Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Curso de Graduação em Engenharia Elétrica

EXPERIMENTOS DIDÁTICOS PARA IDENTIFICAÇÃO E CONTROLE PID UTILIZANDO FERRAMENTA DE GERAÇÃO AUTOMÁTICA DE CÓDIGO

Camila de Oliveira Abrantes

CAMPINA GRANDE
DEZEMBRO DE 2019

Camila de Oliveira Abrantes

EXPERIMENTOS DIDÁTICOS PARA IDENTIFICAÇÃO E CONTROLE PID UTILIZANDO FERRAMENTA DE GERAÇÃO AUTOMÁTICA DE CÓDIGO

Trabalho de conclusão de curso submetido à Unidade Acadêmica de Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau Bacharel em Ciências no Domínio da Engenharia Elétrica.

Orientador: George Acioli Junior

CAMPINA GRANDE
DEZEMBRO DE 2019

Camila de Oliveira Abrantes

EXPERIMENTOS DIDÁTICOS PARA IDENTIFICAÇÃO E CONTROLE PID UTILIZANDO FERRAMENTA DE GERAÇÃO AUTOMÁTICA DE CÓDIGO

Trabalho de conclusão de curso submetido à Unidade Acadêmica de Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau Bacharel em Ciências no Domínio da Engenharia Elétrica.

George Acioli Junior
Orientador

Rafael Bezerra Correia Lima
Professor Convidado

CAMPINA GRANDE
DEZEMBRO DE 2019

Dedico este trabalho à minha família biológica e também à família que construi aqui em Campina Grande.

Agradecimentos

Agradeço principalmente aos meus pais, que permitiram que eu pudesse realizar meu sonho de me tornar engenheira e não mediram esforços para isso.

Agradeço também aos meus avós, tios e demais membros da famíia, que se fizeram presentes e oraram sempre por mim e para que este dia chegasse.

Agradeço aos funcionários do departamento, Gutemberg Gonçalves, Adail Ferreira e Tchaikowisky Oliveira por toda atenção e auxílio durante esses anos.

Agradeço também a todos os meus professores, que de alguma forma contribuiram para o meu crescimento dentro do curso, em especial a George Acioli, meu orientador, e a Pericles Barros por terem me dado a oportunidade de trabalhar com eles e por me receberem de braços abertos.

Gostaria de agradecer também ao Professor Wamberto Lira de Queiroz, que foi meu tutor durante mais de dois anos e reconheço sua participação, assim como a do Programa de Educação Tutorial (PET), na minha formação acadêmica e no meu crescimento pessoal.

Agradeço a todos os meus queridos amigos, em especial à Anna Caroline Lima, que esteve ao meu lado desde quando aprendi a ler e dividiu comigo suas alegrias. Também à Lucas Sales, Breno Alves, Vinicius Freire, Marianne Bianca, Caio Villar e João Carlos, que me acompanharam durante esses últimos anos de curso e fazem parte dessa conquista.

Agradeço em especial a minha querida amiga Wislayne Silva, que durante esses anos dividiu comigo, além de um apartamento, todos os seus momentos de conquistas e fraquezas, se tornando a irmã que nunca tive.

Também gostaria de agradecer a todos aqueles que não foram citados aqui, mas que contribuiram para este momento.

As coisas não caem do céu. É preciso ir buscá-las. Correr atrás, mergulhar fundo, voar alto. Muitas vezes, será necessário voltar ao ponto de partida e começar tudo de novo. As coisas, eu repito, não caem do céu. Mas quando, após haverem empenhado cérebro, nervos e coração, chegarem à vitória final, saboreiem o sucesso gota a gota. Sem medo, sem culpa e em paz. É uma delícia. Sem esquecer, no entanto, que ninguém é bom demais. Que ninguém é bom sozinho. E que, no fundo no fundo, por paradoxal que pareça, as coisas caem mesmo é do céu, e é preciso agradecer.

Luiz Roberto Barroso

Resumo

Controladores PID são largamente utilizados no controle de processos industriais. Desde sua crescente usabilidade na revolução industrial e o avanço tecnológico, suas aplicações são cada vez mais intensas. No entanto, muitos controladores utilizados na industria são mal sintonizados, por esse motivo, os sistemas de controle, por mais simples que pareçam, ainda são alvo de muito estudo. Diversos métodos de sintonia foram criados, porém nunca se chegou a um consenso de qual é o melhor, pois cada método pode ter o melhor desempenho em processos diferentes. Nesse contexto, no presente trabalho são realizados experimentos didáticos de identificação e sintonia de controladores PID, analisando 3 métodos de sintonia: Ziegler-Nichols (ZN), SIMC e CHR, aplicados ao controle de temperatura usando o módulo didático apresentado em Lima et. al. A metodologia usada explora os principais conceitos de controle, apresentando as atividades realizadas com o módulo citado e discutindo os resultados obtidos e os índices de desempenho para cada controlador.

Palavras-chave: Identificação de Sistemas, Sintonia de Controladores, Projeto Baseado em Modelo.

Lista de Figuras

Lista de Tabelas

Sumário

1 Introdução

Os sistemas de controle já fazem parte do nosso dia-a-dia e cada vez mais tem demonstrado a sua importância para os avanços tecnológicos da sociedade. Devido a isso, novas ideias e técnicas surgem com bastante frequência na área, o que torna o trabalho do projetista desafiador, pois deve manter-se atento as novidades que surgem constantemente (Félix, 2018).

As aplicações dessas tecnologias de controle em processos é bastante vantajosa. Objetivando o aumento da qualidade de produtos e da confiabilidade ao mesmo tempo que reduz o tempo de produção, o projeto adequado de sistemas de controle é essencial, podendo evitar, por exemplo, a deterioração antecipada de equipamentos, que sem o devido gerenciamento podem trabalhar acima da sua capacidade. Por isso, a cada dia intensifica-se as pesquisas sobre esse assunto, de maneira que exista uma melhora contínua no desempenho de processos automáticos (Mendes, 2017).

Na Figura 1 é apresentado um sistema genérico controlado. Como observado, os sensores recebem sinais do meio para que seja possível comparar o comportamento desejado com o real para posteriormente realizar o controle propriamente dito.

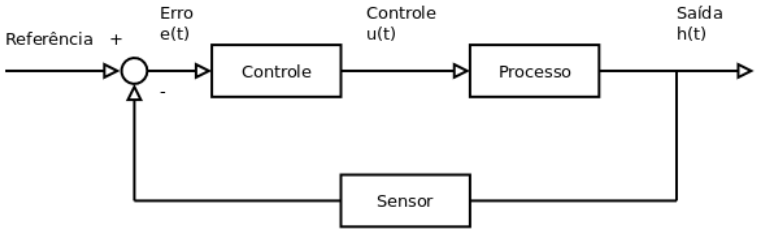

Figura 1: Sistema de controle em malha fechada [Fonte: OGATA, Katsuhiko. Engenharia de Controle Moderno, 5a Ed, Pearson, 2011.]

Com o crescente aumento no grau de sofisticação das atividades humanas e também com o avanço da industria eletrônica, cada vez mais as tarefas de um sistema de controle estão sendo desempenhadas de forma automática, com mais facilidade, eficiência e segurança. Isso permitiu uma ampla disseminação das teorias clássicas de controle. Assim, os sistemas de controle contribuem cada vez mais para o desenvolvimento da tecnologia e civilização (Saraiva, 2011).

Nesse cenário, muitas empresas utilizam-se de placas de alto custo para o controle de processos, limitando a viabilidade em certos casos. Segundo Bartolucci e Andújar (2017) a plataforma Arduino é considerada adequada para tornar-se uma placa de aquisição de sinais de baixo custo. Gartseev, Lee e Krovi (2011) destacam no *software* MATLAB/*Simulink* a facilidade em programar a partir de projetos baseados em modelo, e com a possibilidade de interagir diretamente com o Arduino por meio de bibliotecas específicas no *Simulink*.

Na prática, muitas vezes os controladores industriais são sintonizados através do procedimento de tentativa e erro, fazendo com que o ajuste do processo seja difícil ou ineficiente. Sendo assim, as técnicas de sintonia para o cálculo dos controladores também entram como uma opção mais precisa, simples e eficiente (Saraiva, 2011).

Nesse contexto, as disciplinas de controle são essenciais para a formação do Engenheiro Eletricista, no entanto os estudantes possuem uma tarefa árdua ao iniciar os estudos, isso se deve a grande quantidade de informações que precisa ser assimilada, tendo em vista as inúmeras ideias que lhe são apresentadas. Para isso, os laboratórios e projetos são importantes na construçao do conhecimento e também na motivação do aluno.

1.1 Ensino de Controle nas Universidade

Nos dias atuais, os educadores vem buscando alternativas ao processo tradicional de ensino centrado em memorização e aplicação de conceitos e equações para resolução de exercícios (Mota, 2015). De acordo com (Moratori, 2003), atividades lúdicas aliam o lazer ao desafio, estimulando o interesse dos alunos em solucionar os problemas propostos, mesmo em nível de ensino superior.

Um dos desafios do ensino superior nas áreas de engenharia é estimular essas aplicações práticas dos conhecimentos teóricos adquiridos em sala de aula. Apesar de ambientes simulados fornecerem uma boa condição de aplicabilidade dos conhecimentos do aluno, não ajudam a concretizar tão bem o aprendizado quanto a existência de uma plataforma física, pois apresentará perturbações que podem não estar sendo consideradas no sistema simulado (Lima, 2014).

Para isso, podem ser encontadas no mercado, algumas plataformas, produzidas por empresas como a *Didatech*, *Datapool* e *Feedback*, mas que geralmente são caras e/ou necessitam de muito espaço, tornando a quantidade de alunos por grupo a operá-las cada vez maior. Portanto, sob uma perspectiva educacional, essas condições estão longe das ideais. Primeiramente, o tempo limitado que o aluno pode ter acesso ao *hardware* dificulta que ele tenha familiaridade com a plataforma. Se a plataforma é desconhecida e complexa, o processo de aprendizagem se torna ainda mais difícil. Outro ponto relevante é, a menos que bem conduzido, os alunos deverão trabalhar em grupos grandes, fazendo com que parte do grupo não se engaje nas atividades propostas.

Nas aulas de laboratório, os alunos devem por em prática o assunto estudado nas aulas de teoria. O aprendizado é potencializado pela experiência direta, na qual exemplos concretos com relevância na indústria sejam praticados através de experimentos (Lobo, 2017).

Como alternativa aos kits comerciais muito caros existem algumas ferramentas voltadas para o ensino de controle, um exemplo é o módulo didático apresentado em Lima

et. al., que pode potencializar o ensino e a motivação dos alunos.

Além disso, um kit de baixo custo e de fácil implementação permite que o aluno replique os experimentos em casa a fim de ter mais familiaridade com a plataforma (Lobo, 2017).

1.2 Motivação

As disciplinas de Controle são essenciais na formação do Engenheiro Eletricista, no entanto, nota-se que uma grande parte dos alunos sentem dificuldade para entender os assuntos ministrados na teoria. Isso acontece devido a grande variedade de conceitos e técnicas de projetar sistemas de controle que surgem e que são passadas aos estudantes, exigindo bastante dedicação para assimilar todo o conteúdo. Para facilitar esse processo de aprendizagem, os laboratórios que acompanham as disciplinas teóricas são essenciais para que os estudantes vejam na prática o que já foi ensinado em sala de aula.

Dessa forma, ter uma abordagem que vai além de apenas simulações computacionais pode melhorar o desempenho do aluno na disciplina, além de motivar o interesse e a curiosidade.

1.3 Objetivos

Nesse contexto, o presente trabalho tem por objetivo principal, apresentar experimentos didáticos de identificação e sintonia de controladores PID utilizando o módulo didático apresentado em Lima et. al., permintido o estudo e assimilação das técnicas de sintonia, assim como a análise e a avaliação dos controladores em malha fechada para o controle de processos.

1.4 Estrutura do Documento

Além deste capítulo introdutório, este trabalho conta com mais quatros capitulos:

- Capítulo 2: Fundamentação Teórica, onde são reforçados alguns conceitos teóricos para o entendimento do trabalho desenvolvido;
- Capítulo 3: Materiais e Métodos, onde é apresentado o desenvolvimento das atividades realizadas para a produção deste trabalho;
- Capítulo 4: Resultados e Discussões, onde serão apresentados os resultados e algumas conclusões obtidas nesse trabalho;
- Capítulo 5: Conclusão.

2 Fundamentação Teórica

Neste capítulo são apresentadas bases teóricas para a fundamentação deste trabalho. De forma geral, são reforçados alguns conceitos básicos para o entendimento do trabalho desenvolvido, além da explanação dos *softwares*, já bem conhecidos, MATLAB/*Simulink* e a sua *Toolbox* de geração automática de código para arduino. Também é feita uma apresentação sobre *Model-Based Design*, suas vantagens e abordadens em projetos, além de algumas noções de controle.

2.1 Sistemas de Controle

Os sistemas de controle podem ser de dois tipos: malha aberta ou malha fechada. Os sistemas de controle de malha aberta são aqueles onde a variável de saída não afeta a ação do controlador, ou seja, nesses sistemas não é realizada a medição do sinal de saída e nem a comparação do mesmo com o sinal de referência. Por outro lado, os sistemas de controle de malha fechada, também chamados de sistemas de controle com realimentacão, são aqueles em que o sinal que atua no controlador é a diferença entre a referência e a saída do sistema, conhecida como erro. Para isso utiliza-se uma malha de realimentação realizando a medição do sinal de saída.

Os sistemas de controle de malha fechada apresentam uma vantagem importante em relação aos sistemas de malha aberta, eles são menos sensíveis a distúrbios e as imprecisões dos parâmetros internos da planta. Nas Figuras 2 e 3 pode-se observar a estrutura dos sistemas de controle em malha aberta e em malha fechada, respectivamente.

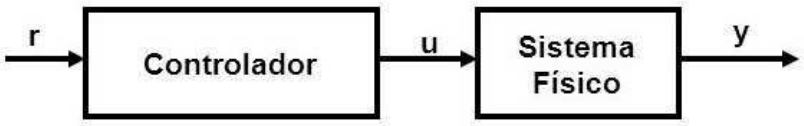

Figura 2: Sistema de controle em malha aberta. [Fonte: Felix, 2018]

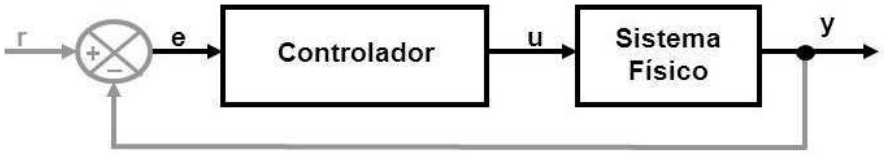

Figura 3: Sistema de controle em malha fechada. [Fonte: Felix, 2018]

Para realizar o controle em malha aberta é necessário grande precisão na construção do modelo da planta e, portanto, na determinação de seus parâmetros. No entanto, para sistemas sem perturbação e onde é possível conhecer as entradas de forma antecipada, o uso do controle de malha aberta torna-se recomendado, tendo em vista que ele apresenta um custo e uma potência menor do o controle de malha fechada (Felix, 2018).

