

**UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
PROJETO DE GRADUAÇÃO**



MATHEUS RANGEL PIMENTEL

**INTEGRAÇÃO DE SISTEMAS SCADA A SERVIÇOS DE
COMPUTAÇÃO EM NUVEM PARA
MONITORAMENTO DE DESEMPENHO DE PLANTAS
INDUSTRIAIS**

VITÓRIA-ES

JULHO/2022

Matheus Rangel Pimentel

**INTEGRAÇÃO DE SISTEMAS SCADA A SERVIÇOS DE
COMPUTAÇÃO EM NUVEM PARA
MONITORAMENTO DE DESEMPENHO DE PLANTAS
INDUSTRIAIS**

Parte manuscrita do Projeto de Graduação do aluno Matheus Rangel Pimentel, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

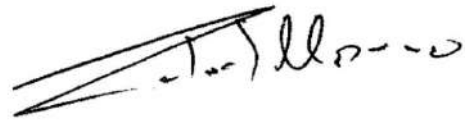
Vitória-ES

Julho/2022

Matheus Rangel Pimentel

INTEGRAÇÃO DE SISTEMAS SCADA A SERVIÇOS DE COMPUTAÇÃO EM NUVEM PARA MONITORAMENTO DE DESEMPENHO DE PLANTAS INDUSTRIAIS

Parte manuscrita do Projeto de Graduação do aluno Matheus Rangel Pimentel, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.



Prof. Dr. Celso Jose Munaro
Universidade Federal do Espírito Santo
Orientador



Prof. Dr. Thomas Walter Rauber
DI/Universidade Federal do Espírito Santo
Examinador



Msc. Gercilio Carlos Zuqui Junior
Vale
Examinador

Vitória-ES

Julho/2022

AGRADECIMENTOS

Gostaria de agradecer a todas as pessoas especiais listadas abaixo, que estão presentes em minha vida.

A meus pais, Manoel e Leidy Pimentel, pelo apoio emocional e financeiro que possibilitaram a conclusão da minha graduação.

A minha namorada, Laís Neves, pelo companheirismo e pelo apoio emocional entregues a mim nessa fase.

A meu orientador, Celso Munaro, pelo auxílio e motivação nesses dois anos de caminhada juntos.

A meus parentes, pelos momentos de felicidade e tranquilidade que vivemos.

A meus amigos da UFES, que tornaram essa caminhada mais leve e descontraída.

A vocês, a minha sincera gratidão.

RESUMO

Em grande parte dos ambientes industriais de médio e grande porte, existem inúmeras malhas de controle responsáveis por controlar os processos. Entretanto, estudos indicam que a maioria desses controladores, após inicialmente sintonizados, não são mais alterados. Devido a mudanças na dinâmica do processo e falhas estruturais, o desempenho de muitos deles se torna pior com o passar do tempo. Assim, ferramentas de diagnóstico e melhora do desempenho se tornam indispensáveis do ponto de vista econômico e de produção. Este trabalho utiliza um serviço de computação em nuvem, integrado a sistemas SCADA, para o monitoramento de malhas de controle. Os dados são enviados à nuvem somente durante mudanças de referência, evitando assim o envio de dados ininterrupto à nuvem. Para o monitoramento, testes de hipótese são realizados com amostras da Integral do Erro Absoluto (IAE) para avaliar desempenho. Além disso, uma metodologia para a resintonia de controladores para a recuperação do desempenho proposta por Mohieddine Jelali foi utilizada. Comparações com outras metodologias de monitoramento de desempenho de malhas de controle foram também realizadas. Os resultados alcançados mostram que tanto a integração de sistemas SCADA a nuvem quanto à proposta de monitoramento de desempenho foram bem sucedidas e são aptas à generalização para outros processos industriais.

Palavras-chave: Monitoramento de Desempenho de Malhas de Controle; CLPM; Internet das Coisas; Computação em Nuvem.

ABSTRACT

In most medium and large industrial environments, there are numerous control loops responsible for controlling the processes. However, studies indicate that most of these controllers, after initially tuned, are no longer changed. Due to changes in process dynamics and structural faults, many of them perform worse over time. Thus, performance assessment and improvement tools become indispensable from an economic and production point of view. This work uses a cloud computing service, integrated with SCADA systems, to monitor control loops. Data is sent to the cloud only during reference changes, thus avoiding uninterrupted data sending to the cloud. For monitoring, hypothesis tests are performed with Integral of Absolute Error (IAE) samples to verify the quality of performance. In addition, a solution for controllers retuning for performance recovery proposed by Mohieddine Jelali was used. Comparisons with other control loop performance monitoring methodologies were also performed. The results achieved suggest that both the integration of SCADA systems to the cloud and the performance monitoring proposal were successful and are able to be generalized to other industrial processes.

Keywords: Control Loop Performance Monitoring; CLPM; Internet of Things; Cloud Computing.

LISTA DE FIGURAS

Figura 1 – Estrutura geral da comunicação utilizando o protocolo MQTT.	16
Figura 2 – Modo de transmissão de dados para (a) QoS 0; (b) QoS 1; (c) QoS 2. . .	17
Figura 3 – Serviços oferecidos pela computação em nuvem.	19
Figura 4 – Diagrama de blocos geral de um sistema em malha fechada.	21
Figura 5 – Diagrama de blocos do sistema.	27
Figura 6 – Estrutura padrão utilizada pelo método de sintonia IMC.	27
Figura 7 – Exemplos de curvas de ganho estáticas. (A), (B) e (C): problema de controle servo. (D) e (E): mistura entre os dois problemas. (F) problema de controle regulatório.	30
Figura 8 – Dados de processo de uma planta com atrito na válvula de controle, antes e depois da nova sintonia.	31
Figura 9 – Comparação entre as relações empíricas e analíticas com modelos numericamente determinados.	33
Figura 10 – Planta piloto LSCA-CPID.	36
Figura 11 – Fluxograma do processo.	37
Figura 12 – Diagrama de controle do processo.	38
Figura 13 – Malhas de controle do processo programadas em linguagem Ladder. . .	39
Figura 14 – Estrutura geral da solução proposta.	40
Figura 15 – PDF de população de médias com distribuição T de Student ($\alpha = 5\%$ e $N = 5$)	42
Figura 16 – Fluxograma da solução proposta.	45
Figura 17 – Tela de operação criada.	47
Figura 18 – Parâmetros dos modelos FOPDT obtidos a partir de respostas ao degrau em malha aberta.	50
Figura 19 – Respostas ao degrau em malha aberta: dados vs. modelo obtido.	51
Figura 20 – Parâmetros dos modelos em malha aberta vs. malha fechada para diferentes valores de λ	52
Figura 21 – Respostas ao degrau do modelo obtido em malha aberta vs. malha fechada para diferentes valores de λ	53
Figura 22 – Treinamento e teste do monitoramento de desempenho para diferentes valores de λ	54
Figura 23 – Monitoramento de desempenho para situação de falha com sintonia realizada para $\lambda = 4s$	55
Figura 24 – Monitoramento de desempenho para situação de falha com sintonia realizada para $\lambda = 8s$	55

Figura 25 – Monitoramento de desempenho para situação de falha com sintonia realizada para $\lambda = 12s$	56
Figura 26 – Monitoramento de desempenho para situação de falha com sintonia realizada para $\lambda = 8s$ e $\theta_a = 6s$	58
Figura 27 – Parâmetros dos modelos obtidos em normalidade e falha.	59
Figura 28 – Respostas ao degrau médias para diferentes valores de λ : normalidade vs. falha.	60
Figura 29 – Resintonia para $\lambda = 4s$	60
Figura 30 – Resintonia para $\lambda = 8s$	61
Figura 31 – Resintonia para $\lambda = 12s$. Primeira tentativa: $K_p = 1, 2K_p$	61
Figura 32 – Tela de operação	62

LISTA DE TABELAS

Tabela 1 – Nível de tensão na bateria por QoS.	17
Tabela 2 – Controlador PI para diferentes valores de λ	51
Tabela 3 – IAE_L para os 3 valores de λ	53
Tabela 4 – IAE_s para os 3 valores de λ	54
Tabela 5 – IAE_s para os 3 valores de λ	57

LISTA DE ABREVIATURAS E SIGLAS

AWS	<i>Amazon Web Services</i>
CLPA	<i>Control Loop Performance Assessment</i>
CLPM	<i>Control Loop Performance Monitoring</i>
CLP	Controlador Lógico Programável
CPID	Centro de Pesquisa, Inovação e Desenvolvimento
CPU	<i>Central Processing Unit</i>
DCS	<i>Distributed Control Systems</i>
DDS	<i>Data Distribution Service</i>
FOPDT	<i>First-Order Plus Deadtime</i>
IaaS	<i>Infrastructure as a Service</i>
IAE	<i>Integral of Absolute Error</i>
IBM	<i>International Business Machine Corporation</i>
IIoT	<i>Industrial Internet of Things</i>
IMC	<i>Internal Model Control</i>
IoT	<i>Internet of Things</i>
KET	<i>Key Enabling Technologies</i>
LSCA	Laboratório de Sistemas de Controle e Automação
LTI	<i>Linear Time-Invariant</i>
MQTT	<i>Message Queuing Telemetry Transport</i>
OASIS	<i>Organization for the Advancement of Structured Information Standards</i>
OPC	<i>Open Platform Communication</i>
PaaS	<i>Platform as a Service</i>
PubSub	<i>Publisher-Subscriber</i>

PID	<i>Proporcional Integral Derivativo</i>
QoS	<i>Quality of Service</i>
ROS	<i>Robot Operating System</i>
RTT	<i>Round-Trip Time</i>
SaaS	<i>Software as a Service</i>
SCADA	<i>Supervisory Control and Data Acquisition</i>
SISO	<i>Single-Input Single-Output</i>
SSH	<i>Secure Shell</i>
UA	Unified Architecture
UFES	Universidade Federal do Espírito Santo

