])/5;
  //propagação do erro
  erro[2] = erro[1];
  erro[1] = erro[0];
  erro[0] = Tsp-Tmed;
  //propagação do sinal de controle
  sinal_c[1] = sinal_c[0];
```

```
  //equação do controle
  //Kc multiplicado por 100 para o sinal de controle se adequar a função pot
  sinal_c[0] = sinal_c[1] + Kc*100*((erro[0]-
erro[1])+(t/1000)/talI*erro[0]+talD/(t/1000)*(erro[0]-2*erro[1]+erro[2]));
  //função de transferência do elemento final de controle
  if (sinal_c[0] > 100){
    pot = 100;
  }
  else{
    pot = sinal_c[0];
  }
  dim = 128-potencia(pot);
  if (dim > 120) {
    dim = 120;
  }
}

Ticker timer1(checagem_dimmer, atualizacao, 0, MICROS_MICROS);
Ticker timer2(PID, t, 0,MILLIS);

void ICACHE_RAM_ATTR zero_cross_detect() {
  digitalWrite(ativador, LOW);// turn off TRIAC (and AC)
  //y=0;
```

```
  zero_cross = true; // set the boolean to true to tell our dimming function that a zero cross has occured
 }

 void setup(){
  Serial.begin(115200);
  pinMode(ativador, OUTPUT);// Set AC Load pin as output
  pinMode(D1, INPUT);
  timer2.start();
  timer1.start();
  attachInterrupt(digitalPinToInterrupt(D1), zero_cross_detect, RISING);
  T[0] = 1.004233129*termopar.readCelsius()-1.743315757;
  for (i=1;i<4;i++)
  T[i] = T[0];
  Tmed = T[0];
 }

 void loop()  {
  timer1.update();
  timer2.update();
  tempo = millis()/1000;
  if (tempo-tempo_ant>11){
   Serial.print("T: " + String(Tmed) + "ºC; ");
   Serial.print("Pot :" + String(pot) + "%; ");
```

```
    Serial.println("T: " + String(tempo) + "s");
    tempo_ant = tempo;
  }
}
```