O objetivo de um sistema de controle é fazer a saída de um processo dinâmico seguir uma determinada referência na presença de distúrbios ou mudanças. Para isto, um modelo deve ser obtido, denominado modelo dinâmico ou matemático. Ele pode ser obtido via princípios físicos ou aplicando sinais e medindo sua resposta. O segundo caso usa técnicas de identificação de sistemas, na qual os parâmetros de modelos pré-definidos são estimados (Lobo, 2017).

2.1.1 Ações de Controle Básicas

Um componente fundamental do sistema é o controlador. Um controlador é um dispositivo eletrônico, pneumático, hidráulico ou mecânico que produz sinal de controle, ou também conhecido como sinal de comando, para o sistema. Tal sinal é gerado com base no valor de referência do sistema, que é o valor desejado para a variável de processo a partir de características da planta.

Controlador *On-Off*

Uma das ações mais básicas é a ação liga-desliga (*on-off*). Neste tipo de ação, permite-se que o elemento de atuação assuma apenas dois estados, ligado e desligado. Logo, nos controladores *on-off*, o sinal de controle $u(t)$ é obtida por uma função não-linear em relação ao sinal do erro $e(t)$, como mostrado na Equação 1.

$$e(t) \;=\; r(t) \;-\; y(t) \tag{1}$$

onde $r(t)$ é o sinal de referência e $y(t)$ é a saída do processo, ambos no domínio do tempo. Matematicamente, tem-se que:

$$\begin{aligned} e(t) > 0 \;\; &\Rightarrow \;\; Modo\ ON(u(t) = u_{max}) \\ e(t) < 0 \;\; &\Rightarrow \;\; Modo\ OFF(u(t) = u_{min}) \end{aligned}$$

Controlador Proporcional (P)

O controlador proporcional é aquele que produz um sinal de controle linearmente proporcional ao erro do sistema. Com isso, a lei de controle será conforme as Equações 2, 3 e 4.

$$u(t) = K_p e(t) \tag{2}$$
$$U(s) = K_p E(s) \tag{3}$$
$$G_c(s) = \frac{U(s)}{E(s)} = K_p \tag{4}$$

onde $U(s)$ é o sinal de controle, $E(s)$ é o sinal de erro, $G_c(s)$ é a função de transferência do controlador, todos representados em Laplace, e K_p é o ganho proporcional [adimensional].

Controlador Proporcional-Integral (PI)

O controlador PI permite rejeitar completamente perturbações constantes. Nesta ação de controle é adicionado um termo integrador, que faz com que o sinal de controle varie com uma taxa proporcional ao sinal de erro. A lei de controle pode ser representada pelas Equações 6, 7 e 8.

$$u(t) = K_p e(t) + \int_{t_0}^{t} K_i e(\tau) d\tau \tag{6}$$
$$U(s) = \left(K_p + \frac{K_i}{s} \right) E(s) \tag{7}$$
$$G_c(s) = K_p + \frac{K_i}{s} \tag{8}$$

onde K_i é o ganho integral.

Controlador Proporcional-Integral-Derivativo (PID)

Este controlador possui as vantagens das ações de controle proporcional e integral combinadas com a ação de controle derivativo, que adiciona um termo derivador, fazendo que com que o sinal de controle tenha um componente proporcional a taxa de variação do sinal de erro.

O controlador PID pode ser representado como nas Equações 10 e 11.

$$u(t) = K_p e(t) + \int_{t_0}^{t} K_i e(\tau) d\tau + K_d \frac{d}{dt} e(t) \tag{10}$$

$$G_c(s) = K_p + \frac{K_i}{s} + K_d s \tag{11}$$

ou equivalente nos processos industriais como na Equação 12.

$$G_c(s) = K_p \left(1 + \frac{1}{T_i s} + T_d s\right) \tag{12}$$

onde K_p é o ganho proporcional do controlador [adimensional], K_d é o ganho derivativo do controlador, T_i é a constante de tempo integral [segundos] e T_d é a constante de tempo derivativa [segundos].

2.1.2 Função de Transferência

A função de transferência, $H(s)$, de um sistema é definida como a relação entre a transformada de Laplace da variável de saída e a transformada de Laplace da variável de entrada, supondo que as condições iniciais do sistema sejam nulas.

$$H(s) = \frac{Y(s)}{U(s)}$$

A ordem de um sistema é definida pela ordem da mais alta potência no denominador, $U(s)$.

Através da função de transferência de um sistema é possível estudar a resposta do sistema alterando apenas a sua entrada. Sendo ela desconhecida, é possível obtê-la de forma experimental, introduzindo sinais de entrada e observando o correspondente sinal de saída.

Quando $u(t)$ é um impulso unitário, temos que $H(s) = Y(s)$, portanto, podemos dizer que $H(s)$ é a transformada de Laplace da resposta ao impulso $h(t)$.

2.2 Identificação de Sistemas via Mínimos Quadrados

A identificação de sistemas é uma ferramenta para obter modelos de sistemas a partir de dados de entrada e saída e é essencial para o projeto de controladores. Ela se torna muito útil em casos que o sistema é muito complexo e fica muito difícil de se determinar um modelo a partir das leis físicas conhecidas (Bittencourt, 2007). Assim, a identicação de sistemas surge como uma ferramenta importante para o projetista do sistema de controle.

O método dos mínimos quadrados é uma solução analítica para a identificação de sistemas e consiste em encontrar um melhor ajuste para um conjunto de dados

tentando minimizar a soma dos quadrados das diferenças entre o valor estimado e os dados observados.

Supõe-se que o modelo do processo é descrito pela função de transferência como vista na Equação 13.

$$G(s) = \frac{Y(s)}{U(s)} = \frac{G_o}{T_1 s \ + \ 1} e^{-Ls} \tag{13}$$

Para a relação 13, a saída $y(t)$ para uma entrada em degrau de amplitude h, é dada pela Equação 14.

$$y(t) = hG_o \left(1 \ - \ e^{-\frac{t-L}{T1}}\right) + \omega(t), \ t \ \geq L \tag{14}$$

onde $\omega(t)$ é o ruido branco presente na medição de y(t). A Equação 14 pode ser reescrita como a Equação 15.

$$e^{-\frac{t-L}{T1}} = 1 - \frac{y(t)}{hG_o} + \frac{\omega(t)}{hG_o}, \ t \ \geq L \tag{15}$$

Integrando $y(t)$ de $t = 0$ até $t = \tau$ com $\tau \ \geq L$ ($y(t) \ = \ 0$ para $t \ \leq L$), tem-se a Equação 16.

$$\int_0^\tau y(t) \ dt = hG_o \left(t + T_1 e^{-\frac{t-L}{T_1}}\right) \Big|_{t=L}^{t=\tau} + \int_0^\tau \omega(t) \ dt \tag{16}$$

Usando a Equação 15 e o fato de que $y(L) \ = \ 0$ temos a Equação 17.

$$\int_0^\tau y(t) \ dt = hG_o \left[\tau \ - \ L \ + \ T_1 \ \frac{y(t)}{hG_o}\right] + [T_1\omega(t)]|_{t=L}^{t=\tau} + \int_0^\tau \omega(t) \ dt \tag{17}$$

Define-se:

$$A(\tau) = \int_0^\tau \ y(t) \ dt$$

e

$$\delta(\tau) = [T_1\omega(t)]|_{t=L}^{t=\tau} + \int_0^\tau \omega(t) \ dt$$

Então a Equação 17 pode ser reescrita como a Equação 18.

$$A(\tau) = hG_o \left[\tau - L + T_1 \frac{y(t)}{hG_o}\right] \ + \ \delta(\tau) = \left[\ h\tau \quad -h \quad y(\tau) \ \right] \left[\begin{array}{c} G_o \\ G_o L \\ T_1 \end{array} \right] \ + \ \delta(\tau) \tag{18}$$

e modo que

$$A(\tau) = \phi^T(\tau)\theta \ + \ \delta(\tau) \tag{19}$$

que é equivalente ao sistema linear usado na solução dos mínimos quadrados com ruído dado pelo termo $\delta(\tau)$.

Portanto,

$$\phi(\tau) \ = \ \left[\begin{array}{ccc} h\tau & -h & y(\tau) \end{array} \right]^T, \tag{20}$$

$$\theta \ = \ \left[\begin{array}{ccc} G_o & G_oL & T_1 \end{array} \right]. \tag{21}$$

2.3 Método da Curva de Reação

A aproximação de primeira ordem para funções de transferência é obtida aplicando-se um degrau unitário na entrada do sistema, gerando em sua saída uma curva chamada 'S', como está representado na Figura 4. Esse método gráfico é também denominado de método da curva de reação [16].

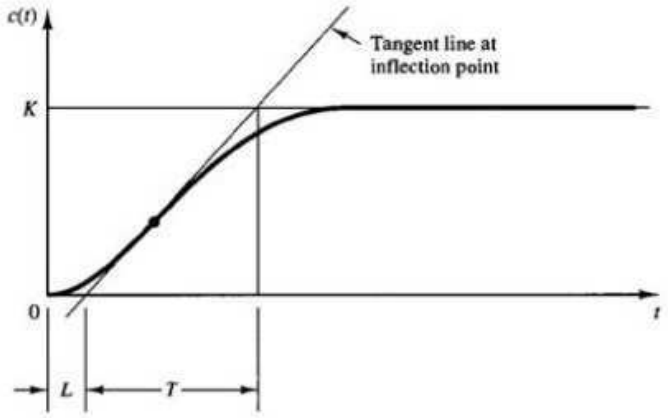

Figura 4: Método da Curva de Reação

A partir de 'S', traçando uma reta tangente ao ponto de inflexão da curva e determinando a intersecção da reta com o eixo do tempo e com o eixo de K, obtém-se a constante de tempo τ (segunda intersecção subtraída do L) , o ganho K (em regime permanente) e o atraso L do processo (corresponde a primeira intersecção). Com L, K e τ calculados, encontra-se a aproximação de primeira ordem com atraso para o sistema:

$$\frac{Y(s)}{U(s)} = \frac{K}{(\tau s + 1)} e^{-Ls} \tag{22}$$

2.4 Método de *Half Rule*

O método de aproximação de *Half Rule* consiste no cancelamento do termo do numerador (T_0s+1) com o denominador ($\tau_0 + 1$) que possua maior constante de tempo τ_0 e que seja mais próxima de T_0. Assim, pode-se aproximar para a primeira e segunda ordem sistemas de ordem superiores com atraso. Para modelos de primeira ordem, a aproximação é dada como mostrado nas Equações 23, 24 e 25 [16].

$$G(s) = \frac{Ke^{-\theta s}}{\tau_1 s + 1} \tag{23}$$

$$Onde,\ \tau_1 = \tau_{10} + \frac{\tau_{20}}{2} \tag{24}$$

$$\theta = \theta_0 + \frac{\tau_{20}}{2} + \sum_{i \geq 3} \tau_{i0} + \sum_j T_{j0}^{inv} + \frac{h}{2} \tag{25}$$

As aproximações são:

$$\frac{T_0 s + 1}{\tau_0 + 1} \approx \begin{cases} \frac{T_0}{\tau_0}, & para\ T_0 \geq \tau_0 \geq \theta \\ \frac{T_0}{\theta}, & para\ T_0 \geq \theta \geq \tau_0 \\ 1, & para\ \theta \geq T_0 \geq \tau_0 \\ \frac{T_0}{\tau_0}, & para\ \tau_0 \geq T_0 \geq 5\theta \\ \frac{(\widetilde{\tau_0}/\tau_0)}{(\widetilde{\tau_0} - \tau_0)s+1}, & para\ \widetilde{\tau_0} \overset{def}{=} min(\tau_0, 5\theta) \geq T_0 \end{cases}$$

2.5 Técnicas de Sintonia

As técnicas de sintonia são utilizadas para calcular as constantes de tempo do controlador, seja ele Proporcional (P), Proporcional – Integral (PI) ou Proporcional- Integral-Derivativo (PID). Uma particularidade é usar a função de primeira ordem com atraso 26 (Batista, 2014).

$$\frac{Y(s)}{U(s)} = \frac{K}{(Ts+1)} e^{-Ls} \tag{26}$$

Dada a função de transferência para um controlador PID:

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

onde K_p é o ganho do controlador, T_i é a constante de tempo integral e T_d é a constante de tempo derivativa. Para os controladores PI é desconsiderado o valor da constante de tempo T_d, resultando na Equação 27.

$$C(s) = K_p \left(1 + \frac{1}{T_i s}\right) \tag{27}$$

Alguns métodos de sintonia são apresentados a seguir.

2.5.1 Método de Ziegler-Nichols (ZN)

A regra de sintonia de ZN é um dos mais clássicos métodos de sintonia para obtenção dos parâmetros do controlador. Obtendo a aproximação de primeira ordem para o sistema, é possível calcular os parâmetros para os controladores utilizando a Tabela 1 (Batista, 2014).

Tabela 1: Parâmetros dos controladores pelo método Ziegler-Nichols

Controlador	K_p	T_i
PI	$\frac{0.9T}{LK}$	$3.3L$

2.5.2 Método SIMC

O método SIMC é uma modificação do Método IMC proposta por Skogestad, onde ele considera a relação entre a constante de tempo, o atraso e a malha fechada desejada.

Nesse contexto, o método "Skogestad IMC"(SIMC) é sugerido ao invés do clássico método IMC devido ter uma rápida resposta com boa robustez quando o parâmetro de sintonia é igual ao atraso da planta L. Além disso, tal método apresenta rápida acomodação para perturbação de carga. Assim para um sistema com aproximação de primeira ordem, os parâmetros do controlador PI podem ser obtidos conforme as regras da Tabela 2 (Batista, 2014).

Tabela 2: Parâmetros dos controladores pelo método SIMC

Controlador	K_p	T_i
PI	$\frac{1}{K}\frac{T}{L+\tau_c}$	$min\{T, 4(\tau_c + L)\}$

2.5.3 Método CHR

Esse método foi proposto por Chien, Hrone e Reswick (CHR) em 1952, com a intenção de obter a resposta mais rápida possível sem sobressinal e também com 20% de sobressinal. Os parâmetros do controlador CHR com 0% de sobressinal podem ser calculados segundo a Tabela 3 (Batista, 2014).

Tabela 3: Parâmetros dos controladores pelo método CHR para 0% de sobressinal

Controlador	K_p	T_i
PI	$\frac{0.35T}{KL}$	$1.2T$

2.6 Análise de Desempenho

A teoria de controle moderno admite que o engenheiro de sistema pode especificar quantitativamente o desempenho requerido do sistema. Então, um índice de desempenho é uma medida quantitativa do sistema e é escolhido de modo que seja dada ênfase para as especificações importantes do sistema.

Nas subseções a seguir, abordam-se os principais índices para avaliação do desempenho e robustez dos sistemas de controle em malha fechada com os controladores propostos.