SUMÁRIO

1	INTRODUÇÃO	13
1.1	Apresentação	13
1.2	Objetivos	14
1.2.1	Objetivo Geral	14
1.2.2	Objetivos Específicos	14
1.3	Estrutura do Texto	14
2	EMBASAMENTO TEÓRICO	15
2.1	Protocolo MQTT	15
2.1.1	Princípio de funcionamento	15
2.1.2	Qualidades de serviço	16
2.1.3	Comparação de desempenho	17
2.1.3.1	Brokers	17
2.1.3.2	MQTT versus OPC-UA	18
2.2	Computação em Nuvem	18
2.2.1	Serviços de Nuvem	18
2.2.2	Contexto Industrial	19
2.3	Estimação de modelos em malha fechada	20
2.3.1	Método dos Mínimos Quadrados	22
2.3.2	Índices de avaliação do modelo	24
2.3.3	Obtenção dos parâmetros do modelo contínuo a partir do modelo discreto	24
2.4	Método de sintonia IMC	26
2.5	Avaliação de desempenho de malhas de controle	29
2.5.0.1	Índice de Harris	31
2.5.0.2	Avaliação baseada em dados de resposta a mudanças de <i>setpoint</i>	32
2.6	Trabalhos Relacionados	34
3	METODOLOGIA E ETAPAS DE DESENVOLVIMENTO	36
3.1	Planta Piloto	36
3.2	Conexão e Programação do CLP	38
3.2.1	Conexão dos instrumentos industriais ao CLP	38
3.2.2	Programação dos algoritmos de controle	38
3.2.3	Implementação do MQTT	39
3.3	Monitoramento de desempenho	41
3.4	Resintonia	44
3.5	Monitoramento local e em nuvem	45

3.5.1	Mastertool	46
3.5.2	Nuvem	46
4	RESULTADOS E DISCUSSÕES	50
4.1	Modelagem e Sintonia	50
4.2	Monitoramento de Desempenho	52
4.2.1	Treinamento e teste em normalidade	52
4.2.2	Teste em falha	54
4.3	Resintonia	58
4.4	Interface para operação local e em nuvem	62
5	CONCLUSÃO	64
	REFERÊNCIAS	66

1 INTRODUÇÃO

1.1 Apresentação

Os controladores de malhas de processos industriais são projetados para garantir um desempenho mínimo, definido por diferentes índices (GRELEWICZ et al., 2021). Entretanto, esse desempenho é afetado quando a dinâmica do processo é alterada, devido a alterações no ponto de operação, degradação nos sensores e atuadores ou presença de distúrbios (JELALI, 2012); ou quando há uma má sintonia do controlador, o que, segundo Bauer et al. (2016) é um dos maiores causadores de falhas nas malhas de controle. Portanto, um monitoramento periódico de desempenho com possível atualização do controlador torna-se necessário para a preservação da eficiência do processo (STARR; PETERSEN; BAUER, 2016).

No entanto, boa parte dos algoritmos envolvidos nessas tarefas são computacionalmente custosos (BAUER et al., 2016) e demandam operações matemáticas disponíveis apenas em linguagens de programação de alto nível, funcionalidades não suportadas por controladores industriais.

Nesse quesito, a computação em nuvem tem características atrativas: é facilmente acessível, distribuída e escalável, além de aliar um alto poder computacional, habilidade de armazenar e analisar grandes bancos de dados e a possibilidade de um sistema de monitoramento centralizado (GONG et al., 2010). Todavia, embora tenha muitos atributos positivos, há preocupações da indústria quanto a seu uso; cibersegurança, confiabilidade, latência, topologia de rede e a operação em tempo real são alguns desses aspectos críticos (PUTHAL et al., 2015).

Baseado na experiência de Capaci e Scali (2020) na área de engenharia de controle e Monitoramento de Desempenho de Malhas de Controle (CLPM, do inglês *Control Loop Performance Monitoring*), existe uma grande dificuldade de estabelecimento do processo de monitoramento remoto e centralização. Isto se dá pela dificuldade em se transferir habilidades desenvolvidas durante anos por operadores em plantas individuais. Ademais, segundo os autores, outro fator que influencia nessa relutância é o fato de dados de processo geralmente serem confidenciais, e as empresas não desejam que esses sejam movidos de um computador local para a nuvem, externa à rede local, devido ao surgimento de problemas de segurança e confiabilidade. Sendo assim, a integração de Sistemas de Supervisão e Aquisição de Dados (SCADA, do inglês *Supervisory Control And Data Acquisition*) a serviços de computação em nuvem de forma racional, com o envio apenas de dados estritamente necessários para diagnóstico, é uma forma de minimizar mas não sanar o problema. Logo,

uma proposta para evitar tais tipos de problema é a utilização de computação em nuvem interna à empresa, o que mantém os dados confidenciais à rede da mesma. Esse serviço pode ser realizado pela própria empresa ou por via de contratação de terceiros.

1.2 Objetivos

1.2.1 Objetivo Geral

Este trabalho tem como objetivo integrar de forma racional sistemas SCADA a serviços de computação em nuvem para monitoramento de desempenho de processos industriais. Será avaliado o desempenho do controlador baseado em suas mudanças de referência, e para casos de mau desempenho, o controlador do processo será atualizado.

1.2.2 Objetivos Específicos

Os objetivos específicos são:

- Implementar um sistema SCADA para a planta industrial presente no Laboratório de Sistemas, Controle e Automação (LSCA);
- Realizar a comunicação da planta com um serviço de nuvem utilizando o protocolo MQTT;
- Gerar um programa em linguagem Python na nuvem para avaliação da malha de controle, e possível re-sintonia;
- Adaptar o sistema SCADA, para que o usuário possa monitorar o desempenho provido pela nuvem em tempo real;
- Validar e testar a integração realizada.

1.3 Estrutura do Texto

O presente trabalho está estruturado da seguinte maneira: O capítulo inicial é a Introdução que tem como objetivo explicitar o panorama geral da proposta e explorar vantagens e desvantagens nas pesquisas sobre o tema em desenvolvimento. Além disso, nele expõe-se os objetivos e se apresenta a estrutura textual da presente proposta. Após isso, o segundo capítulo desse texto é o Embasamento Teórico, onde são apresentadas as teorias que serão utilizadas para o desenvolvimento da metodologia proposta. Depois, inicia-se o capítulo de Resultados e Discussões, onde são mostrados e analisados os resultados obtidos. Por fim, no capítulo final serão apresentadas as conclusões do presente trabalho de conclusão de curso.

2 EMBASAMENTO TEÓRICO

Para que seja possível realizar a integração de sistemas SCADA a serviços de computação em nuvem para monitoramento de desempenho de plantas industriais, é necessário compreender as ferramentas básicas utilizadas, de modo a entender seus usos para otimizar a aplicação desejada. Vantagens e desvantagens das ferramentas escolhidas também serão mostradas.

2.1 Protocolo MQTT

O MQTT (IBM; EUROTECH, s.d.) é um protocolo aberto de mensagens criado pela IBM em 1999. O seu uso efetivo ocorreu tardiamente a sua criação, devido ao fato de que sua padronização pela Organização para o Avanço de Padrões de Informação Estruturados (OASIS, do inglês *Organization for the Advancement of Structured Information Standards*) ter ocorrido apenas em 2014 (YUAN, 2017).

Como é leve e flexível, esse protocolo é atraente aos desenvolvedores de serviços de Internet das Coisas (IoT, do inglês *Internet of Things*), por permitir a comunicação assíncrona entre os componentes do sistema, o que permite que haja leves interrupções na conexão entre os hardwares sem perdas significativas. Além disso, o protocolo exige muito pouco da rede para comunicação, o que reforça o seu uso em aplicações onde a largura de banda pode ser limitada e a latência pode ser alta.

2.1.1 Princípio de funcionamento

Toda conexão via MQTT dispõe de dois tipos de agentes: os clientes e o intermediador (*broker*). Os clientes são os hardwares conectados à internet que trocam informações (mensagens) via protocolo. Eles podem ser publicadores (*publishers*) e/ou assinantes (*subscribers*) (KASHYAP; SHARMA; GUPTA, 2018). Os publicadores enviam e os assinantes recebem as mensagens. Os clientes podem ser controladores, softwares de programação, dispositivos supervisionados ou até mesmo sensores.

O intermediador é um servidor, geralmente executado em um serviço de nuvem, que recebe as mensagens dos publicadores juntamente com o tópico de publicação e envia para os assinantes que solicitaram as mensagens daquele tópico. A estrutura geral da comunicação via protocolo MQTT é mostrada na Figura 1. Esse arranjo permite desacoplar os clientes entre si, tornando necessário apenas o endereço do intermediador e o tópico de publicação para troca de mensagens, o que possibilita que a comunicação possa ser facilmente diversificada quanto ao número de assinantes e quanto ao número de publicadores.

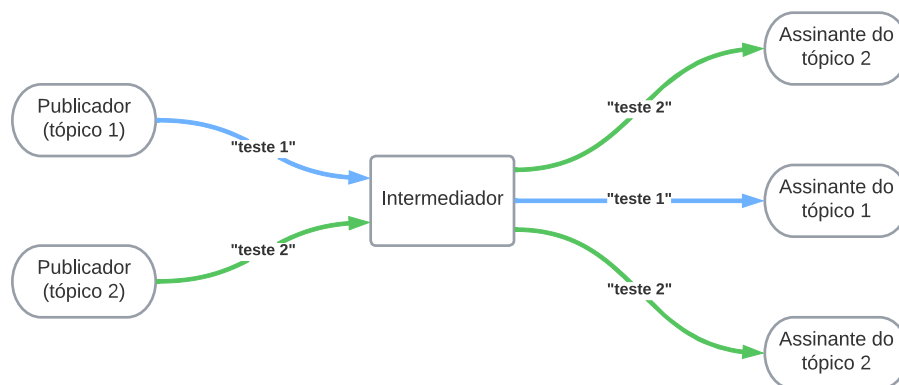


Figura 1 – Estrutura geral da comunicação utilizando o protocolo MQTT.

Fonte: Produção do próprio autor.

2.1.2 Qualidades de serviço

Segundo a IBM (2021), criadora do protocolo, existem três tipos de Qualidades de Serviço (QoS, do inglês *Qualities of Service*):

- QoS 0: a mensagem é entregue no máximo uma vez e não há confirmação da entrega, como mostrado na Figura 2(a). Utilizando essa QoS, a mensagem não é armazenada, podendo ser perdida caso o cliente se desconecte ou o servidor enfrente problemas.
- QoS 1: o modo padrão de transferência de pacotes. A mensagem é sempre entregue, pelo menos uma vez, pois o remetente espera receber uma confirmação de recebimento, representada por ACK (do inglês, *acknowledge*) na figura 2(b). Caso não haja, a mensagem é enviada novamente até que seja reconhecida a entrega. Devido a isso, o destino pode receber múltiplas mensagens iguais;
- QoS 2: a entrega é realizada exatamente uma vez. Esse é o mais seguro, porém mais lento modo de transmissão de dados, já que são realizadas ao menos duas trocas de mensagem entre o destinatário e o remetente. Na primeira, a mensagem é enviada, e o publicador aguarda a confirmação de recebimento (REC). No segundo par de transmissões, o remetente envia uma requisição de exclusão da mensagem (REL, do inglês *Release*) aguarda o processamento da mensagem pelo assinante, enviando uma mensagem requisitando a exclusão da imagem. Por fim, o intermediador envia uma mensagem indicando que a mensagem foi processada e que a transmissão está completa (COMP) (ver Figura 2(c)).

Quanto ao gasto energético pelas três QoS, Toldinas et al. (2019) realizou um estudo de caso utilizando um módulo Wi-Fi IoT ESP-WROOM-02 alimentado por uma bateria de

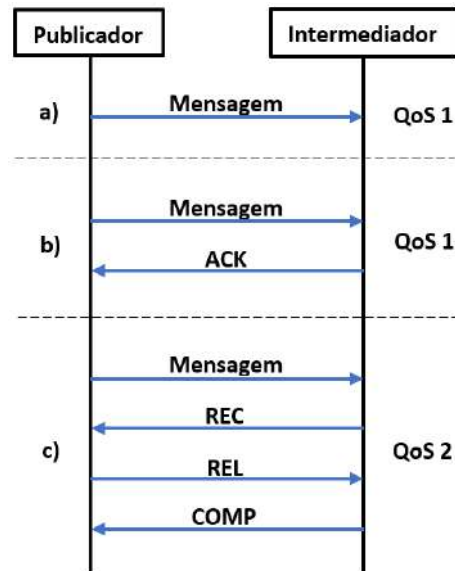


Figura 2 – Modo de transmissão de dados para (a) QoS 0; (b) QoS 1; (c) QoS 2.

Fonte: Rocha et al. (2019).

lítio. A tensão entregue pela bateria para o módulo foi medida e pode-se observar que o nível de tensão na bateria e conseqüentemente o consumo de energia pelo módulo crescem de acordo com a complexidade da qualidade de serviço (Tabela 1).

Tabela 1 – Nível de tensão na bateria por QoS.

QoS	Tensão (V)
0	0.1596
1	0.2054
2	0.2989

Fonte: Toldinas et al. (2019).

2.1.3 Comparação de desempenho

2.1.3.1 Brokers

Após a popularização do protocolo MQTT, diversos intermediadores surgiram no mercado. Exemplos desses são: mosquitto, HiveMQ, ActiveMQ e MQTTRoute. Em um estudo de desempenho realizado por Mishra e Mishra (2020), os quatro *brokers* citados foram avaliados quanto a taxa de mensagens, uso da CPU e latência média. Segundo ele, para toda as QoS, o mosquitto (Eclipse Foundation, 2009) foi considerado o mais eficiente e otimizado dos intermediadores MQTT postos à teste.

2.1.3.2 MQTT versus OPC-UA

No contexto da Indústria 4.0 e da IoT, outros protocolos também são abrangentemente utilizados. Exemplos desses são os protocolos: Plataforma de Comunicação Aberta de Arquitetura Unificada (OPC UA, do inglês *Open Platform Communications Unified Architecture*), OPC UA Publicadores e Assinantes (OPC UA Pub Sub, do inglês *Open Platform Communications Unified Architecture Publisher Subscriber*), Sistema de Operação de Robôs (ROS, do inglês *Robot Operating System*) e sistemas de distribuição de dados (DDS, em inglês *Data Distribution System*) (PROFANTER et al., 2019).

(ROCHA et al., 2019) realizou um estudo comparativo entre dois desses protocolos: OPC-UA e MQTT. Testes sobre a carga útil das mensagens, tempo de entrega e número de clientes foram realizados. O MQTT para todas as qualidades de serviço (QoS, do inglês *Qualities of Service*) apresentou resultados melhores ou semelhantes ao OPC UA. Isso ocorre porque a pilha de protocolos deste é complexa, e ele apresenta diversos serviços na troca de dados, ao contrário do MQTT, que não é estruturado.

2.2 Computação em Nuvem

A computação em nuvem surgiu em 2007 (WANG et al., 2010) como uma solução relacionada a poder de processamento, velocidade de transmissão de dados e comunicação móvel. Ela pode ser considerada um modo de computação terceirizado onde os recursos computacionais são agrupados em grandes centros de dados (*datacenters*) externos e acessados pelos clientes através da Internet.

2.2.1 Serviços de Nuvem

Existem diversos serviços que podem ser oferecidos em nuvem (Figura 3). Quando se carrega uma aplicação via browser na internet ao invés de baixar o software em uma Unidade Central de Processamento (CPU, do inglês *Central Processing Unit*) física, tem-se o que se chama de programa como serviço (SaaS, do inglês *Software as a Service*). Exemplos de SaaS são Google, Facebook, Overleaf. Já a disponibilidade de um ambiente de desenvolvimento onde o usuário tem a liberdade de utilização e produção de seus próprios códigos, trata-se da plataforma como serviço (PaaS, do inglês *Platform as a Service*). Exemplos deste tipo de serviço são Linux, Apache, MySQL, Ruby. Por fim, tem-se o que a literatura chama de infraestrutura como serviço (IaaS, do inglês *Infrastructure as a Service*), onde os provedores disponibilizam armazenamento, rede, sistema operacional, hardware, tudo sob demanda, o que permite uma escalabilidade do serviço, pagando de acordo com a quantidade utilizada. Grandes plataformas fornecedoras deste tipo de serviço são a AWS, a IBM Watson, a Microsoft Azure e a DigitalOcean (MERCADÉ, 2018).

Figura 3 – Serviços oferecidos pela computação em nuvem.



Fonte: Srivastava e Khan (2018).

2.2.2 Contexto Industrial

O estado atual das tecnologias de automação e controle presentes na indústria são em sua maioria baseadas nos Sistemas de Controle Distribuído (DCS, do inglês *Distributed Control Systems*) (DIJKSTRA, 1974) – tecnologia que, majoritariamente, não foi alterada. Essa falta de mudança ocorre por razões de segurança e também devido à natureza conservadora de determinadas indústrias em específico. Indústrias de produção e abastecimento de energia, por exemplo, tem horizontes de planejamento de aproximadamente 50 anos. Existe uma grande discrepância em termos de poder computacional, facilidade de uso e manutenção entre sistemas de automação e tecnologias de pontas existentes, e.g. computadores (BAUER; SCHLAKE, 2017). Devido a isso, segundo Clark (2016), muitos fornecedores de automação preveem uma mudança na tecnologia existente, movidos por áreas da indústria que têm utilizado tecnologia de ponta, por exemplo, telecomunicações e semicondutores.

Existem muitas soluções inovadoras para a área de automação na literatura. Givhchi et al. (2014) propôs a implementação de CLPs virtuais baseado em computação em nuvem. Outros, no entanto, tem utilizado a nuvem como uma ferramenta auxiliar de dispositivos de automação preexistentes. Capaci e Scali (2020) propõe um sistema CLPM onde dados são enviados de CLPs distantes fisicamente, e são processados em um sistema centralizado em nuvem. A grande vantagem desse último tipo de implementação reside na facilidade de integração e no aumento da capacidade computacional em relação à provida por CLPs, sem que haja necessidade de substituir os dispositivos de automação já instalados.

2.3 Estimação de modelos em malha fechada

Em processos industriais, para sintonia de controladores baseados em modelo, a identificação do processo deve ser realizada a partir da relação dos sinais medidos no sistema. A estimação do modelo pode ser realizada em malha aberta ou malha fechada. Na maioria das vezes o modelo é obtido em malha aberta, ou seja, sem nenhum tipo de realimentação, uma relação entre o sinal de entrada e saída do sistema é obtida a partir dos dados do processo. Entretanto, é comum encontrar casos onde a identificação do processo utilizando dados de malha aberta não é possível já que é necessário parar o processo para efetuar a estimação do modelo, justificando o uso da estimação em malha fechada (FORSSELL; LJUNG, 1999).

Tratando-se de malha fechada, os dados para a estimação podem ser obtidos de duas formas, a partir de inserção de perturbações no sistema ou a partir mudança de referência. O sistema em malha fechada tratado no processo de estimação de modelos é mostrado no diagrama de blocos da Figura 4. O sistema é tratado no domínio discreto, e a equação que representa seu comportamento é

$$y[k] = \frac{C(z^{-1})G(z^{-1})}{1 + C(z^{-1})G(z^{-1})}r[k] + \frac{G_e(z^{-1})}{1 + C(z^{-1})G(z^{-1})}d[k], \quad (2.1)$$

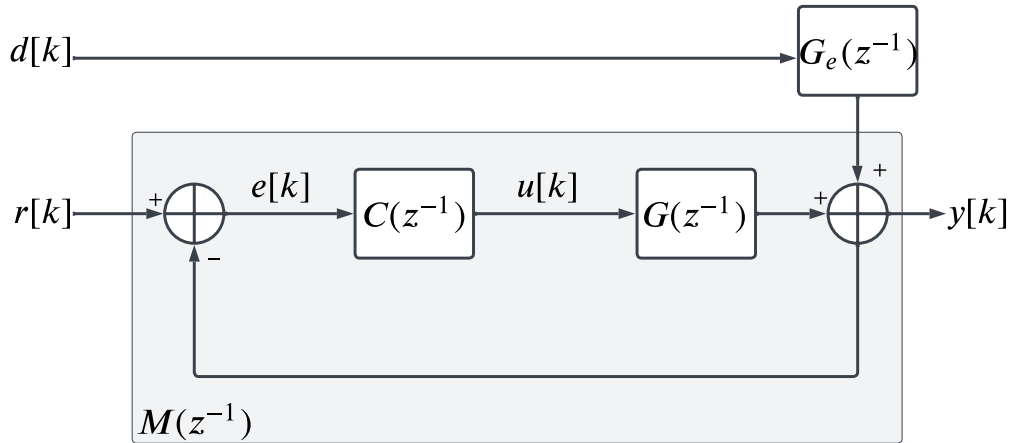
onde $C(z^{-1})$, $G(z^{-1})$, $G_e(z^{-1})$ são as funções de transferência do controlador, do processo e do erro, respectivamente. $r[k]$ é a referência (*setpoint*), $y[k]$ é o valor atual da variável de processo e $d[k]$ é o distúrbio. A partir desse sistema, serão apresentados os três diferentes métodos presentes na literatura para estimação em malha fechada: a identificação direta, a identificação indireta e a identificação conjunta de entrada-saída.