A Figura 5 ilustra alguns pontos que são levados em consideração e são postos como fundamentais para essa análise do sistema.

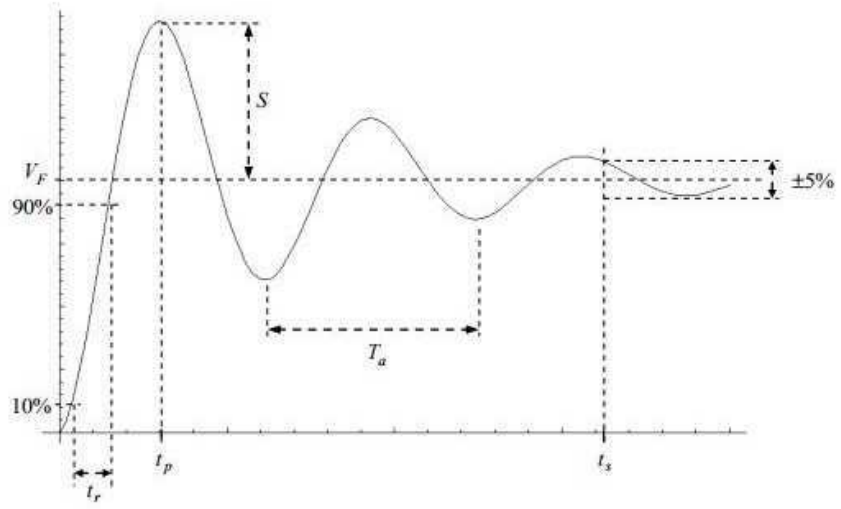

Figura 5: Parâmetros de desempenho

Margem de Ganho (GM)

Define-se como margem de ganho o fator limite em que o ganho pode ser aumentado sem que o sistema perca a estabilidade. Sabendo que a margem de ganho é o módulo da função quando a fase é -180º em malha aberta, é possível calcular seu valor a partir da Equação 28

$$GM = \frac{1}{|G(jw_m)H(jw_m)|} \tag{28}$$

sendo w_m a frequência em que a função de malha aberta tem fase -180º.

Quanto maior a margem de ganho, menor será a sensibilidade do sistema a alguma perturbação.

Margem de Fase (PM)

Similar a margem de ganho, a margem de fase indica o valor angular limite em que a função pode ser acrescida ou decrescida sem que o sistema perca a estabilidade. É possível calcular seu valor através da Equação 29

$$PM = 180 + argG(jw_o)H(jw_o) \tag{29}$$

sendo w_o a frequência para que $|G(jw_m)H(jw_m)| = 1$.

Tempo de Subida (t_r)

Tempo de subida é o tempo em que a resposta leva entre a primeira vez que cruza um determinado limite inferior e a primeira vez que cruza um determinado limite superior. Estes limites são geralmente definidos em percentagem do valor final. É comum usar tempos de subida 10%–90%, 5%–95% e 0–100%.

Tempo de Acomodação (t_s)

Tempo ao fim do qual a resposta se encontra definitivamente dentro de determinada margem em torno do valor final. É habitual definir-se a largura dessa margem em percentagem do valor final, e é frequente a utilização duma margem de 5% .

Sobressinal (M_p)

Diferença entre o valor máximo e o valor final da resposta, geralmente medida como percentagem ou fração do valor final. O sobressinal máximo (calculado em percentual) indica diretamente a estabilidade relativa do sistema.

2.7 *Model-Based Design*

Model-Based Design (MBD) ou Projeto Baseado em Modelo é uma abordagem matemática e visual para o desenvolvimento de sistemas de controle. É o uso sistemático de modelos ao longo do processo de desenvolvimento para projetos, análise, simulação, geração automática de código e verificação. É bastante utilizado no controle de movimento e em aplicações aeroespaciais e automotivas (MathWorks, Inc).

Essa abordagem também ajuda a otimizar o projeto geral do sistema. Por meio do MBD, engenheiros podem observar se o funcionamento do sistema ocorre como o esperado, mesmo antes do *hardware* estar disponível para teste. Desenvolvedores de software embarcados podem gerar automaticamente o código a partir dos modelos de simulação.

Na Figura 6 podemos ver as etapas do desenvolvimento do projeto baseado em modelo. Primeiro é preciso de uma especificação gráfica, pode ser algo que já esteja no *Simulink* ou algo ainda no papel que especifique o projeto. Com o sistema já modelado em *Simulink* é possível testar e validar seu projeto antes do *hardware* estar disponível. Por fim, depois de verificar que o funcionamento do sistema ocorre como deveria é possível gerar código automaticamente e enviá-lo para a placa sem precisar escrever nenhuma linha de código.

Figura 6: Projeto baseado em modelo: do conceito ao código [Fonte: MathWorks, Inc.]

Dentre os benefícios atrelados a MBD, além dos previamente citados, pode-se mencionar o fato de que esta técnica proporciona uma maior facilidade a pessoas de diferentes formações fazerem parte da equipe de desenvolvimento, pois as mesmas não precisariam dominar completamente as minúcias de uma linguagem de programação para representar seus modelos e embarcar o código em uma plataforma física, como um microcontrolador (Sarmento, 2015).

A uso do MBD, como mostrado nos paragrafos anteriores, é um fator de mudança para o desenvolvimento de sistemas embarcados. Tanto para projetos da ordem de milhares ou milhões de dolares, como para de projetos de baixo custo, em todos os cenários empresas, estudantes e pesquisadores obtem vários benefícios.

2.8 Arduino

O Arduino foi criado em 2005 no norte da Itália por 5 pesquisadores: Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, e David Mellis. Eles tinham como objetivo principal desenvolver um dispositivo que fosse ao mesmo tempo barato, funcional e fácil de usar e programar. Dessa forma, seria acessível a estudantes e a projetistas amadores. Além disso, foi criado um conceito de *hardware* livre, o que significa que qualquer um pode montar, modificar, melhorar e personalizar o Arduino, partindo do mesmo *hardware* básico, um exemplo disso é o Genuíno, que é uma versão mais barata do arduino, que também é vendida comercialmente (arduino.cc).

Ao longo dos anos, o Arduino tem sido o cérebro de milhares de projetos, desde objetos do cotidiano até instrumentos científicos complexos. Uma comunidade mundial de criadores - estudantes, amadores, artistas, programadores e profissionais - reuniu-se em torno dessa plataforma de código aberto, e suas contribuições resultaram em uma incrível quantidade de conhecimento acessível que pode ser de grande ajuda para

aqueles que estão começando (arduino.cc).

Figura 7: Arduino Uno [Fonte: arduino.cc]

As placas Arduino são compostas por microcontroladores da Atmel (demais especificações do Arduino Uno podem ser vistas na Tabela 4) e podem ser facilmente conectadas ao computador e programadas pela IDE (*Integrated Development Environment*) ou Ambiente de Desenvolvimento Integrado, utilizando uma linguagem baseada em C/C++, sem a necessidade de equipamentos extras além de um cabo USB.

Tabela 4: Especificações da Placa Arduino Uno

Microcontrolador	ATmega328P
Tensão de operação	5V
Tensão de entrada	7-12V
Pinos de E/S Digitais	0 - 13
Pinos de saída PWM	3,5,6,9,10 e 11
Pinos de entrada analógica	A0 - A5
Peso	25g

A princípio, o Arduino era destinado a estudantes sem formação em eletrônica e programação, mas aos poucos foi atingindo uma comunidade mais ampla e para atender as novas necessidades e desafios começou a mudar e se adaptar. Essa expansão da plataforma ocorreu principalmente porque outros dispositivos e microcontroladores eram muito caros e dificeis de usar, sendo inacessíveis a estudantes e amadores.

Atualmente, existem diversos materiais online e livros para aqueles que querem iniciar na plataforma e também para quem já tem experiencia e pretende projetos mais avançados. Para isso, os desenvolvedores possuem uma gama de opções de plataformas para vários tipos de aplicações, das mais simples as mais complexas. Para citar alguns além do UNO, que está na categoria de iniciantes, temos o Arduino Due, para os que já possuem experiencia; Arduino Ethernet, na categoria de internet das coisas; e o LilyPad, desenvolvido para ser costurado em roupas e dispositivos vestíveis.

2.9 Matlab e *Simulink*

O MATLAB é um ambiente interativo de linguagem de alto nível para computação numérica, visualização e programação desenvolvido pela *MathWorks, Inc.*

Nos últimos anos, o MATLAB tem se estabelecido como referência em computação numérica, processamento de sinais e desenvolvimento de algoritmos. Por muito tempo, a versão profissional do *software* era a única disponível. A chegada da versão mais barata para estudantes possibilitou seu uso em sala de aula.

A família de programas MATLAB inclui o programa principal e mais uma variedade de *toolboxes*, uma coleção de arquivos especiais que estendem a funcionalidade do programa principal. Juntos, o programa principal e o *Control System Toolbox* propiciam a capacidade para projetar e analisar sistemas de controle.

Figura 8: Navegador de bibliotecas do Simulink. [Fonte: MathWorks, Inc.]

O *Simulink* é integrado com MATLAB, sendo capaz de incorporar algoritmos do programa principal em seus modelos e exportar resultados de simulações para o MATLAB, possibilitando análises posteriores. É um ambiente de diagramas de blocos para simulação de projetos baseados em modelos. Ele suporta projeto a nível do sistema, simulação, geração automática de código e verificação contínua de sistemas embarcados.

Engenheiros de automação tem descoberto cada vez mais que a simulação no desenvolvimento de sistemas de controle avançados é uma ferramenta vital para otimizar tempo e custo do projeto. Como uma ferramenta de desenvolvimento, o *Simulink* tem se tornado padrão de excelência pela flexibilidade e pela sua capacidade de simulação e modelagem. Como resultado de sua arquitetura aberta, o Simulink permite que engenheiros criem bibliotecas de blocos para que possam apoiar trabalhos científicos futuros.

Além da gama de projetos que poderiam ser desenvolvidos no MATLAB/*Simulink*, ainda podemos usar os pacotes de suporte disponibilizados pelo *software* e que podem ser instalados quando necessário para aplicações mais específicas. Nesse trabalho, foram usados os pacotes de suporte do MATLAB e do *Simulink* para Arduino.

2.9.1 *Simulink Coder*

O *Simulink Coder* é um complemento do *Simulink* que gera automaticamente código em C/C++ a partir de um diagrama de blocos e o carrega para a plataforma selecionada (MathWorks, Inc.). Juntamente com as ferramentas e componentes da MathWorks, o *Simulink Coder* permite:

- Gerar código automático para diversas plataformas diferentes;
- Desenvolvimento rápido e direto para implementação de sistemas;
- Integração do Matlab com o *Simulink*;
- Interface gráfica simples para o usuário;
- Arquitetura aberta e de possível implementação de novas extensões.

2.9.2 Pacotes de Suporte para Arduino

O pacote de suporte do MATLAB e *Simulink* para arduino permite escrever um código ou um modelo que é carregado, compilado e executado na placa arduino. O pacote permite ler e escrever nas entradas e saídas do arduino estabelecendo a comunicação com os dispositivos perifericos.

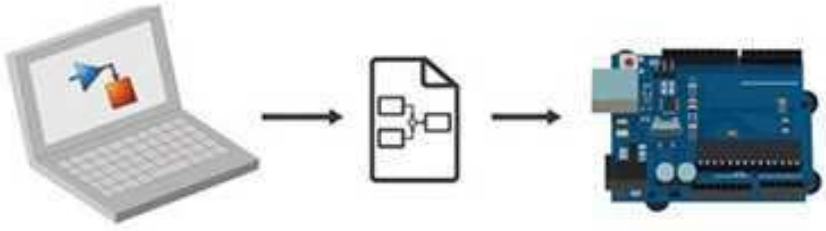

Figura 9: Fluxograma de funcionamento do *Simulink Support Package for Arduino Hardware*. [Fonte: MathWorks, Inc.]

O pacote de suporte do *Simulink* permite carregar as funcionalidades diretamente para a plataforma de modo que as funções só serão executadas enquanto a placa estiver

conectada, mas ao usar a geração automatica de código, o arduino pode continuar executando o código mesmo desconectado, usando outra forma de alimentação.

Mesmo assim, esses pacotes de suporte não permitem a funcionalidade em todas as placas e também nem todas as funções disponiveis na plataforma.

Para isso também é possível instalar outros pacotes que disponibilizam blocos que podem ser usados em conjunto com o Arduino para utilização de sensores. Um exemplo é o *Rensselaer Arduino Support Package Library (RASPLib)*, que permite a utilização de vários sensores comumentes utilizados que permitem medir temperatura, distância, pressão, umidade, entre outras gradezas.

Os principais benefícios de usar os pacotes de suporte do MATLAB e *Simulink* é desenvolver algoritmos mais complexos e avançados e modifica-los facilmente, reduzindo custo e tempo.

2.9.3 Modo Externo

Ao operar no modo externo, o codificador do *Simulink* gera um algoritmo referente ao diagrama de blocos que irá operar no PC e o no *target hardware* (servidor). Esse algoritmo gera um programa executável Matlab *executable* (MEX). Usando a terminologia de computação cliente/servidor, o *Simulink* é o cliente e o programa externo é o servidor. Ao utilizar o Arduino como *target hardware*, a conexão é feita de forma serial através de um cabo USB.

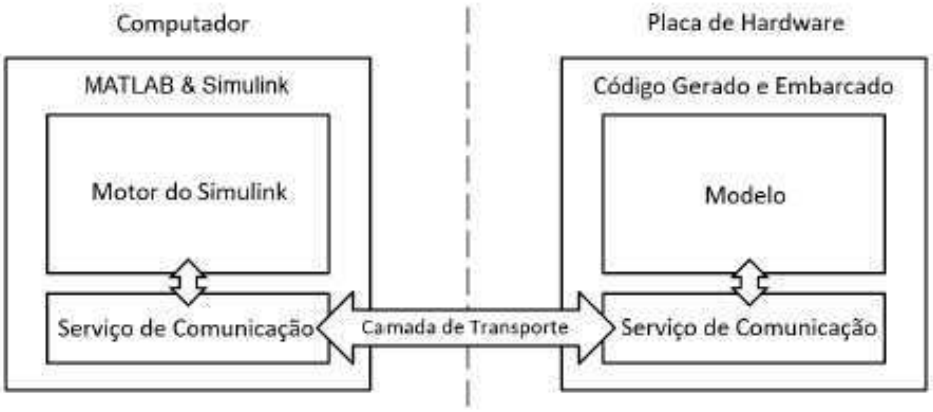

Figura 10: Operação do Simulink no modo externo com um hardware ligado ao PC. [Fonte: Lobo, 2017]

Portanto, o modo externo cria um serviço de comunicação entre o computador e o *hadware* externo. O serviço de comunicação presente no computador recebe pacotes de dados através da camada de transporte e atualiza os valores no modelo do *Simulink*. Por outro lado, ao realizar uma modificação em algum parâmetro no diagrama de

blocos presente no *Simulink*, o *hardware* externo recebe a informação e a atualiza. A Figura 10 mostra como é realizada a comunicação.