Na identificação indireta, a função de transferência do sistema pode ser reescrita da seguinte forma

$$y[k] = M(z^{-1})r[k] + M_e(z^{-1})d[k], \quad (2.2)$$

onde $M(z^{-1})$ e $M_e(z^{-1})$ são os modelos do processo e do distúrbio em malha fechada. Então, assumindo o pré-conhecimento do controlador $C(z^{-1})$, a função de transferência do processo em malha aberta $G(z^{-1})$ pode ser obtida a partir de

Figura 4 – Diagrama de blocos geral de um sistema em malha fechada.



Fonte: Produção do próprio autor.

$$G(z^{-1}) = \left[\frac{C(z^{-1})}{M(z^{-1})} - C(z^{-1}) \right]^{-1}. \quad (2.3)$$

Como muitos processos $G(z^{-1})$ são bem representados por modelos de ordem baixa e o modelos em malha fechada do sistema $M(z^{-1})$ em geral apresentam ordem alta, muitos cancelamentos ocorrem entre $C(z^{-1})$ e $M(z^{-1})$. Todavia, caso a estimação de $M(z^{-1})$ seja não ideal, esses cancelamentos podem não ocorrer, gerando assim um modelo de alta ordem para o processo.

Na identificação conjunta de entrada-saída, considera-se que a entrada $R(z)$ e saída $Y(z)$ são saídas conjuntas do sistema, excitadas por algum outro sinal extra, como o sinal de controle $U(z)$ ou o erro $E(z)$. Para esse método não é necessário haver conhecimento prévio do sistema, porém demanda mais capacidade computacional (MANDLOI; SHAH, 2015).

Já na identificação direta, o fato de o sistema estar em malha fechada é ignorado, e, partindo do conhecimento do sinal de controle $u[k]$ e da variável de processo $y[k]$. É importante ressaltar que para a aplicação desse tipo de identificação o distúrbio $d[k]$ deve ser decorrelacionado do sinal de controle $u[k]$. A partir disso, é possível obter o modelo do processo a partir de

$$y[k] = G(z^{-1})u[k]. \quad (2.4)$$

O interessante desse método é que o modelo do distúrbio $G_e(z^{-1})$ também pode ser estimado simultaneamente a identificação de $G(z^{-1})$ (SHARDT, 2015).

Diversos métodos podem ser utilizados para a identificação direta de modelos. O Método dos Mínimos Quadrados (LSM, do inglês *Least Squares Method*) é um dos mais utilizados. Ele permite a identificação de um modelo ARX (do inglês, *Autoregressive Model with Exogenous Inputs*) $G(z^{-1})$ a partir do sinal de controle $u[k]$ e da saída $y[k]$. Linguagens de programação de alto nível apresentam funções disponíveis para a realização da estimação do modelos ARXs em suas *toolboxes*/bibliotecas. Em MATLAB, o modelo discreto pode ser estimado a partir do LSM utilizando a função `arx`. Em Python, o módulo SIPPY (em inglês, *System Identification Package for Python*) está disponível no GitHub para a utilização da função `arx` (similar a do MATLAB). As funções de ambas linguagens recebem como entradas os dados de entrada $u[k]$ e saída $y[k]$, além de três parâmetros do modelo: a ordem do sistema (número de polos), o número de zeros, e o atraso (em número de amostras), que é a quantidade de amostras atrasadas da saída $y[k]$ em relação a entrada $u[k]$.

2.3.1 Método dos Mínimos Quadrados

O Método dos Mínimos Quadrados (LSM) é um dos métodos mais utilizados na literatura para a identificação de modelos. A primeira descrição do método foi realizado em Legendre (1806) para minimização de erros de estimação entre órbitas de planetas e foi posteriormente apresentado formalmente por Gauss (GAUSS, 1809).

Sejam os dados de entrada $u[k]$ e saída $y[k]$ com $k = 1, \dots, n$ coletados de um determinado processo. Sabe-se que a partir da equação 2.4 é possível realizar a identificação direta do modelo $G(z^{-1})$ do processo. Considerando que haja uma relação linear entre $u[k]$ e $y[k]$, um modelo de primeira ordem representado por uma regressão linear da forma $y = au + b$ se ajusta bem aos dados. Portanto, deseja-se estimar os parâmetros $[a, b]$ da regressão que minimizem o erro quadrático em relação aos pares de dados $(u[k], y[k])$. Para isso, inicialmente, organiza-se um sistema matricial conforme 2.5.

$$\begin{bmatrix} y[1] \\ \dots \\ y[n] \end{bmatrix} = \begin{bmatrix} u[1] & 1 \\ \dots & \dots \\ u[n] & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = UA \quad (2.5)$$

A matriz U é chamada de matriz de regressores. Pré-multiplicando 2.5 por U^T obtêm-se a equação 2.6.

$$U^T Y = U^T U A \quad (2.6)$$

A matriz $U^T U$ é quadrada, e portanto invertível caso seu posto seja completo. Dado isso, os parâmetros do modelo podem ser obtidos diretamente a partir da equação 2.7, que é a estimativa dos modelo que minimizam o erro quadrático entre as medições e as estimativas. A matriz $U^+ = [U^T U]^{-1} U^T$ é denominada a pseudo-inversa à esquerda de U .

$$A = [U^T U]^{-1} U^T Y \quad (2.7)$$

Em geral, toda função da forma $y = \sum_{k=0}^n a[k] \cdot f[k]$ pode ser colocada na forma mostrada na equação 2.8, tendo assim seus parâmetros estimados por $A = F^+ \mathbf{y}$.

$$y = \begin{bmatrix} f[1] & \dots & f[n] \end{bmatrix} \begin{bmatrix} a[1] \\ \dots \\ a[n] \end{bmatrix} = F A \quad (2.8)$$

Finalmente, considere um sistema representado pela equação a diferenças da forma $y[k + 1] + ay[k] = bu[k - d]$, ou seja, com uma função de transferência discreta $G(z) = \frac{bz^{-d}}{1-az^{-1}}$, sendo d o atraso em número de amostras. Os parâmetros do modelo do sistema podem ser estimados via LSM conforme a equação 2.9, considerando os dados $(u[k], y[k])$ para $k = 1, \dots, n$.

$$\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} -y[1] & u[k] \\ \dots & \dots \\ -y[n] & u[n-d] \end{bmatrix}^+ \begin{bmatrix} y[2] \\ \dots \\ y[n+1] \end{bmatrix} \quad (2.9)$$

Em MATLAB, o mesmo modelo pode ser obtido a partir do código

```

1 data = iddata(u,y,Ts);
2 na = 1;
3 nb = 1;
4 d = 1;
5 G = arx(data,[na nb d]);

```


sendo **data** a estrutura de dados contendo o vetor de dados de entrada u , o vetor de dados de saída y , e o tempo de amostragem dos dados T_s ; **na** a ordem do modelo a ser estimado; **nb** o número de zeros; e **d** o atraso em número de amostras.

2.3.2 Índices de avaliação do modelo

Um dos índices mais utilizados para avaliação de modelos é o fit. Ele é um índice que varia de 0 a 100% e calcula o ajuste do modelo aos dados (quanto maior melhor). Considerando que o \tilde{y} a estimativa de y , o fit pode ser calculado conforme a equação 2.10.

$$\text{fit} = 100 \frac{1 - \|y - \tilde{y}\|}{\|\tilde{y}\|} \quad (2.10)$$

Outro bom índice a se calcular é o número de condição, que estabelece uma estimativa da precisão que se pode obter para a solução do problema $AX = b$. Ele pode ser calculado conforme

$$\kappa(A) = \|A^{-1}\| \cdot \|A\|. \quad (2.11)$$

, onde A é uma matriz inversível e κ o número de condição.

Caso os dados de entrada e saída para cômputo do modelo sejam correlacionados, κ pode apresentar altos valores, pois o posto da matriz X pode não ser completo e a matriz $X^T X$ não ser invertível, ou mal condicionada.

Em MATLAB, ele pode ser estimado pela função `cond`. Em Python, a biblioteca `numpy` disponibiliza a função `linalg.cond`.

2.3.3 Obtenção dos parâmetros do modelo contínuo a partir do modelo discreto

Os parâmetros obtidos via LSM são do modelo discreto, pois são obtidos a partir de dados amostrados. Em geral, para sintonia de controladores utiliza-se modelos contínuos. A conversão discreto-contínuo pode ser realizada a partir do conhecimento da equação do modelo e do tempo de amostragem dos dados.

Considere um sistema SISO LTI representado por um modelo contínuo de primeira ordem com atraso $G(s) = \frac{K}{\tau s + 1} e^{-ds}$. Na equação 2.12 é mostrada a transformada Z de Laplace do

modelo contínuo, mais um segurador de ordem zero $G_h(s) = \frac{1-e^{-Ts}}{s}$, sendo T o tempo de amostragem dos dados.

$$G(z) = Z[G_h(s)G(s)] = z^{-d} \frac{\tau(1 - e^{-T/\tau})}{z - e^{-T/\tau}} \quad (2.12)$$

$G(z)$ é obtida a partir de identificação de parâmetros, sendo necessário obter K e τ a partir dela. Os parâmetros a e b do modelo discreto obtidos via LSM são o polo e ganho do sistema, respectivamente. A partir da equação 2.12, é possível obter a relação entre a constante de tempo τ e o polo a da FT discreta (mostrada na equação 2.13).

$$\ln(e^{-T/\tau}) = \ln(a) \leftrightarrow \tau = \frac{-T}{\ln(a)} \quad (2.13)$$

O ganho K é obtido aplicando o teorema do valor final nos modelos contínuo e discreto, que resulta em

$$\begin{aligned} \lim_{s \rightarrow 0} G(s) &= \lim_{z \rightarrow 1} G(z) \\ K &= \frac{b}{1 - a}. \end{aligned} \quad (2.14)$$

Com essa metodologia, a partir do conhecimento do modelo discreto e do tempo de amostragem, os parâmetros do modelo contínuo do sistema de primeira ordem podem ser obtidos.