O Arduino, como já citado anteriormente, é uma placa criada por uma empresa italiana. Dentre os modelos presentes no mercado, somente o Mega 2560 e o Due permitiam a operação no modo externo do *Simulink*. A partir da versão R2017a, o MATLAB passou a suportar também o Arduino Uno, Leonardo e Mega ADK no modo externo (Lobo, 2017).

2.9.4 *Toolbox* para Identificação de Sistemas do MATLAB

A *toolbox* de identificação de sistemas do MATLAB, desenvolvido por Ljung, tem uma Interface Gráfica para Usuário (GUI), como pode ser vista na Figura 11. A GUI pode ser inicializada com o comando *ident* na linha de comando ou na página principal do MATLAB em *Apps - System Identification*. Cada conjunto de dados que pode ser importado pela *toolbox* são apresentados por um ícone do lado esquerdo da interface e os modelos estimados do sistema ficam do lado direito.

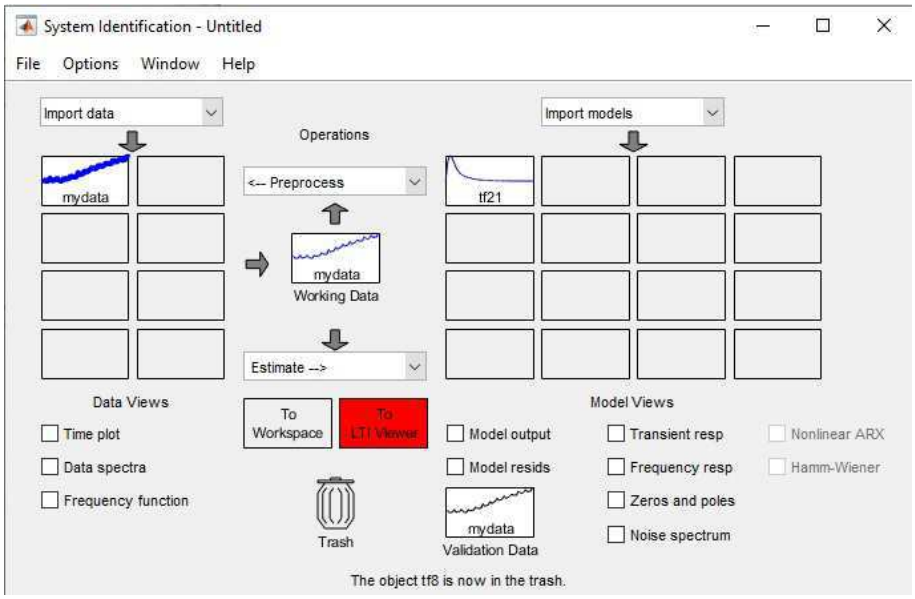

Figura 11: Interface Gráfica do Usuário (GUI) da *toolbox* de identificação do MATLAB.

As opções disponíveis na parte inferior da interface possibilitam plotar gráficos dos dados importados e do modelo estimado, como também das duas informações juntas para uma validação do modelo.

Na opção *Model Output* é possivel visualizar os dados importados e todos os modelos estimados a partir dele, mostrados em ordem do mais aproximado para o mais distinto, como pode ser visto na Figura 12.

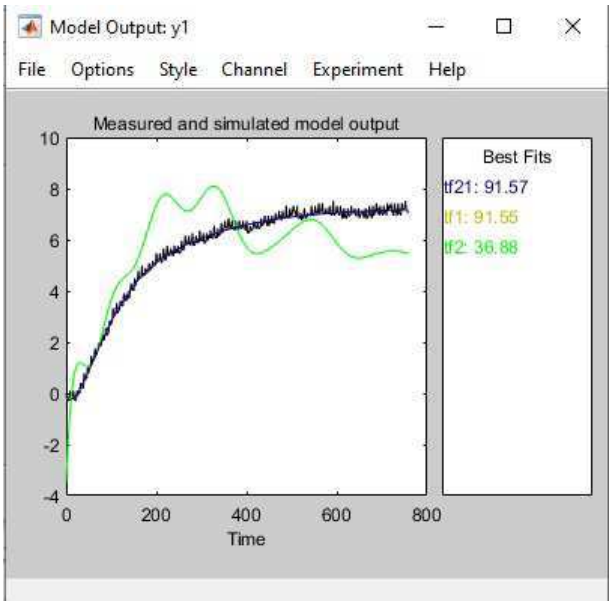

Figura 12: *Toolbox* de identificação do MATLAB - Opção *Model Output*.

2.10 Módulo Didático

Os idealizadores do módulo apresentado em Lima et al. (2017), mostrado na Figura 13, buscavam permitir que os alunos da disciplina de Laboratório de Controle Digital tivessem contato prático com a teoria ministrada em sala de aula, implementando através de uma interface MATLAB-*Simulink*-Arduino técnicas de sintonia de controladores PID e de identificação.

Para a concepção do módulo, foi utilizada a dissipação de calor provocada por dispositivos semicondutores, mais precisamente de Transistores de Efeito de Campo (MOSFET) e da dissipação de calor sobre as placas de circuito impresso.

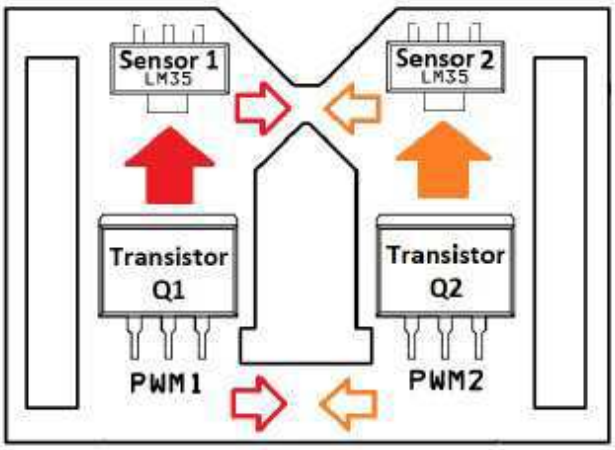

Figura 13: Esquema da disposição dos elementos eletrônicos principais no módulo didático. [Fonte: Lima, 2017]

Na Figura 13 é possível observar a disposição dos transistores e suas distâncias dos sensores LM35, cuja função é obter a temperatura da placa de circuito impresso.

O projeto do módulo seguiu a padronização dos *shields* proposta pela arduino para as plataformas Arduino Mega 2560 e Arduino Uno.

Para o uso do módulo em sala de aula, foi desenvolvida uma interface que pode ser vista na Figura 14. O valores medidos pelos sensores, assim como do *DutyCycle* são plotados nesse gráfico, com tempo de amostragem igual a 2 segundos.

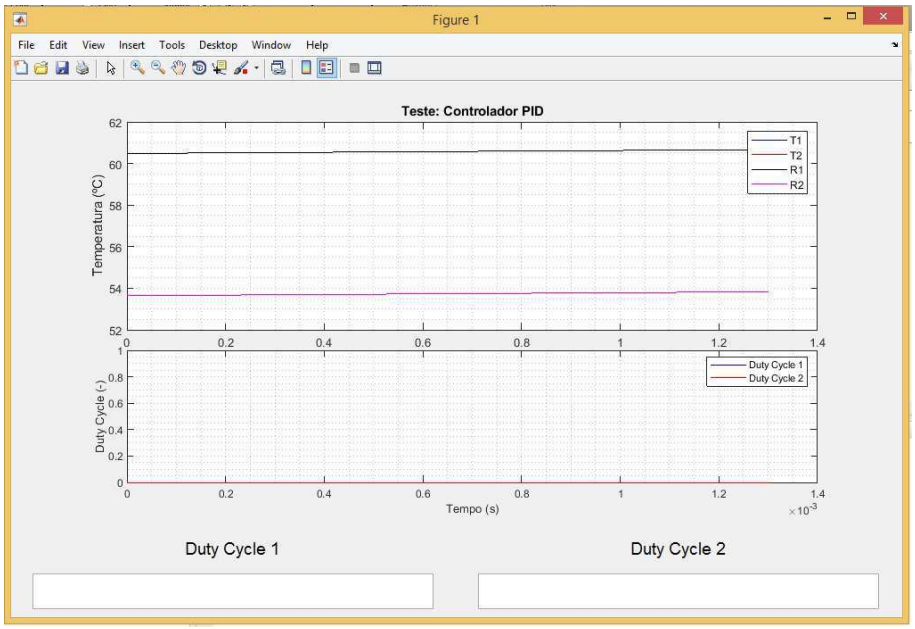

Figura 14: Interface Gráfica para o Usuário usada para medições usando o Módulo.

O módulo trabalha em 3 modos de operação, são eles:

- Malha Aberta (MA): nesse modo, o valor do *DutyCycle* varia entre 0 a 1, podendo ser definido pelo usuário;
- Malha Fechada (MF): nesse modo, o valor de referência, em °C, é escolhido pelo usuário, assim como os valores de K_p e T_i para os controladores;
- FeedForward (FF): O valor do *DutyCycle* também varia entre 0 a 1 e pode ser escolhido pelo usuário. A entrada da malha nesse modo de operação é considerada uma perturbação para a outra malha do sistema, portanto, o controle *feedforward* provê um valor de *DutyCycle* a ser adicionado ao sinal de controle da outra malha de modo a compensar o efeito dessa perturbação.

3 Materiais e Métodos

Neste capítulo são apresentados os materiais e os meios pelos quais foram desenvolvidas as atividades deste tabalho. A seguir são descritos os procedimentos experimentais e as técnicas usadas para identificação do sistema e para o cálculo dos parâmetros dos controladores usados aqui.

3.1 Atividade Experimental

Para a realização da atividade experimental e da aquisição de dados com o uso do Módulo Didático (Lima et al.) utilizando a interface disponibilizada, que pode ser vista na Figura 14, foram seguidos os passos descritos nos próximos paragrafos.

Primeiro, foi escolhido o modo de operação em malha aberta. Então, para a obtenção desses modelos foram excitadas ambas as entradas - U1 e U2, uma por vez, num ensaio de 2500 segundos (42 minutos), dos quais os primeiros 700 segundos foram destinados a colocar o sistema num ponto de operação desejado. A partir daí era observada a dinâmica de subida e de descida, através do degrau aplicado.

Nas Figuras 15 e 16 podem ser vistos os valores medidos pelos sensores 1 e 2 e a entrada aplicada em cada Transistor - Q1 e Q2.

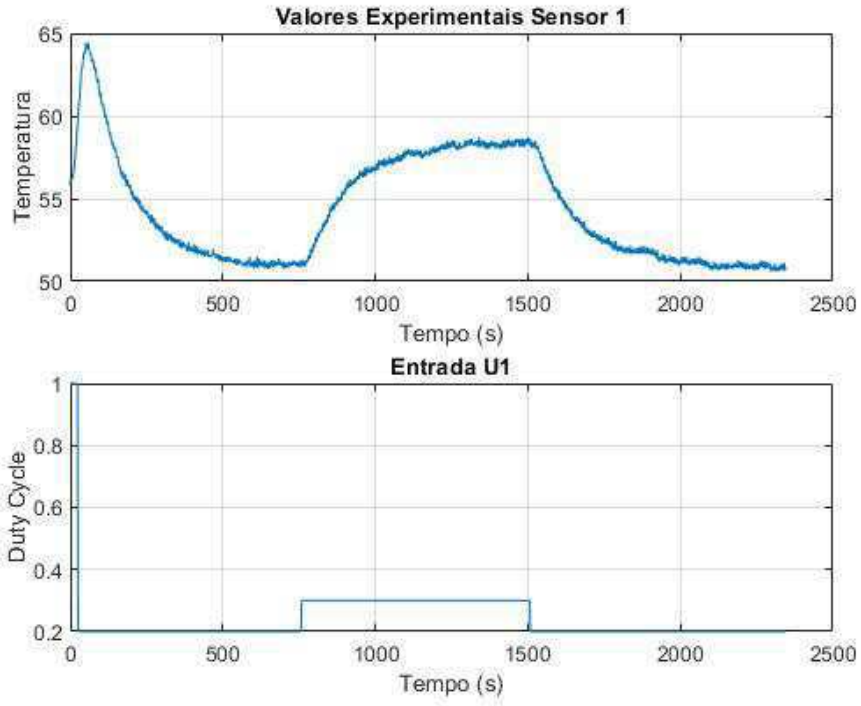

Figura 15: Valores Medidos pelo LM35 - Sensor 1

Para o primeiro sensor, a entrada U1 foi excitada com um degrau unitário durante 20 segundos, em seguida U1 foi setado em 0.2 até que o sistema estivesse em regime.

Para a observação da dinâmica de aquecimento do sensor, foi aplicado um degrau de 0.1 até que o sistema estivesse em regime novamente. Da mesma maneira, para a observação da dinâmica de resfriamento do sensor foi aplicado um degrau de -0.1. O resultado experimental pode ser visto na Figura 15. Para as medições usando o sensor 1, a entrada 2 foi setada igual a zero ($U2 = 0.0$).

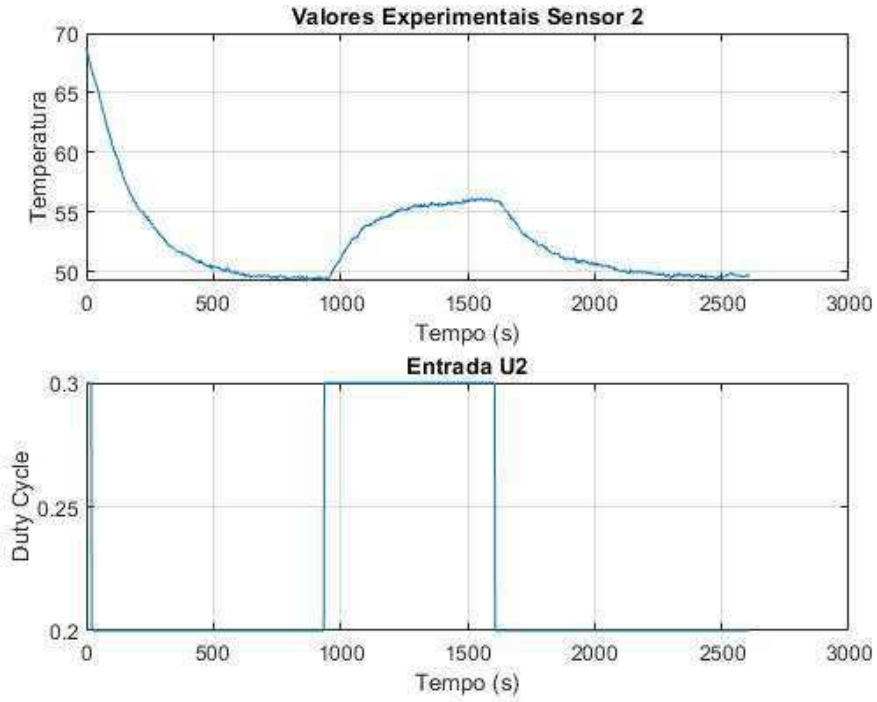

Figura 16: Valores Medidos pelo LM35 - Sensor 2

De maneira equivalente ao sensor 1, para o segundo sensor, a entrada U2 foi excitada com um degrau de 0.3 durante 20 segundos, em seguida U2 foi setado em 0.2 até que o sistema estivesse em regime. Para a observação da dinâmica de aquecimento do sensor, foi aplicado um degrau de 0.1 até que o sistema estivesse em regime novamente. Da mesma maneira, para a observação da dinâmica de resfriamento do sensor foi aplicado um degrau de -0.1. O resultado experimental pode ser visto na Figura 16. Para as medições usando o sensor 2, a entrada 1 foi setada igual a zero ($U1 = 0.0$).