Outra forma obtenção dos parâmetros do modelo contínuo é a partir da resposta ao degrau do modelo discreto, caso seja semelhante a resposta de um modelo de primeira ordem. Para obter τ , então, utiliza-se basta verificar o tempo que o modelo demora para alcançar aproximadamente 63,3% do valor de regime, conforme

$$\tau = t(0,633h_f) - \theta, \quad (2.15)$$

onde y_f é o valor em regime após o degrau aplicado e $\theta = d \cdot T_s$ o atraso do processo.

O ganho K do modelo contínuo também pode ser obtido a partir da resposta do modelo discreto, a partir de

$$K = \frac{\Delta y}{\Delta r} = \frac{y_f - y_0}{r_f - r_0}, \quad (2.16)$$

onde Δr é a amplitude do degrau aplicado e Δy é a variação da saída a partir da aplicação do degrau. Assim, evita-se a utilização das operações matemáticas envolvidas nas conversões discreto-contínuo.

Ademais, em MATLAB é possível transformar um modelo discreto em contínuo a partir de diferentes métodos utilizando a função `d2c`.

2.4 Método de sintonia IMC

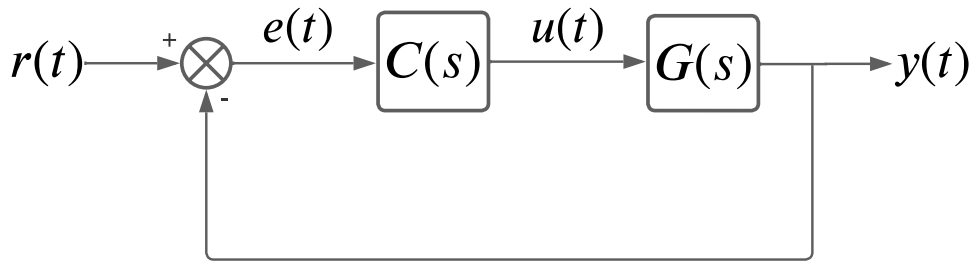
Diversos métodos de sintonia de controladores proporcional integral derivativo (PID) para os mais diferentes tipos de processo foram propostos na literatura (MORARI; ZAFIRIOU, 1989; VISIOLI, 2006; ÅSTRÖM; HÄGGLUND; ASTROM, 2006; FOLEY; JULIEN; COPELAND, 2005; BORASE et al., 2021). Dentre esses, o Método do Modelo Interno (IMC, do inglês *Internal Model Control*) tem sido largamente utilizado, devido a sua facilidade em ajustar o controlador PID para processos que apresentam modelo de baixa ordem. De fato, ele dá ao usuário a liberdade de escolher a resposta desejada do sistema baseada em um só parâmetro (λ).

O método IMC foi popularmente sintetizado em Garcia e Morari (1982), que apresentou-o como um método para sintonia de malhas de alto desempenho. Inicialmente o método era válido apenas para sistemas SISO LTI, até que trabalhos posteriores publicados pelos mesmos autores (GARCIA; MORARI, 1985a; GARCIA; MORARI, 1985b) estendeu a aplicação a sistemas de múltiplas entradas e múltiplas saídas (MIMO, do inglês *Multiple-Inputs Multiple-Outputs*) LTI. Nesse trabalho será abordado o IMC apenas para sistemas SISO LTI.

Considere um sistema SISO LTI estável, representado por um modelo FOPTD (do inglês *First Order Plus Time Delay*) $G(s)$, controlado por $C(s)$ (ver figura 5). $r(t)$ é a referência, $y(t)$ é a saída do sistema e $u(t)$ é o sinal de controle.

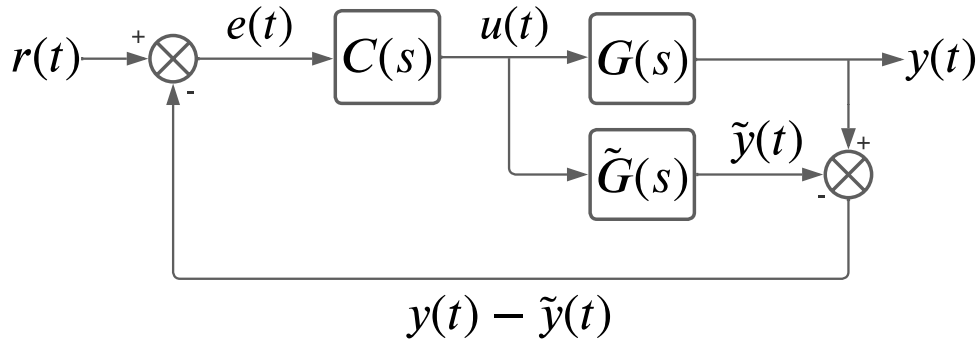
Considere agora a estrutura padrão do método de sintonia IMC (ver Figura 6). $\tilde{G}(s)$ é o modelo do sistema e $C^*(s)$ outro controlador. Considerando que os diagramas das Figuras 5 e 6 tratam do mesmo sistema, a equação 2.17 mostra que um sistema IMC pode ser facilmente representado por um sistema de realimentação padrão, caso os controladores respeitem essa relação.

Figura 5 – Diagrama de blocos do sistema.



Fonte: Produção do próprio autor.

Figura 6 – Estrutura padrão utilizada pelo método de sintonia IMC.



Fonte: Produção do próprio autor.

$$C(s) = \frac{C^*(s)}{1 - C^*(s)\tilde{G}(s)} \tag{2.17}$$

Tratando de $C^*(s)$, caso $\tilde{G}(s)$ fosse um modelo perfeito do sistema (*fit* 100%), o sistema seguiria instantaneamente a referência se $C^*(s) = \tilde{G}^{-1}(s)$. Entretanto, isso não é possível devido aos atrasos e zeros no semiplano direito presentes em sistemas reais. Sendo assim, para que $\tilde{G}(s)$ seja invertível, ele deve ser fatorado conforme a equação 2.18, sendo $\tilde{G}_+(s)$ todas as componentes de fase não-mínima do sistema.

$$\tilde{G}(s) = \tilde{G}_+(s)\tilde{G}_-(s) \tag{2.18}$$

Como sistemas reais têm dinâmicas causais (sistema contínuo), a saída $Y(s)$ não pode sofrer mudanças instantâneas de valor - o que levaria a uma descontinuidade no sistema.

Sendo assim, um filtro passa-baixo é utilizado em série com $\tilde{G}_-^{-1}(s)$. O filtro é da forma $F(s) = \frac{1}{(\lambda s + 1)^n}$, sendo n a ordem apropriada para que o controlador seja realizável (VISIOLI, 2006). Portanto, $C^*(s)$ pode ser representado como mostra a Equação 2.20.

$$C^*(s) = F(s)\tilde{G}_-^{-1}(s) \quad (2.19)$$

A partir das Equações 2.17 e 2.19, $C(s)$ pode ser representado conforme 2.20.

$$C(s) = \frac{C^*(s)}{1 - C^*(s)\tilde{G}(s)} = \frac{F(s)\tilde{G}_-^{-1}(s)}{1 - F(s)\tilde{G}_-^{-1}(s)\tilde{G}(s)} = \frac{F(s)\tilde{G}_-^{-1}(s)}{1 - F(s)\tilde{G}_+(s)} \quad (2.20)$$

Baseado na figura 5, é possível calcular a função de transferência do sistema. Após alguma manipulação algébrica, obtêm-se 2.21.

$$\frac{y(t)}{r(t)} = \frac{C(s)G(s)}{1 + C(s)G(s)} = \frac{\tilde{G}_+(s)}{(\lambda s + 1)^n} \quad (2.21)$$

Quanto a malha de nível controlada pelo inversor da bomba, da planta piloto CPID, é possível representá-la satisfatoriamente por um modelo de primeira ordem com tempo morto (FOPDT, do inglês *first-order plus deadtime*).

$$G(s) = \frac{K e^{-\theta s}}{\tau s + 1}, \quad (2.22)$$

sendo K , τ e θ representam respectivamente o ganho, a constante de tempo e o tempo morto aparente do modelo em malha aberta. A Equação 2.23 mostra o controlador para esse caso.

$$C(s) = \frac{\tau s + 1}{K(\lambda + \theta)s} \quad (2.23)$$

Sabe-se que a função de transferência típica de um controlador PI é $K_p \frac{T_i s + 1}{T_i s}$. Comparando sua equação com a equação 2.23, chega-se aos ganhos

$$K_p = \frac{\tau}{K(\lambda + \theta)}, \quad (2.24)$$

$$T_i = \tau.$$

A resposta do sistema em malha fechada é dada pela Equação 2.25. Como o tempo morto aparente θ é um parâmetro inerente ao sistema, a resposta tem apenas um grau de liberdade, que é a constante de tempo de malha fechada desejada λ .

$$\frac{y(t)}{r(t)} = \frac{C(s)G(s)}{1 + C(s)G(s)} = \frac{e^{-\theta s}}{\left(\frac{\lambda}{\theta} + 1\right)\theta s + e^{-\theta s}} \quad (2.25)$$