Ao fim do experimento é gerado um aquivo *.mat* com os dados do experimento.

3.2 Identificação do Sistema

Essencial para o projeto de controladores, a identificação de sistemas é muito útil para sistemas muito complexos, difíceis de se determinar a partir das leis físicas. Assim, a identificação surge como uma ferramenta muito importante para o projetista do sistema de controle. Para o sistema descrito aqui, foram usadas duas formas de identificação,

a *Toolbox* de Identificação do MATLAB e o método dos mínimos quadrados. Por fim foi escolhido o modelo mais aproximado do original.

3.2.1 *Toolbox* para Identificação de Sistemas

Para a identificação do sistema usando o *ident*, foram usadas as variáveis do *Workspace* geradas no MATLAB após a realização do experimento. Usando a *Toolbox* a partir do método *Process Model* foram estimados 4 modelos, dois para cada sensor, sendo dois de aquecimento e dois de resfriamento. Todos os modelos estimados aqui tem uma aproximação com o sistema superior a 90%.

Para o Sensor 1 foram estimados os modelos de primeira ordem com atraso das Equações 30 e 31 e o modelo médio está na Equação 32. Os modelos podem ser vistos nas Figuras 17 e 18.

$$G_{aquecimento}(s) = \frac{71.61}{140.4s + 1} e^{-17s} \tag{30}$$

$$G_{resfriamento}(s) = \frac{73.6}{145.7s + 1} e^{-16s} \tag{31}$$

$$G_{medio1}(s) = \frac{72.61}{143 + 1} e^{-16.5s} \tag{32}$$

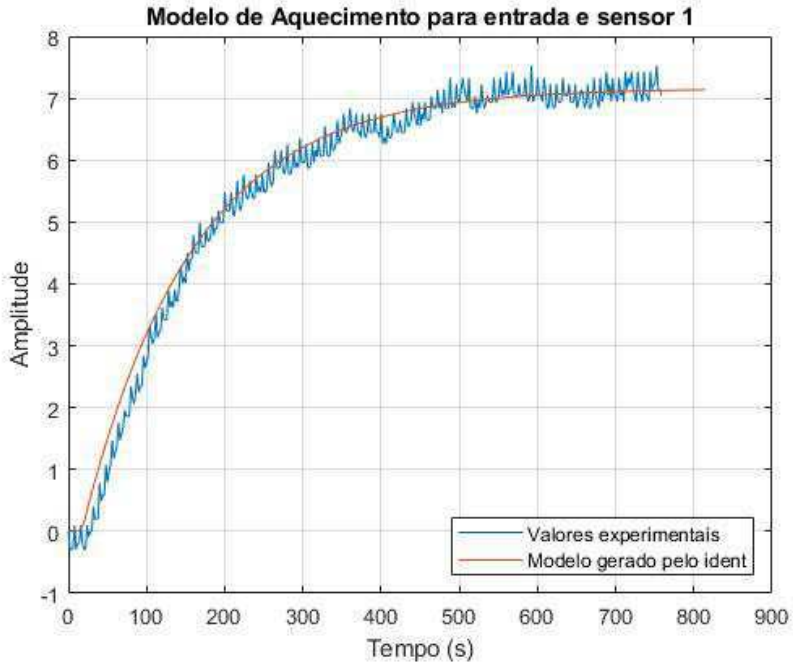

Figura 17: Modelo estimado para o Sistema 1 - Dinâmica de Aquecimento

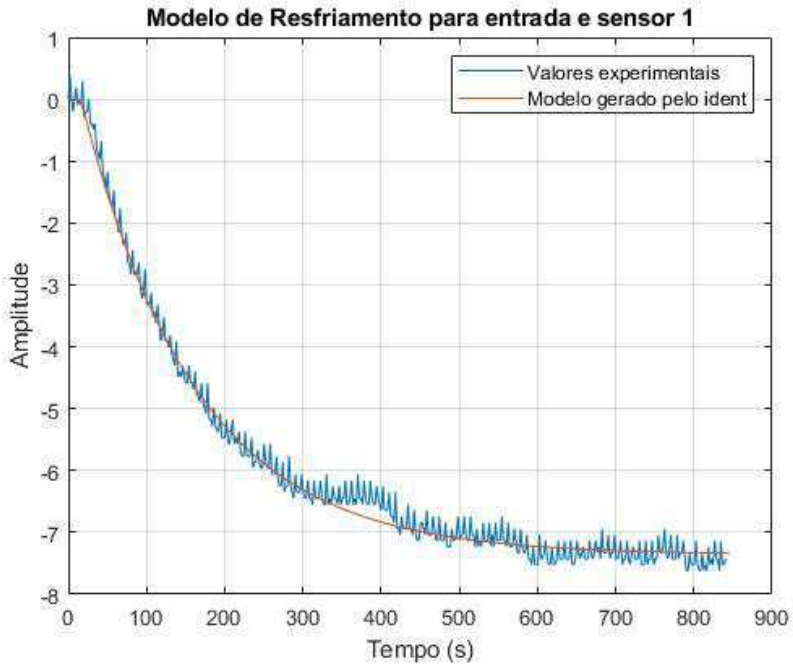

Figura 18: Modelo estimado para o Sistema 1 - Dinâmica de Resfriamento

Para o Sensor 2 foram estimados os modelos de primeira ordem com atraso das Equações 33 e 34 e o modelo médio está na Equação 35. Os modelos podem ser vistos nas Figuras 19 e 20.

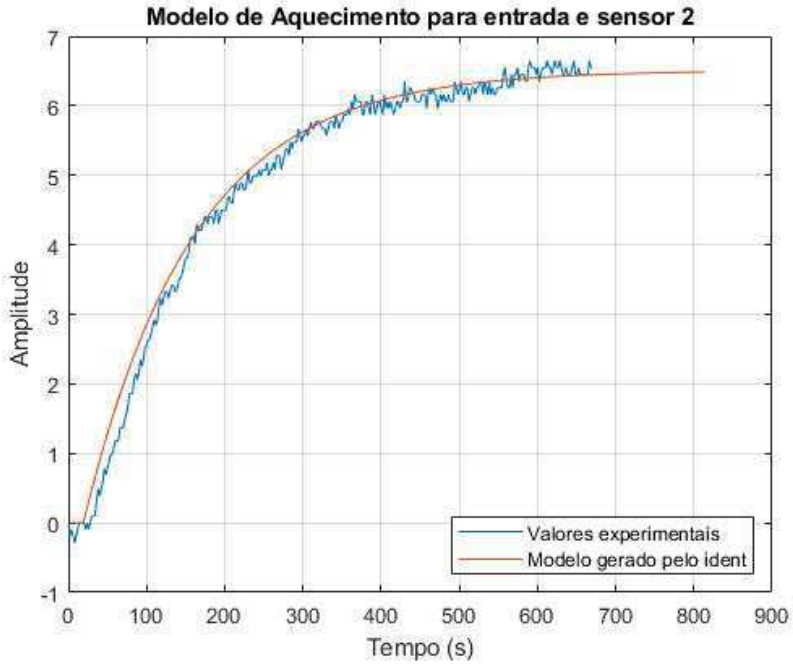

Figura 19: Modelo estimado para o Sistema 2 - Dinâmica de Aquecimento

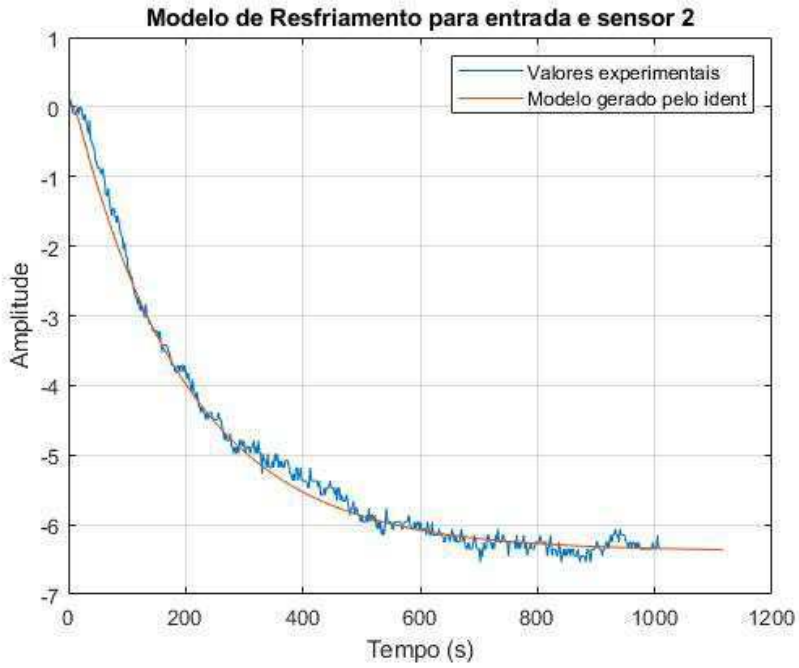

Figura 20: Modelo estimado para o Sistema 2 - Dinâmica de Resfriamento

$$G_{aquecimento}(s) = \frac{65.08}{140.3s + 1} e^{-19.2s} \tag{33}$$

$$G_{resfriamento}(s) = \frac{63.78}{192.3 + 1} e^{-11.9s} \tag{34}$$

$$G_{medio2}(s) = \frac{64.43}{166.3 + 1} e^{-15.5s} \tag{35}$$

3.2.2 Método dos Mínimos Quadrados

O método dos mínimos quadrados consiste em encontrar um melhor ajuste para um conjunto de dados tentando minimizar a soma dos quadrados das diferenças entre o valor estimado e os dados observados.

Para o Sensor 1 foram estimados os modelos de primeira ordem com atraso das Equações 36 e 37 e o modelo médio está na Equação 38. Os modelos podem ser vistos nas Figuras 21 e 22.

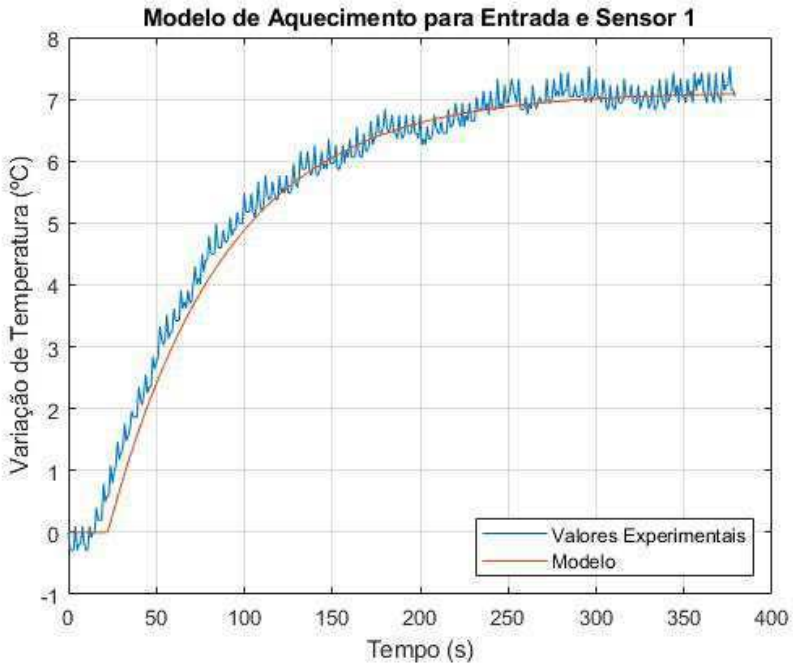

Figura 21: Modelo estimado para o Sistema 1 - Dinâmica de Aquecimento

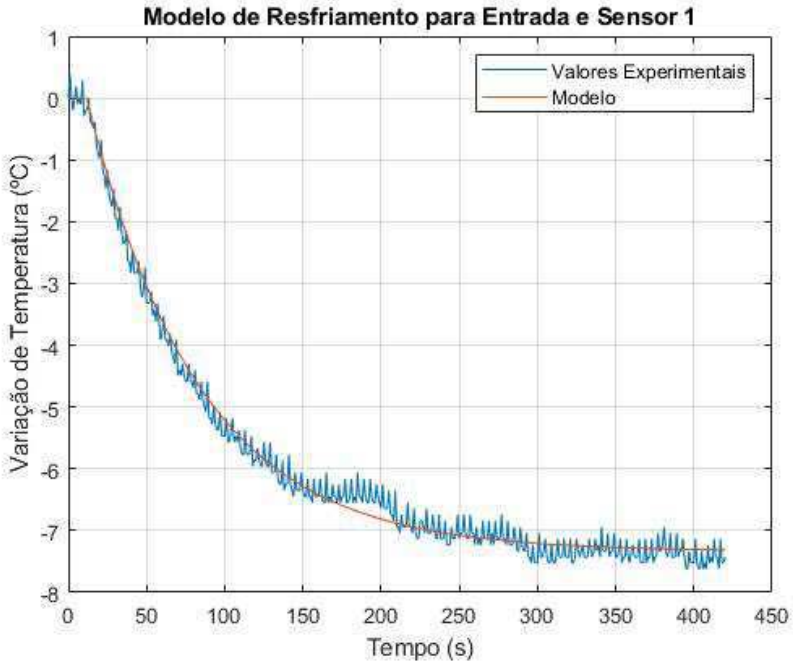

Figura 22: Modelo estimado para o Sistema 1 - Dinâmica de Resfriamento

$$G_{aquecimento}(s) = \frac{71.22}{67.05s + 1} e^{-17.4s} \quad (36)$$

$$G_{resfriamento}(s) = \frac{73.36}{71.26s + 1} e^{-10.7s} \quad (37)$$

$$G_{medio1}(s) = \frac{72.29}{69.155 + 1} e^{-14.05s} \quad (38)$$

Para o Sensor 2 foram estimados os modelos de primeira ordem com atraso das Equações 39 e 40 e o modelo médio está na Equação 41. Os modelos podem ser vistos nas Figuras 23 e 24.