2.5 Avaliação de desempenho de malhas de controle

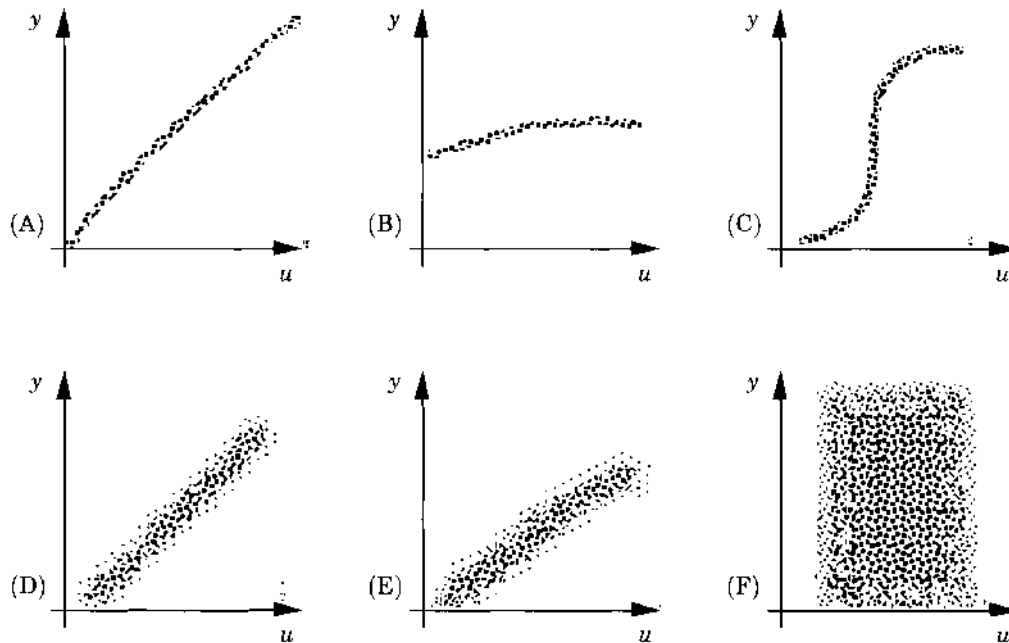
Em plantas industriais de grande porte há centenas ou milhares de malhas de controle. Muitas plantas utilizam pacotes de softwares industriais para a primeira sintonia de suas malhas, que após isso não são mais ajustadas. Devido ao fato do desempenho de malhas de controle se deteriorarem com o tempo por conta de mudanças na dinâmica do sistema, uma grande parte das malhas de controle industriais operam com o desempenho abaixo do ideal. Um estudo realizado em aproximadamente seiscentas mil malhas de controle monitoradas pela ABB (*Asea Brown Boveri*) (STARR; PETERSEN; BAUER, 2016) mostrou que, ao iniciar o monitoramento, aproximadamente 75% dessas malhas não estavam operando de forma satisfatória devido a problemas de sintonia. Em outro estudo apresentado por Bauer et al. (2016), má sintonia e atrito na válvula foram apresentados como os principais causadores de redução de desempenho.

Para realização do monitoramento de desempenho, inicialmente é necessário saber qual o tipo de problema os dados estão indicando. A partir de um gráfico que relaciona a entrada com a saída em estado estacionário, chamado de curva de ganho estática, é possível verificar se a maior parte das variações na saída são provocadas por mudanças na referência ou por distúrbios de carga, i.e., se estamos lidando com um problema de controle servo ou um problema de controle regulatório. Tem-se um problema de controle servo caso seja desejado que a saída siga a referência, e um problema de controle regulatório se a saída deve rejeitar distúrbios. Obviamente, há sistemas que mesclam um pouco dos dois problemas.

A Figura 7 mostra exemplos de curvas de ganho estáticas, sendo u o sinal de entrada e y o sinal de saída do processo. Para problemas de controle regulatório, a mudança de referência não é a maior causadora da alteração da relação entre entrada e saída. Muitas vezes são

a inserção de cargas no processo e/ou mudanças na dinâmica de malhas interativas que modificam essa relação.

Figura 7 – Exemplos de curvas de ganho estáticas. (A), (B) e (C): problema de controle servo. (D) e (E): mistura entre os dois problemas. (F) problema de controle regulatório.



Fonte: Åström, Hägglund e Astrom (2006).

Em um problema de controle servo, as variações no ganho do sistema podem ser determinadas. Isso auxilia a determinar se há necessidade de alteração nos ganhos do controlador. Além disso, as curvas de ganho estáticas podem ser utilizadas para detectar mudanças na dinâmica do sistema, auxiliando na detecção e no diagnóstico de falhas no sistema. O estudo de caso realizado na planta-piloto CPID apresenta ambos problema de controle servo e regulatório.

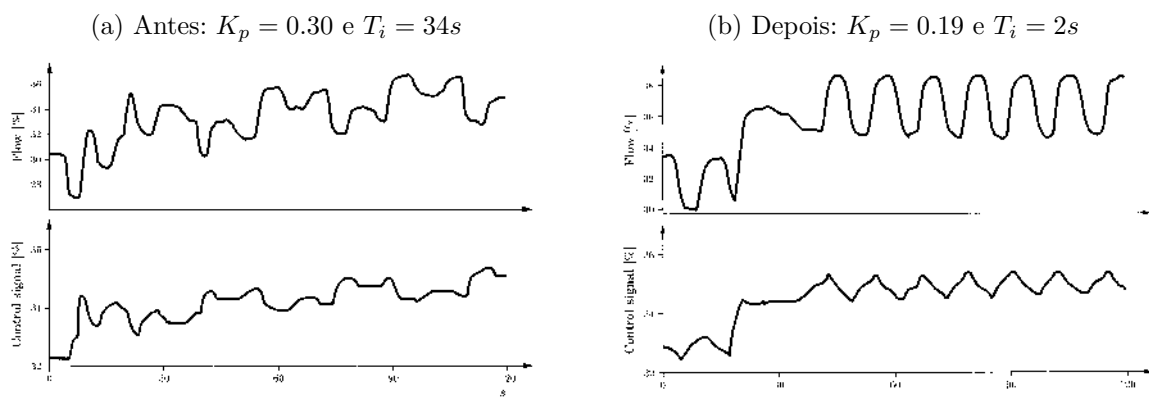
Em geral, a CLPA é uma tarefa que pode ser realizada por meio da sequência de passos sugerida por Jelali (2006), que são:

- Cálculo de um índice de desempenho utilizando os dados disponíveis.
- Definir um valor de referência, que representa o desempenho desejado.
- O desvio do desempenho do controle atual em relação à referência, considerando assim o controle satisfatório ou não.
- Como há vários possíveis motivos para uma malha de controle apresentar mau desempenho, as razões para isso podem ser diagnosticadas.

- Ações para a correção do desempenho são sugeridas ao operador da planta.

Alguns problemas na dinâmica do sistema podem ocasionar diminuição do desempenho da malha de controle. Devido a isso, o CLPM pode ser utilizado também para diagnosticar oscilações nas malhas e aumentar provisoria ou definitivamente o desempenho por meio de uma nova sintonia de controladores. A causa mais comum para oscilações é atrito nas válvulas de controle. Esse problema gera uma saída do processo com forma de onda quadrada e sinal de controle com forma triangular. Em Åström, Hägglund e Astrom (2006) dados de um processo controlado por um PI em uma planta com atrito na válvula foram mostrados. A figura 8 mostra a resposta do sistema a uma mudança de referência antes e depois de uma re-sintonia. K_p e T_i são respectivamente o ganho proporcional e o tempo integral, parâmetros do controlador PI utilizado para controle do processo em malha fechada. O tempo de estabelecimento do sistema foi reduzido e após a sintonia pode-se notar de forma mais clara o comportamento oscilatório esperado devido ao atrito na válvula de controle. Assim como no exemplo da Figura 8, o presente trabalho utiliza a resintonia de controladores para a recuperação do desempenho de malhas de controle.

Figura 8 – Dados de processo de uma planta com atrito na válvula de controle, antes e depois da nova sintonia.



Fonte: Åström, Hägglund e Astrom (2006).

Muitas são as soluções propostas para CLPA/CLPM na literatura (HARRIS, 1989; VERONESI; VISIOLI, 2009; SWANDA; SEBORG, 1999; YU et al., 2011). Elas podem ser divididas em dois grupos principais: monitoramento de desempenho estocástico e determinístico. O monitoramento estocástico avalia o desempenho do sistema ao lidar com distúrbios estocásticos e o monitoramento determinístico utiliza métricas de avaliação das malhas como mudanças de referência e rejeição de perturbação de carga.

2.5.0.1 Índice de Harris

Um dos índices de avaliação estocásticos mais utilizados é o Índice de Harris (HARRIS, 1989), exposto na Equação 2.26. O objetivo desse índice é calcular a variância da variável

de processo σ_y e compará-la com a mínima variância alcançável σ_{MV} . Ele varia entre 0 e 1, onde valores próximos de 0 indicam que a variância do processo é próxima a mínima variância, o que significa que a malha está com um bom desempenho. Caso a variância do processo esteja alta quando comparada a mínima variância, o índice terá valores próximos de 1.

$$I_H = 1 - \frac{\sigma_{MV}^2}{\sigma_y^2} \quad (2.26)$$

Para cálculo do índice de Harris é necessário conhecer a mínima variância do processo, que pode ser calculada apenas com o tempo-morto do processo. Entretanto, para controladores simples, como PI e PID, há uma grande dificuldade em determinar-se valores razoáveis para o índice. Além disso, quando obtidos valores razoáveis, geralmente esses levam a controles muito agressivos (ÅSTRÖM; HÄGGLUND; ASTROM, 2006).

2.5.0.2 Avaliação baseada em dados de resposta a mudanças de *setpoint*

Nos métodos determinísticos, diversas literaturas com abordagens similares para a avaliação baseada em mudanças de *setpoint* foram propostas (SWANDA; SEBORG, 1999; VERONESI; VISIOLI, 2009; YU et al., 2011). Em linhas gerais, os trabalhos analisados comparam o desempenho da malha de controle ao desempenho de um modelo FOPDT (do inglês, *First Order Plus Dead Time*) do sistema controlado por um PI(D) sintonizado via método IMC, sendo esse a referência de desempenho. O índice de desempenho básico utilizado nas abordagens é a Integral Absoluta do Erro (IAE), calculado a partir de

$$\text{IAE} = \sum_{i=1}^N |e[i+1] - e[i]|T_s, \quad (2.27)$$

onde $e[i]$ é o erro, T_s é o tempo de amostragem e N é o número de amostras enviadas à nuvem.