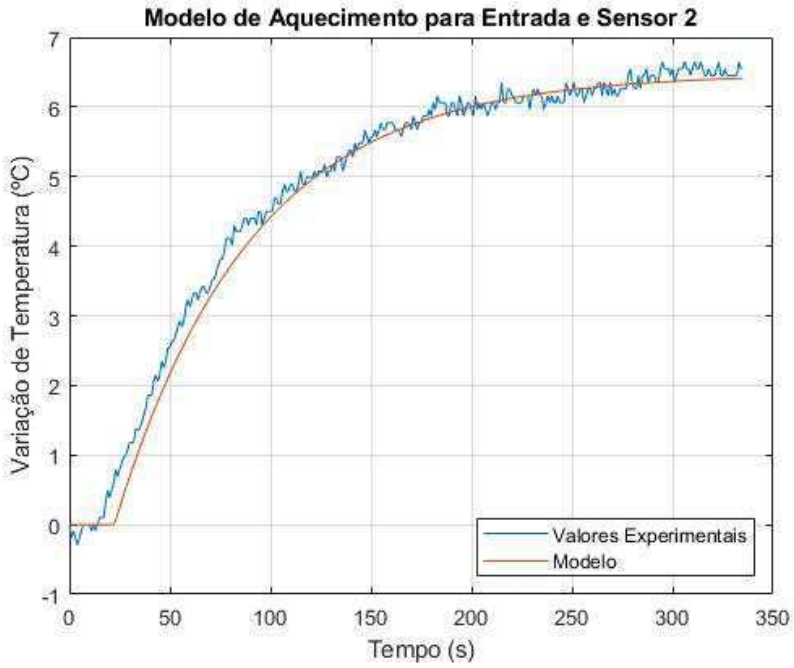

Figura 23: Modelo estimado para o Sistema 2 - Dinâmica de Aquecimento

$$G_{aquecimento}(s) = \frac{64.76}{67.79s + 1} e^{-17.4s} \quad (39)$$

$$G_{resfriamento}(s) = \frac{73.36}{71.26s + 1} e^{-10.7s} \quad (40)$$

$$G_{medio1}(s) = \frac{69.06}{69.525 + 1} e^{-14.5s} \quad (41)$$

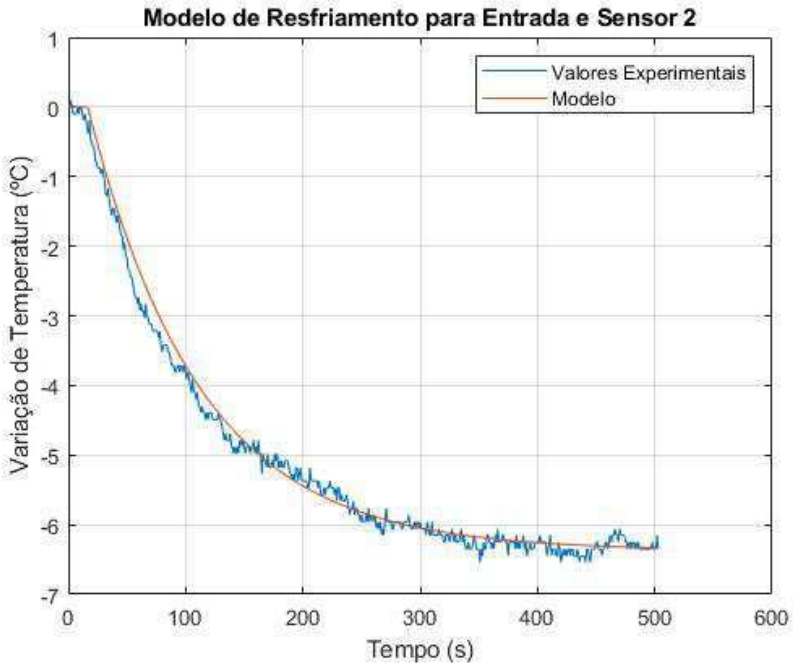

Figura 24: Modelo estimado para o Sistema 2 - Dinâmica de Resfriamento

3.3 Cálculo dos Parâmetros dos Controladores

Para o cálculo dos parâmetros dos controladores foram usados os modelos médios calculados pelo método de identificação de sistemas do MATLAB de cada sensor, mostrados nas Equações 32 e 35. Na seção 4.1 será explicado porque a escolha desse modelo. Para estes modelos, temos os parâmetros mostrados na Tabela 5.

O cálculo foi realizado conforme descrito no Seção 2.5 e os valores podem ser vistos nas subseções a seguir.

Tabela 5: Parâmetros dos Modelos Médios para o Cálculo dos Controladores

Sensor 1	Sensor 2
$K = 72.61$	K_64.43
$T = 143$	$T = 166.32$
$L = 16.524$	$L = 15.5$

Pelo Método Ziegler-Nichols

Usando o Método Ziegler-Nichols foram calculados os controladores PI para os dois sensores. Os valores podem ser vistos na Tabela 6.

Tabela 6: Parâmetros pelo método Ziegler-Nichols

Sensor 1	Sensor 2
$K_p = 0.1073$	$K_p = 0.1494$
$T_i = 54.53 \ \rightarrow \ K_i = 0.00197$	$T_i = 51.302 \ \rightarrow \ K_i = 0.0029$

Pelo Método SIMC

Usando o Método SIMC foram calculados os controladores PI para os dois sensores. Os valores podem ser vistos na Tabela 7.

Tabela 7: Parâmetros pelo método SIMC

Sensor 1	Sensor 2
$K_p = 0.006803$	$K_p = 0.010034$
$T_i = 70.086 \rightarrow K_i = 0.000097$	$T_i = 66.184 \rightarrow K_i = 0.000152$

Pelo Método CHR

Usando o Método CHR foram calculados os controladores PI para um sobressinal de 0% para os dois sensores. Os valores podem ser vistos na Tabela 8.

Tabela 8: Parâmetros pelo método CHR com 0% de sobressinal

Sensor 1	Sensor 2
$K_p = 0.041715$	$K_p = 0.05829$
$T_i = 171.6 \ \rightarrow \ K_i = 0.000243$	$T_i = 199.584 \ \rightarrow \ K_i = 0.00029$

4 Resultados e Discussões

Neste capítulo são apresentados os resultados obtidos pelas atividades experimentais e cálculos desenvolvidos ao longo deste trabalho. Nas seções seguintes é descrito como foi escolhido o modelo usado, o sistema em malha fechada com os controladores calculados e uma análise do desempenho de cada um dos controladores usados.

4.1 Identificação do Sistema

Para a identificação do sistema em questão, foi usado o método dos mínimos quadrados e a *Toolbox* de identificação do MATLAB. Para a escolha do modelo a ser utilizado foi feita uma comparação importando o modelo G, gerado pelo método dos mínimos quadrados pra o *ident* e ambos os modelos foram plotados com o valores experimentais, como pode ser visto nas Figuras 25 e 26.

Os modelos gerados pela *Toolbox* são nomeados por P1D e os estimados pelos mínimos quadrados por G. O valores experimentais são apresentados na cor preta.

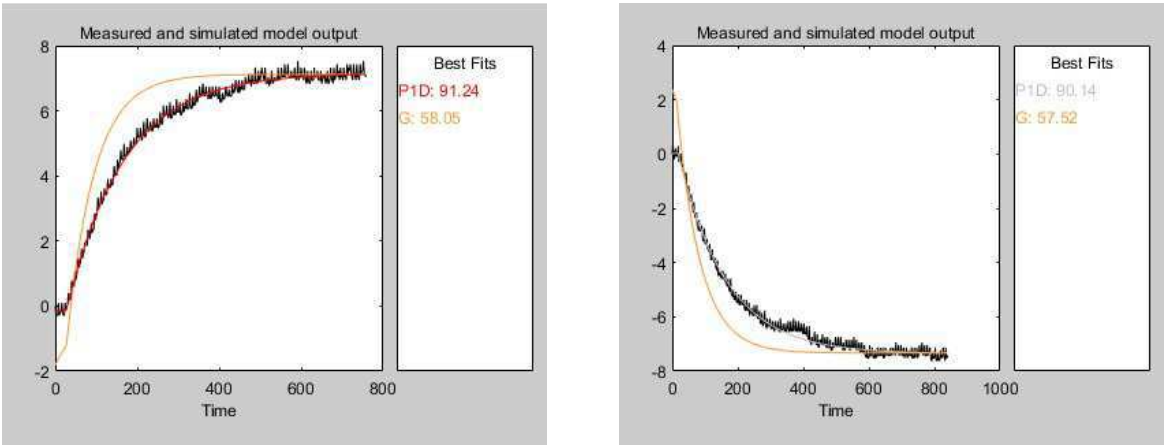

Figura 25: Modelos estimados para o Sensor 1

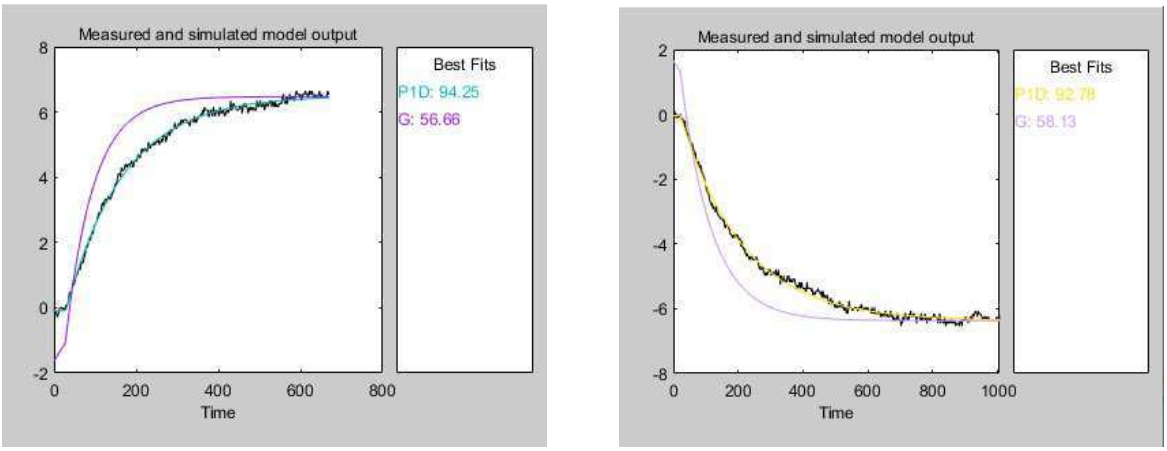

Figura 26: Modelos estimados para o Sensor 2

É nítido que os modelos estimados pelo *System Identification* se aproximam mais do sistema a ser controlado, sendo mais similares que os estimados pelos mínimos quadrados cerca de mais de 30%. Por esse motivo os modelos gerados pelo *ident* foram usados para o cálculo dos controladores na seção 3.3.

4.2 Análise de Desempenho dos Controladores

Para fazer a análise de desempenho dos controladores foram usados os índices descritos na seção 2.6. A partir dos controladores calculados, foram usadas funções do MATLAB para encontrar os valores desses índices, assim como para plotar a resposta ao degrau vista nas Figuras 27 e 28.

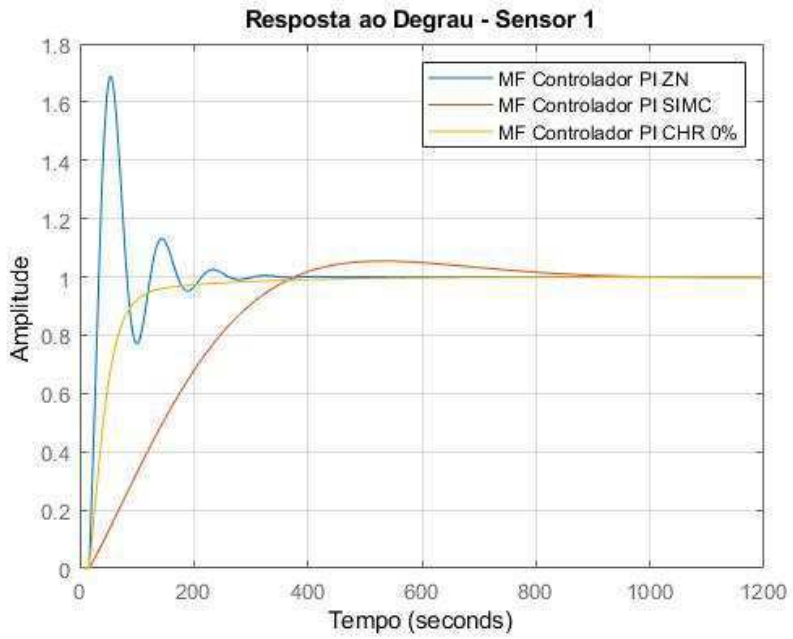

Figura 27: Resposta ao Degrau do Sistema em Malha Fechada para os Controladores Calculados - Sensor 1

Tabela 9: Parâmetros de desempenho dos controladores calculados para o Sensor 1

Parâmetros de desempenho	GM	PM	Tr	Ts	Mp
PI ZN	1.5748 dB	25.3430°	13.4325	243.3466	68.7599
PI SIMC	25.8884 dB	66.8536°	253.2410	772.3566	5.4232
PI CHR	4.5271 dB	73.1862°	68.6380	245.6969	0

Para os controladores calculados para o sistema 1, é possível ver na Tabela 10 e na Figura 28, que o PI SIMC tem um sobressinal de 5% e também possui a margem ganho mais alta dos 3 métodos calculados. Porém, apresenta tempos de subida e de

acomodação muito altos, indicando que o sistema demora bastante para chegar ao regime, sendo este o controlador com o pior desempenho.

O PI ZN possui os tempos de subida e de acomodação menores dos 3 métodos, porém tem um sobressinal maior que 68% e a margem de ganho e de fase menor dos 3 controladores, indicando uma maior variação do sistema devido a pertubações. Em comparação, o PI CHR possui tempos de subida e de acomodação pouco maiores que ele, mas tem margem de ganho e de fase melhores e um sobressinal de 0%, possuindo assim o melhor desempenho dentre os três controladores apresentados.