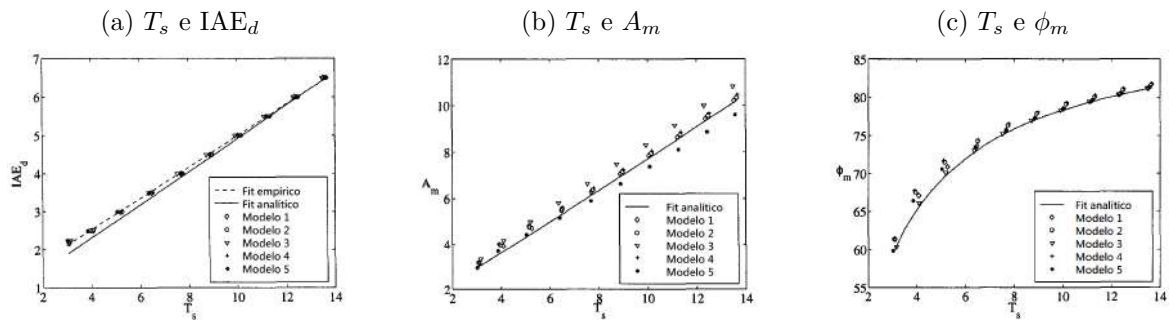
No trabalho realizado por Veronesi e Visioli (2009), dois índices adimensionais CI e SI são calculados, e a partir deles um processo iterativo para melhora do desempenho é realizado. Entretanto, o processo não tem uma abordagem direta ao problema, gerando resultados não interpretáveis e em alguns casos processos iterativos longos.

Em Yu et al. (2011) o índice adimensional η_{IAE} é responsável pela avaliação da malha de controle do sistema. Ademais, um índice livre de ruídos $\tilde{\eta}_{IAE}$ também é calculado. Comparando os dois índices é possível verificar em qual intensidade o ruído afeta os

resultados, e, caso os dois tenham uma grande diferença entre si, a avaliação por η_{IAE} deve ser desconsiderada e refeita.

Em Swanda e Seborg (1999) dois índices adimensionais são calculados. O primeiro, IAE_d , é a razão entre o IAE normalizado pelo tamanho do degrau e o tempo morto aparente do sistema em malha fechada θ_a . O segundo, T_s , apresenta uma relação direta com a razão entre a constante de tempo de malha fechada τ_c e θ_a . Além disso, Swanda e Seborg também constataram, após simplificações matemáticas, que o IAE_d é diretamente proporcional a T_s . Sendo assim, valores de IAE_d podem ser obtidos diretamente a partir do conhecimento de τ_c e θ_a . O autor também constatou que o T_s (e conseqüentemente IAE_d) apresenta relações conhecidas com as margens de ganho (A_m) e fase do sistema (ϕ_m), possibilitando ao usuário gerar o desempenho desejado baseado na margem de fase e ganho. Cinco modelos de ordens distintas, incluindo um de ordem cinco, foram testados e o comportamento obtido foi semelhante ao modelo FOPDT. A Figura 9 apresenta as relações apresentadas e os testes para diversos modelos.

Figura 9 – Comparação entre as relações empíricas e analíticas com modelos numericamente determinados.



Fonte: Swanda e Seborg (1999).

O autor gerou intervalos de valores para T_s , IAE_d e o sobressinal da resposta (OS), calculado a partir de

$$OS = 100 * (y_{max} - \Delta r) / \Delta r, \quad (2.28)$$

onde y_{max} é o valor máximo da saída durante a aplicação do degrau e Δr a variação da referência. O desempenho dos controladores foram separados em três classes: alto desempenho, excessivamente lento e mal sintonizado. Uma comparação entre a metodologia proposta por Swanda e Seborg (1999) e o presente trabalho será realizada.

Todos os trabalhos até agora citados utilizam um modelo do processo controlado por um PI(D) sintonizado via método IMC como referência. Simplificações tomadas nas três

propostas levam o cálculo da referência da IAE depender apenas do tempo morto, do tamanho da mudança de referência aplicada e da constante de tempo do sistema em malha fechada. Em alguns casos, um caso específico do IMC criado por Skogestad (SKOGESTAD; POSTLETHWAITE, 2007) é utilizado, retirando a dependência da constante de tempo em malha fechada para o cálculo do IAE. Apesar de muito simples, em determinadas situações onde a dinâmica do processo é alterada e refletida apenas no ganho em malha fechada do modelo, a piora do desempenho não é notada na referência de IAE.

Os métodos apresentados na literatura são baseados na existência de um modelo obtido, em geral, em malha aberta, que é utilizado para calcular o limite de desempenho para o IAE. Entretanto, nem sempre esse modelo está disponível, como é o caso de quando está o sistema esteja operando em malha fechada. Além disso, em ocasiões de mudança na dinâmica do sistema, como em situação de falha, o modelo pode sofrer alteração, não sendo mais válidos para a avaliação do desempenho. Neste trabalho, portanto, tanto o cálculo do limiar do desempenho quanto seu monitoramento serão realizados unicamente a partir de dados de operação coletados continuamente da malha de controle.

2.6 Trabalhos Relacionados

Desde o final do século passado, muitos trabalhos renomados têm sido publicados na área de Monitoramento e/ou Avaliação de Desempenho de Malhas de Controle (CLPM/CLPA, do inglês *Control Loop Performance Monitoring/Control Loop Performance Assessment*) (LYNCH; DUMONT, 1996; CHOUDHURY; SHAH; THORNHILL, 2004; QIN, 1998; HÄGGGLUND, 1995; HUANG, 2008). Alguns deles tem ganhado popularidade na comunidade acadêmica (ZHAN; LI; YANG, 2019; SHAHNI; YU; YOUNG, 2019); outros, na indústria (STARR; PETERSEN; BAUER, 2016; DESBOROUGH; MILLER, 2002). Ademais, com o surgimento da Indústria 4.0 e evolução de suas Tecnologias Chaves de Habilitação (KET, do inglês *Key Enabling Technologies*), trabalhos que utilizam tecnologias como Internet das Coisas Industrial (IIoT, do inglês *Industrial Internet of Things*) e computação em nuvem têm se disseminado na academia (CAPACI; SCALI, 2020; GAO et al., 2016; ALI et al., 2020).

O trabalho de Capaci e Scali (2020) tem gerado resultados concretos quanto à integração dos ambientes SCADA e nuvem via protocolo Transporte Enfileirado de Dados por Telemetria (MQTT, do inglês *Message Queuing Telemetry Transport*). Em nuvem, os dados são recebidos e tratados, e as malhas de controle são avaliadas. Possíveis falhas nos dispositivos da planta também são diagnosticadas pelo sistema, que retorna a avaliação geral ao controlador industrial. Além disso, a arquitetura proposta permite a centralização do sistema de monitoramento, o que possibilita a utilização de uma só nuvem para a avaliação de diversas plantas industriais. Contudo, tratando-se de uma solução prática

para a indústria, o problema de cibersegurança e confiabilidade do sistema ainda estão presentes, já que os dados são enviados pelo Controlador Lógico Programável (CLP) de forma ininterrupta à nuvem. Diferentemente, nesse trabalho não haverá envio contínuo de dados, mas apenas quando houver necessidade de um serviço da nuvem, reduzindo a demanda de comunicação.

Ademais, os resultados parciais deste trabalho foram aceitos para publicação no XXIV Congresso Brasileiro de Automática, Outubro de 2022, Fortaleza, Brasil; com o título "Sistema baseado em nuvem para o monitoramento de uma planta industrial".

3 METODOLOGIA E ETAPAS DE DESENVOLVIMENTO

3.1 Planta Piloto

Para a realização desse trabalho de conclusão de curso, a planta industrial presente no LSCA-CPID foi utilizada (ver Figura 10). Ela é chamada pela literatura de tanque quádruplo, e é muito utilizada para ensinar conceitos importantes de teoria de controle. Ela contém quatro tanques, duas bombas centrífugas, quatro válvulas pneumáticas, 14 válvulas manuais, dois medidores de vazão, quatro medidores de nível e o reservatório inferior. Para a aplicação realizada, o intuito foi controlar o nível apenas no tanque inferior esquerdo, utilizando-se para isso apenas uma válvula pneumática, duas válvulas manuais, um medidor de vazão e um medidor de nível (parte selecionada na Figura 10).

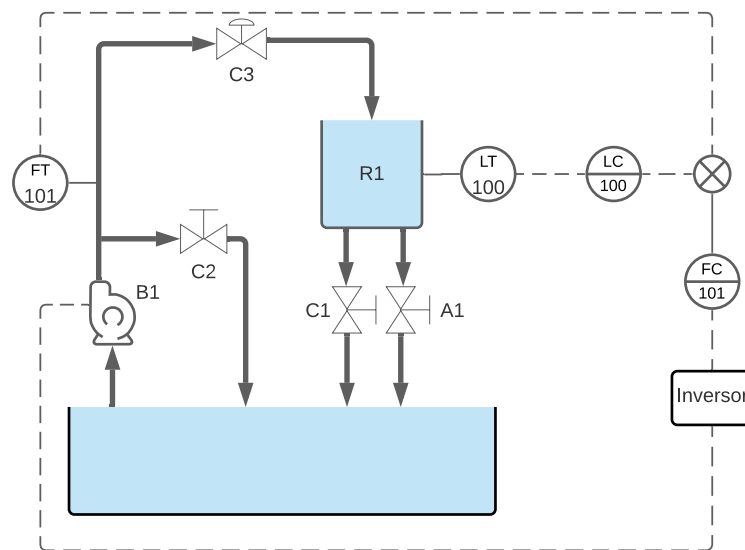
Figura 10 – Planta piloto LSCA-CPID.



Fonte: Produção do próprio autor.

A Figura 11 mostra a ligação dos componentes utilizados e das malhas de controle. Em operação normal a bomba B1, controlada pelo inversor de frequência, bombeia água do reservatório inferior para o tanque R1, que retorna a água ao reservatório inferior por meio da válvula manual A1. Em termos de simulação de falha, a válvula manual C1 simula um vazamento do reservatório, a válvula manual C2 simula um mau desempenho da bomba, e a válvula pneumática C3 simula um entupimento nos dutos de condução do fluido. O controle do sistema é feito a partir de duas malhas de controle. O controle de nível (LC) gera uma referência de vazão para o controle de vazão (FC) e o controle de vazão gera um sinal para o inversor, obtendo assim o nível desejado.