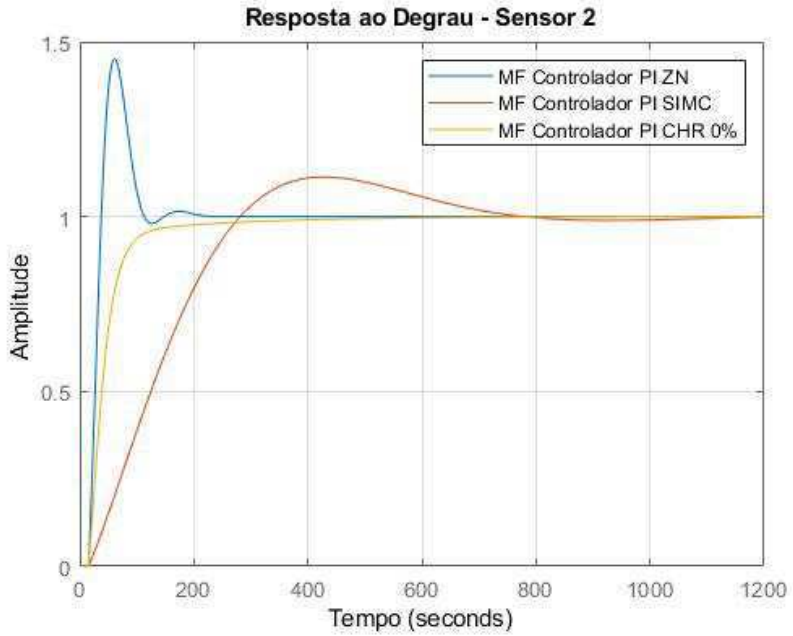

Figura 28: Resposta ao Degrau do Sistema em Malha Fechada para os Controladores Calculados - Sensor 2

Tabela 10: Parâmetros de desempenho dos controladores calculados para o Sensor 2

Parâmetros de desempenho	GM	PM	Tr	Ts	Mp
PI ZN	2.1904 dB	35.6304°	17.3568	108.0860	45.1168
PI SIMC	24.1674 dB	59.2702°	196.6033	696.1143	11.3403
PI CHR	4.5040 dB	72.5162°	62.7701	231.6964	0

Para os controladores calculados para o sistema 2, é possível ver na Tabela 10 e na Figura 28, que o PI SIMC tem um sobressinal de 11% e também possui a margem ganho mais alta dos 3 métodos calculados. Porém, apresenta tempos de subida e de acomodação muito altos, indicando que o sistema demora bastante para chegar ao regime, sendo este o controlador com o pior desempenho.

O PI ZN possui os tempos de subida e de acomodação menores dos 3 métodos, porém tem um sobressinal maior que 45% e a margem de ganho e de fase menor dos 3 controladores, indicando uma maior variação do sistema devido a pertubações. Em comparação, o PI CHR possui tempos de subida e de acomodação pouco maiores que ele, mas tem margem de ganho e de fase melhores e um sobressinal de 0%, possuindo assim o melhor desempenho dentre os três controladores apresentados.

4.3 Teste dos Controladores

Para o teste dos controladores calculados foi realizado um ensaio experimental usando o módulo didático. Para isso, a seguinte dinâmica foi adotada: primeiro, foi escolhido o modo de operação de cada malha; para as medições com o sensor 1, o modo de operação 1 usado foi o MF. Para que não houvesse nenhuma pertubação da malha 2, o modo de operação 2 usado foi o MA, com $DutyCycle$=0.0. De modo equivalente, foi feito o ensaio para os testes dos controladores para a malha 2.

Em ambos os ensaios foi seguido o mesmo processo da primeira atividade experimental de aquisição de dados descrita na seção 3.1. O módulo foi excitado até chegar em 50°C (ponto de operação), quando o sistema estivesse em regime o valor de referência foi atualizado para 55°C e, para finalizar, retornou-se para 50°C. Esses dois últimos momentos, de aquecimento e resfriamento, serão utilizados para analisar a performance dos controladores, visto que este é o intervalo que se assumiu linearidade para então calcular o modelo utilizado para sintonizar os controladores.

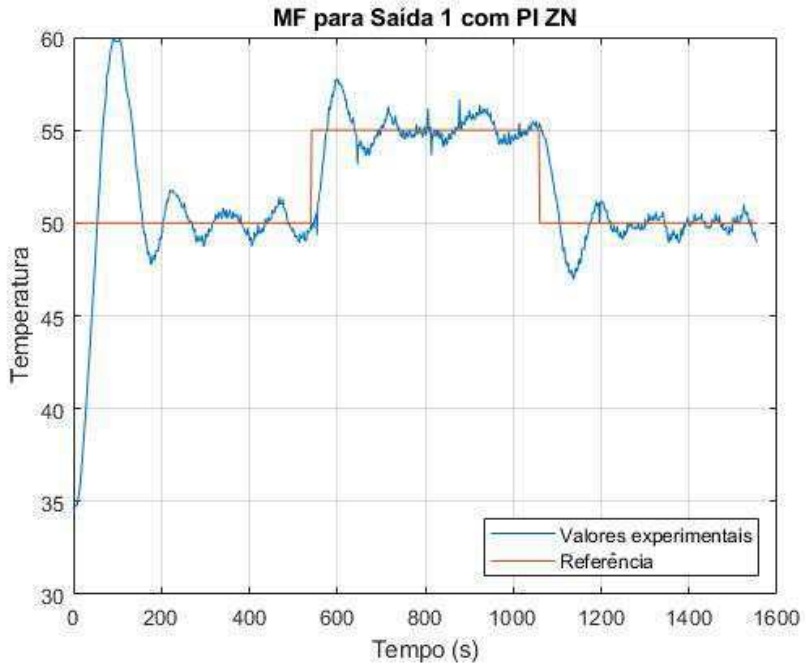

Figura 29: Experimento em Malha Fechada para o Sensor 1 com o controlador PI Ziegler-Nichols

O tempo de execução de cada teste variou de acordo com o tempo de acomodação de cada controlador, sendo o valor de referência atualizado quando passados o dobro do tempo de acomodação. Nas Figuras 29, 30 e 31 podem ser vistos o resultados dos experimentos em Malha Fechada para a Saída 1 para os 3 controladores.

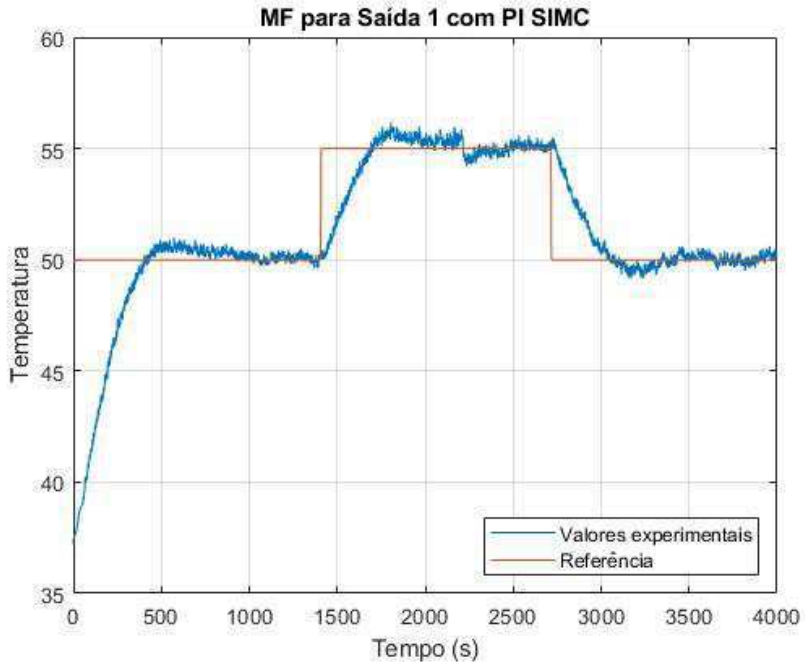

Figura 30: Experimento em Malha Fechada para o Sensor 1 com o controlador PI SIMC

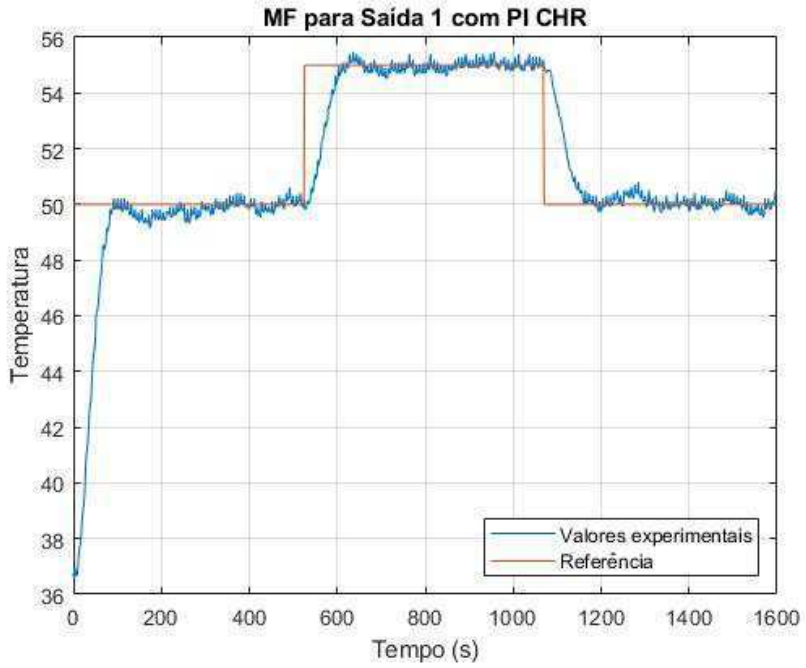

Figura 31: Experimento em Malha Fechada para o Sensor 1 com o controlador PI CHR

Para o sensor 2 foi realizada a mesma dinâmica do ensaio para o sensor 1. Nas Figuras 32, 33 e 34 podem ser vistos os resultados dos experimentos em Malha Fechada para a Saída 2 para os 3 controladores.

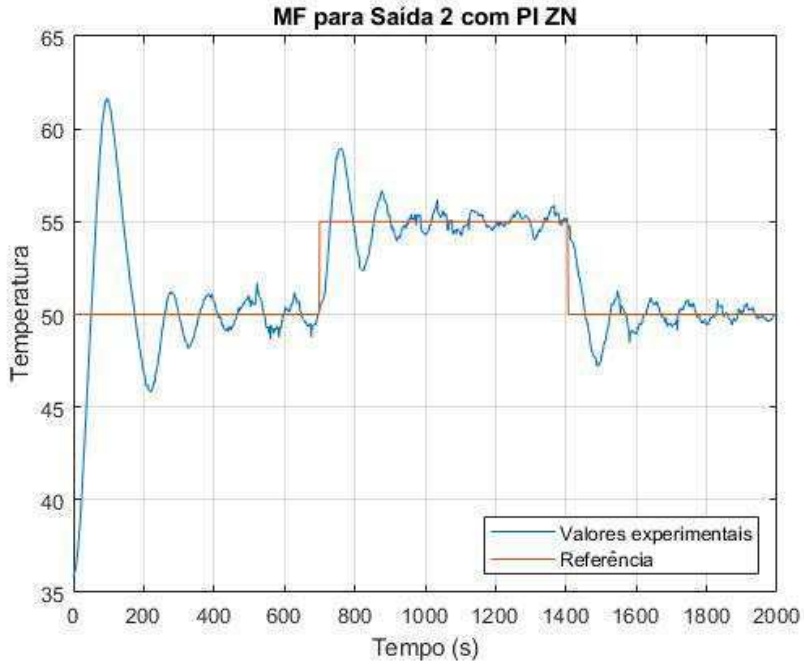

Figura 32: Experimento em Malha Fechada para o Sensor 2 com o controlador PI Ziegler-Nichols

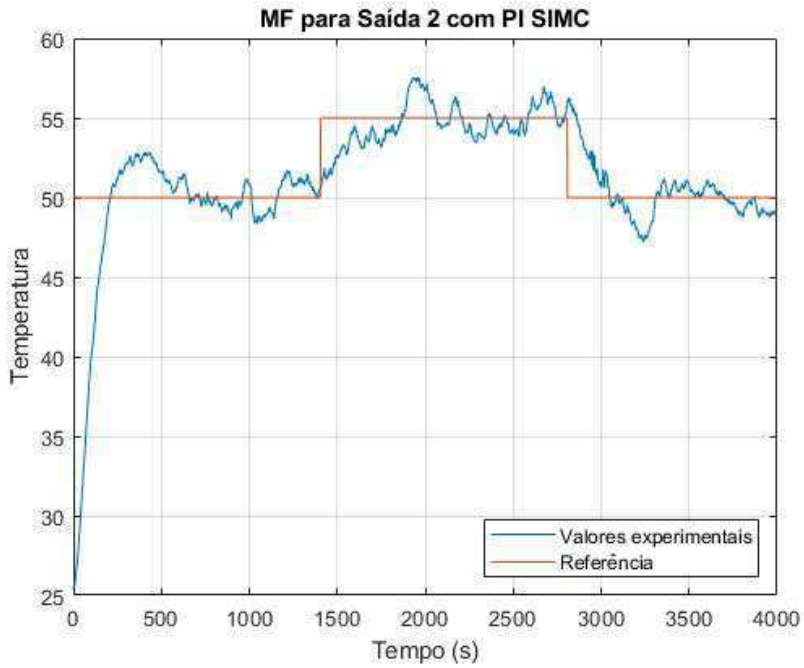

Figura 33: Experimento em Malha Fechada para o Sensor 2 com o controlador PI SIMC

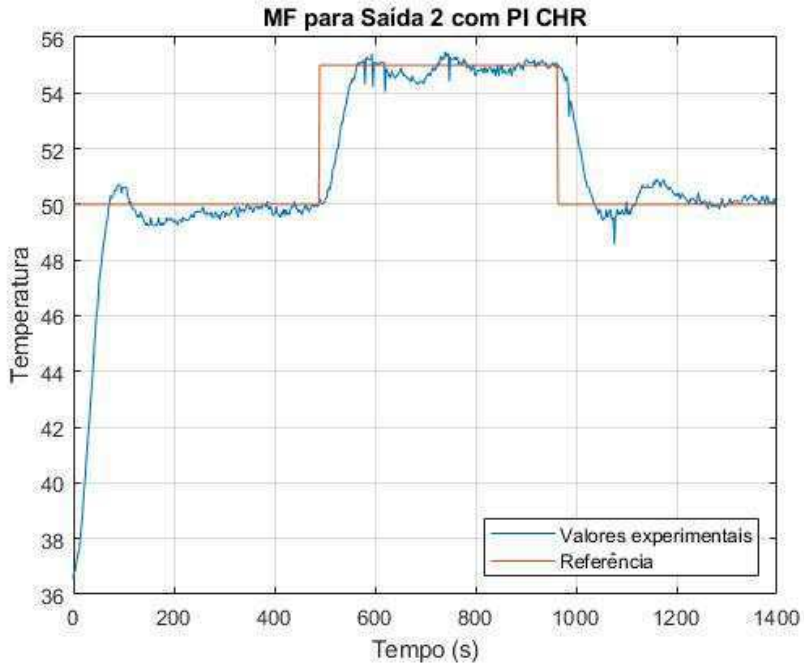

Figura 34: Experimento em Malha Fechada para o Sensor 2 com o controlador PI CHR

Nos ensaios realizados é possível observar o que já foi discutido na seção 4.3. Para os controladores PI SIMC, o sistema demora bastante para atingir o valor de referência, além disso, para a saída 2 o sistema possui muitas variações, mesmo após o tempo de acomodação.

Para os controladores PI ZN, o sistema atinge o valor de referência rapidamente, porém tem um sobressinal muito alto e sofre também muitas variações após o tempo de acomodação.

Para os controladores PI CHR, o valor de referência é atingido rapidamente, sofrendo poucas variações após o tempo de subida, sem sobressinal e com a margem de fase mais alta dos 3 controladores calculados, fazendo deste controlador o de melhor desempenho para as duas malhas estudadas.

5 Conclusões

O presente trabalho apresentou um estudo de identificação de sistemas e sintonia de controladores PID por meio da realização de experimentos didáticos, usando o módulo apresentado em Lima et. al.

Para a realização dos experimentos, foi utilizado o módulo citado e o código *PlataformaTermica.m* disponibilizado para realização dos experimentos em sala de aula. Para os ensaios em malha aberta, os valores dos ciclos de trabalho foram variados como descrito na seção 3.1. Depois do experimento, é gerado um arquivo *.mat* com os dados obtidos, esse arquivo foi usado para realizar a identificação do sistema usando o *ident* e o método dos mínimos quadrados. Após comparados os dois modelos obtidos com o valores do ensaio, viu-se que o *ident* estimou um modelo mais aproximado, por esse motivo foi usado para as demais atividades.

Em seguida, a partir do modelo obtido, foram sintonizados os controladores usando 3 métodos de sintonia: Ziegler-Nichols (ZN), SIMC e CHR. Por fim, foi realizado o ensaio em malha fechada usando os controladores calculados e então foi feita uma análise de desempenho de cada um deles. Foi observado, então, que o controlador calculado pelo método CHR obteve melhor desempenho para as duas malhas estudadas.

A partir dos resultados obtidos, foi observado que todos os controladores sintonizados tiveram um bom desempenho, sendo possível, com base nos critérios de desempenho e no teste em malha fechada, escolher o controlador mais adeqado para o controle do processo. Conclui-se então, que os objetivos do trabalho foram alcançados.

De modo geral, os controladores PID são largamente utilizados no controle de processos industriais, por esse motivo seu estudo é de alta importância para o engenheiro eletricista.

Referências

[1] LIMA, A.; BARROS, p.; ACIOLI, G. Módulo Didático para Ensino da Teoria de Controle, 2017.

[2] Model-Based Design. Disponível em <www.mathworks.com/solutions/model-based-design.html>. Acesso em 10 de Abril de 2019.

[3] Simulink Support Package for Arduino Hardware. 2016, The MathWorks, Inc., Natick, MA, United States.

[4] MORATORI, P. B. Por que utilizar jogos educativos no processo de ensino aprendizagem? UFRJ. Rio de Janeiro. 2003.

[5] Mendes, F. MODELAGEM E CONTROLE DE NÍVEL DE UM TANQUE DE ÁREA VARIÁVEL. João Monlevade, MG, 2017.

[6] Felix, T. R. Desenvolvimento de Experimentos para o Ensino de Controle Analógico. Campina Grande, PB, 2018.

[7] Lobo, F. M. Contribuições ao Ensino de Controle Usando MATLAB, Arduino e *Hardware* de Baixo Custo. UFES, Vitória, ES, 2017.

[8] OGATA, K. Engenharia de Controle Moderno, 4a edição, Prentice Hall, São Paulo, SP, Brasil, 2003.

[9] GARTSEEV, I.; LEE, L.; KROVI, N. A low-cost real-time mobile robot platform (ArEduBot) to support project-based learning in robotics mechatronics. 2011.

[10] SEGURA, F.; BARTOLUCCI, V.; ANDÚJAR, J. Hardware/Software Data Acquisition System for Real Time Cell Temperature Monitoring in Air-Cooled Polymer Electrolyte Fuel Cells. 2017.

[11] Mota, R. A. T. Projeto e Implementação de Plataforma Experimental para Laboratório de Controle Digital Baseada em Arduino. Campina Grande, Paraíba, 2015.

[12] Cheng, Hongtai. Establishing the Connection between Control Theory Education and Application: An Arduino Based Rapid Control Prototyping Approach. The Journal of Learning and Teaching, 2016.

[13] Bittencourt, M. C. Identificação de Sistemas Dinâmicos Lineares - Métodos Paramétricos e Não Paramétricos. Brasília, DF, 2007.

[14] Batista, L. C. Estudo Comparativo de Técnicas de Sintonia de Controladores PID para Sistemas de Primeira Ordem com Atraso. 2014.

[15] Saraiva, F. A. Métodos de Sintonia em Controladores PID. 2011.

[16] Laboratório de Controle Digital. Guia Experimental - Identificação de Sistemas, 2017.

A Instalação do *MATLAB/Simulink Support Package for Arduino*

Antes do início das atividades foi preciso realizar a instalação e configuração dos pacotes de suporte. Para isso foi usada a aba *Add-Ons* encontrada na tela principal do MATLAB e selecionada a opção *Get Hardware Support Packages*, como pode ser visto na Figura 35.

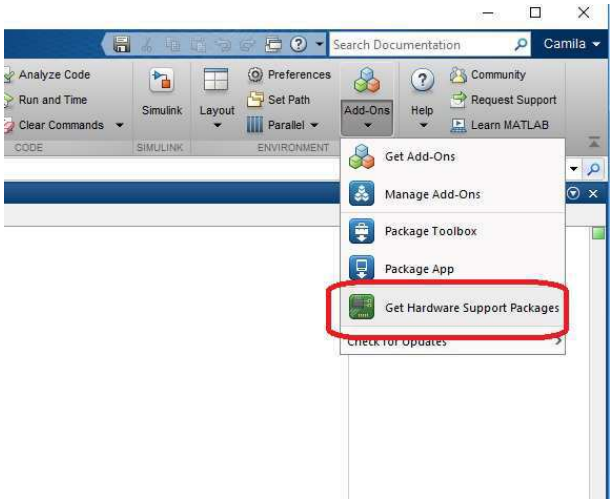

Figura 35: Instalção dos pacotes de suporte para arduino

Depois de selecionada a opção acima se tem acesso a tela mostrada na Figura 36, onde é possivel ver os dois pacotes que devem ser instalados (MATLAB e *Simulink Support Package for Arduino Hardware*).

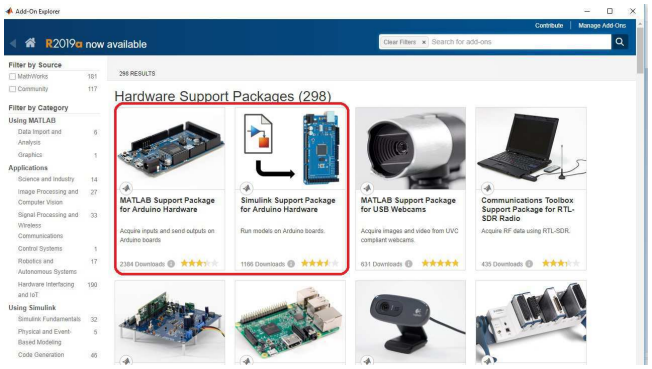

Figura 36: Pacotes de suporte para arduino

Ao selecionar a opção de instalação, o MATLAB irá pedir para entrar com um cadastro, que pode ser feito no site da `Mathworks.com`. Despois de instalados, é possível ter acesso e fazer a comunição do *software* com a plataforma através dos blocos mostrados na Figura 37, encontrados na biblioteca do *Simulink* na pasta *Simulink Support Package for Arduino Hardware*.

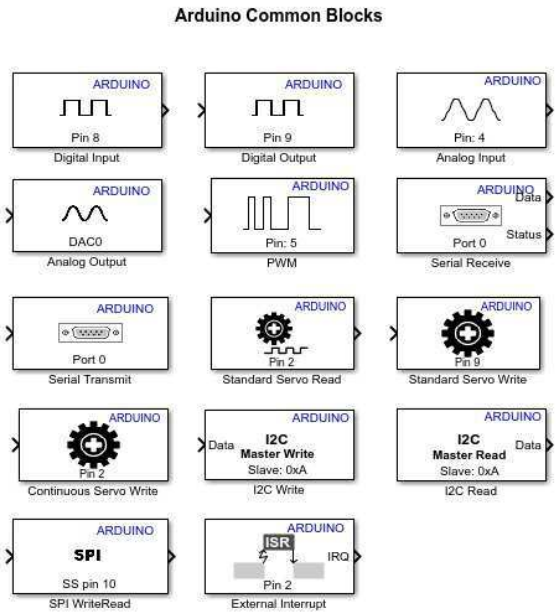

Figura 37: Blocos disponíveis pelo pacote de suporte para arduino

Mesmo com o pacote de suporte do *Simulink*, ainda não se tem acesso a muitas das funções que o arduino pode oferecer. Para aumentar o leque de possibilidades alguns exemplos de suportes que permitem ter acesso a outras funcionalidades da plataforma arduino no *Simulink* são a *Rensselaer Arduino Support Package Library (RASPLib)* e *Arduino Additional Sensors Library (DHT, LPS331)*. Para realizar a instalação dos dois é preciso seguir as etapas listadas a seguir:

- No prompt de comando do MATLAB digite "mex -setup", se o compilador já estiver instalado então pode-se seguir para o próximo item, se não,
  - Feche o MATLAB e o abra novamente clicando com o botão direito do mouse e escolhendo a opção 'rodar como administrador';
  - Abra a aba de *Add-Ons* como foi mostrado anteriormente;
  - Do lado esquerdo desmarque a opção '*Hardware Support Packages*' e busque por *MATLAB Support for the MinGW-w64 C/C++ Compiler* e realize a instalação do compilador.
- Em seguida, com o compilador instalado, desmarque a opção '*Hardware Support Packages*', busque por *Rensselaer Arduino Support Package Library (RASPLib)* ou *Arduino Additional Sensors Library (DHT, LPS331)* e realize a instalação.

B Configurações do Modo Externo

Uma das opções do *Simulink* para testar o modelo é usar o modo externo. Nesse modo o programa é compilado, enviado ao arduino e a simulação começa a ser executada com o *hardware*, permitindo visualizar os dados e alterar os parâmentos necessários em tempo real antes de gerar o código final.

Para usar esse modo é preciso realizar as seguintes cofigurações no *Simulink*. Na aba *Simulation – Model Configuration Parameters - Solver* foram marcadas as configurações mostradas na Figura 38.

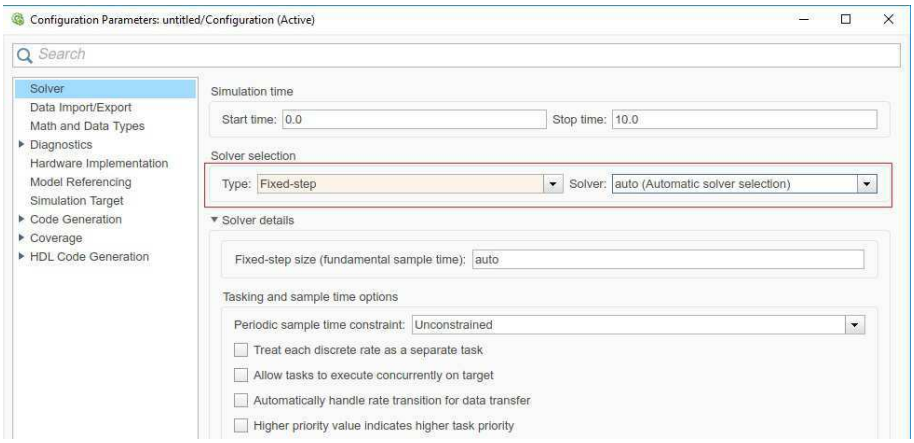

Figura 38: Configurações para usar o modo Externo

Depois das configurações, foram realizadas as alterações mostradas na Figura 39 na tela principal do *Simulink*. Depois de feitas as configurações descritas, o modelo pode ser executado.

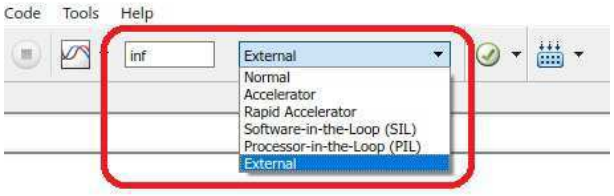

Figura 39: Modo Externo

C Configurações para Geração Automática de Código

Após a instalação dos pacotes necessários e da implementação do sistema, sendo este já validado, é possível realizar a geração de código na linguagem C/C++ e a transferência automática para o sistema embarcado utilizado. Para isto é utilizada a ferramenta *Simulink Coder*.

Para realizar a geração de código e a transferência para o microcontrolador, é necessário realizar o ajuste de alguns parâmetros, informando ao *software* a plataforma que está sendo utilizada. A Figura 40 mostra a janela de configuração dos parâmetros, que é encontrada no ambiente principal do *Simulink*, na aba *Simulation – Model Configuration Parameters - Code Generation Options*.

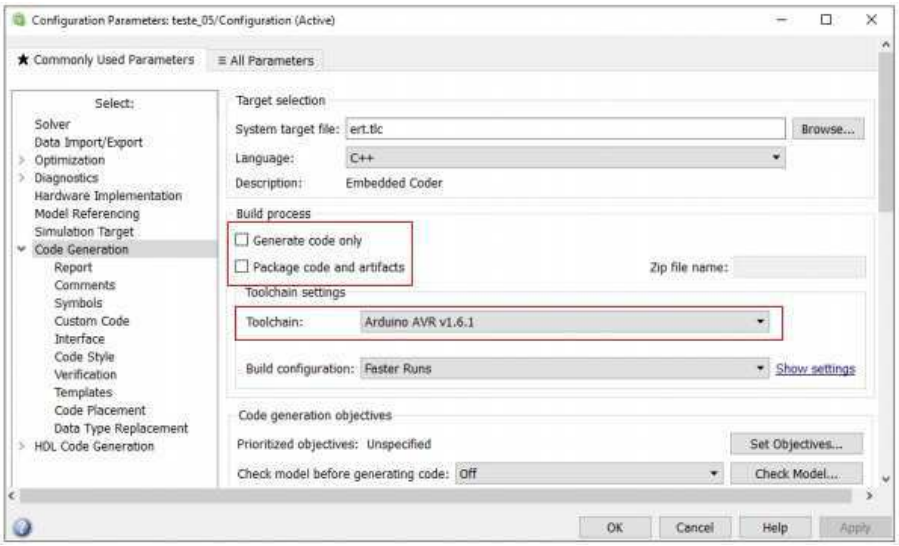

Figura 40: Janela de configuração dos parâmetros do *Simulink Coder*

Na janela mostrada na Figura 40 foram selecionadas as seguintes opções:

- na aba *Code generation* no campo *Build process* foram ativadas as opções *Generate Code only* e *Package Code and artifacts*, nessa opção é possível escolher o nome do arquivo compactado que será salvo o código gerado;
- ainda na aba *Code generation* no campo *Toolchain* foi selecionada a opção Arduino AVR v1.6.1.
- na aba *Hardware Implementation* - *Hardware Board* foi selecionada a opção Arduino Uno, indicando a placa que receberá o código gerado.

Figura 41: Indicação para a geração automática de código

Após as configurações descritas anteriormente, o código é gerado acionando o botão indicado na Figura 41 e em seguinda *Deploy to Hardware*, encontrado na tela principal de programação do Simulink. Após a geração de código e transferência para o microcontrolador, são geradas algumas informações e relatórios que permitem a visualização dos arquivos e códigos gerados. Um documento compactado contendo o código pode ser encontrado no mesmo diretório onde o projeto está salvo.