Figura 11 – Fluxograma do processo.

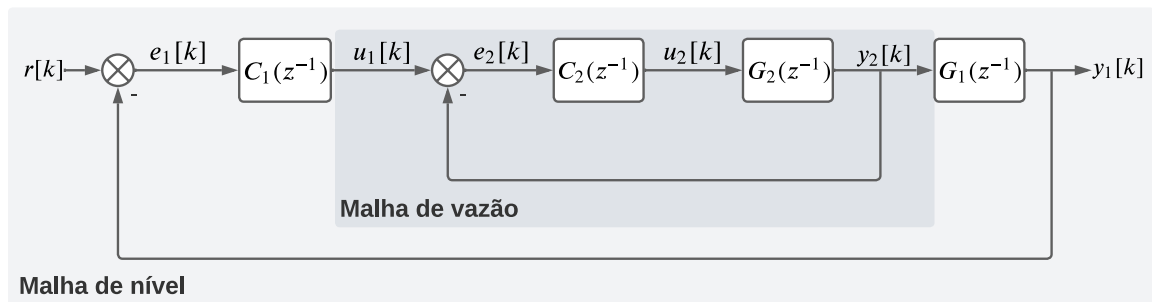


Fonte: Produção do próprio autor.

O diagrama de controle do processo é mostrado na Figura 12. Uma mudança na referência de nível $r[k]$ gera um aumento do erro $e_1[k]$. Conseqüentemente o controlador de nível $C_1(z^{-1})$ demanda uma referência de vazão $u_1[k]$ para que esse erro diminua. A mudança de $u_1[k]$ gera um aumento do erro $e_2[k]$, o que modifica o sinal de controle $u_2[k]$ enviado pelo controlador de vazão $C_1(z^{-1})$ ao inversor de frequência, gerando assim uma nova vazão $y_2[k]$ e um novo nível $y_1[k]$.

A estrutura mestre-escravo é realizável por conta da diferença entre as constantes de tempo de malha aberta do nível ($\tau_1 \approx 90s$) e da vazão ($\tau_2 \approx 0.3s$). Sendo assim, a malha de vazão é considerada como um ganho unitário pela malha de nível. Essa composição é vantajosa porque os distúrbios de vazão são compensados rapidamente pela malha interna, o que reduz o impacto na variável controlada, o nível.

Figura 12 – Diagrama de controle do processo.



Fonte: Produção do próprio autor.

3.2 Conexão e Programação do CLP

3.2.1 Conexão dos instrumentos industriais ao CLP

Os instrumentos industriais da planta-piloto CPID estavam inicialmente conectados a um sistema de automação DeltaV da Emerson. Para o controle de nível do reservatório inferior esquerdo pelo CLP Nexto NX3005, parte dos instrumentos ligados ao DeltaV foram desconectados e conectados ao Nexto NX3005.

3.2.2 Programação dos algoritmos de controle

O *software* de programação dos CLPs da fabricante Altus é o Mastertool. Nele é possível criar programas nas linguagens de programação disponíveis na norma IEC 61131-3, criar telas de supervisão, disponibilizar dados de processo em servidores OPC, utilizar diferentes bibliotecas para auxiliar a programação dos CLPs, etc. É importante ressaltar também, que o Mastertool, assim como diversos outros *softwares* utilizados para programação em CLPs, é uma plataforma baseada no *CodeSYS*. Sendo assim, o processo realizado para o CLP presente é o mesmo realizado para de outros fabricantes.

Para controle do nível do reservatório e da vazão foram utilizados controladores PI. PIDs não foram utilizados porque os sinais obtidos pelos medidores de nível e vazão, principalmente o de vazão, são bastante ruidosos. O ganho derivativo pode amplificar o ruído nesses casos.

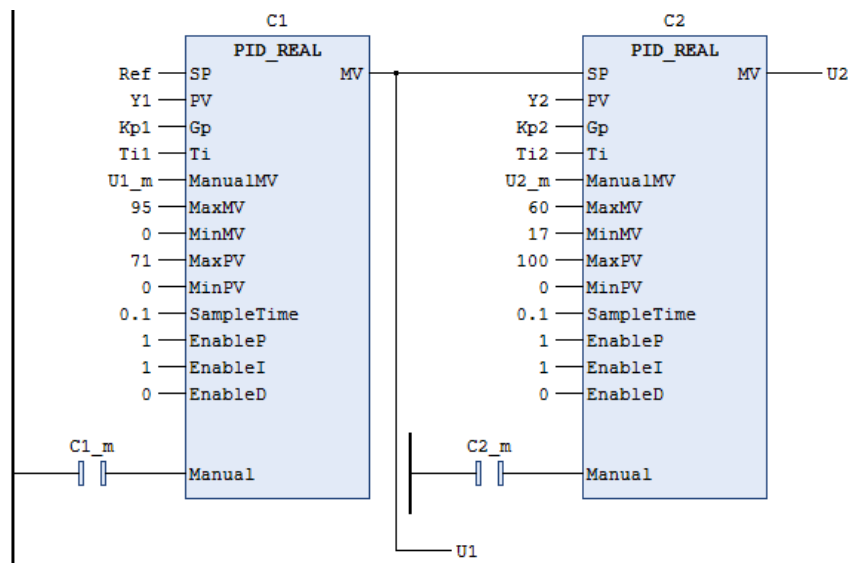
Para adicionar os controladores foi necessário inserir a biblioteca NextoPID ao projeto. Os dois PIs em cascata foram adicionados e configurados conforme mostra a Figura 13. Para o funcionamento das malhas, foi necessário obter alguns parâmetros dos instrumentos industriais, para configuração dos limites das variáveis manipuladas (MV) e de processo

(PV).

A variável de processo do controlador C1 é o nível do reservatório, que pode variar de $MinPV = 0\text{cm}$ a $MaxPV = 71\text{cm}$. A variável manipulada pelo controlador C1 é a vazão de água disponibilizada pela bomba, que varia de $MinMV = 0\text{L/min}$ a aproximadamente $MaxMV = 95\text{L/min}$. A variável de processo do controlador C2 também é a vazão, e por isso apresenta os mesmos limites que MV de C1. Já a variável manipulada do controlador C2 é o sinal de controle enviado ao inversor de frequência, dado em Hertz, que varia de $MinMV = 17\text{Hz}$, que é a frequência onde a bomba tem força suficiente para alimentar o reservatório R1 com água, a $MaxMV = 60\text{Hz}$.

A referência dos dois controladores é representado pelo *SP* (do inglês, *setpoint*). Os ganhos proporcionais e os tempos integrais são representados por *Gp* e *Ti*, respectivamente. *ManualMV* é o valor de MV quando o controlador está em modo manual, ou seja, quando a malha está aberta. *SampleTime* é o período do ciclo de execução do controlador. *EnableP*, *EnableI* e *EnableD* são as chaves responsáveis por ativar ou desativar o controle proporcional integral e derivativo, respectivamente.

Figura 13 – Malhas de controle do processo programadas em linguagem Ladder.



Fonte: Produção do próprio autor.

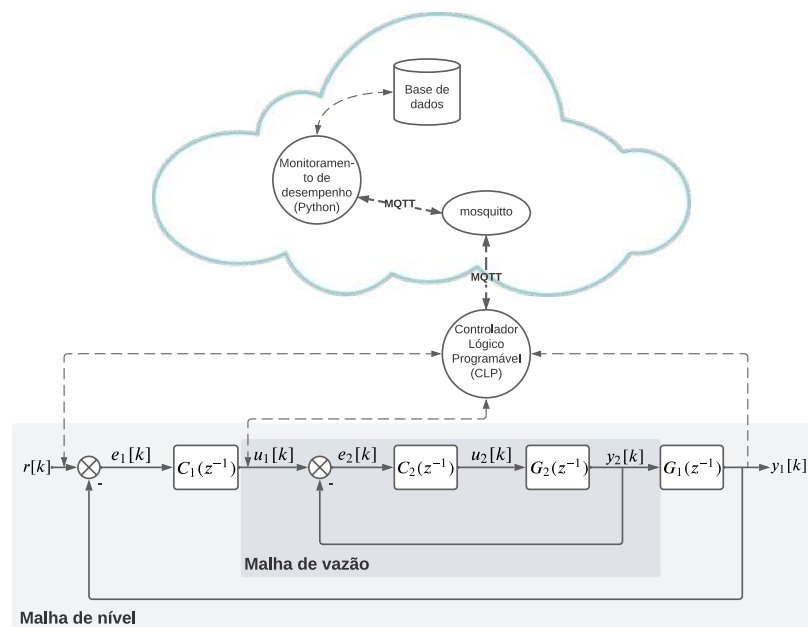
3.2.3 Implementação do MQTT

Para que o CLP consiga se comunicar com máquinas localizadas em redes externas, é necessário que ele se conecte diretamente ao *gateway* padrão, em geral o roteador. Ele deve estar conectado a mesma rede que o computador detentor do Mastertool, para que seja possível realizar a transferência do código do computador para o CLP.

O CLP Nexto NX3005 tem uma biblioteca nativa de nome LibMQTT, que permite a configuração de programas para envio e recebimento de dados via protocolo MQTT. Ela apresenta dois parâmetros a serem configurados, que é o número de tópicos de publicação e de inscrição. A Altus também disponibiliza na sua página um leiaute para facilitar a publicação e inscrição de dados. Ele contém as funções necessárias para que o CLP se comunique com o intermediador. Deve-se então, por fim, modificar tal leiaute e os parâmetros da biblioteca para publicar e assinar os tópicos desejados, além do endereço do intermediador.

A estrutura geral (ver Figura 14) consiste então na troca de dados entre rotina de monitoramento e o CLP via MQTT. O CLP coleta dados de processo, e os envia à rotina de monitoramento de desempenho programada em Python na nuvem. O agente principal do protocolo MQTT é o intermediador. Devido às questões de desempenho mostradas na subseção 2.1.3.1 o mosquitto foi escolhido como o intermediador oficial da solução. A partir disso são realizados processamentos dos dados na rotina de programação e os mesmos são persistidos em um banco de dados. Eventualmente, após a requisição pelo CLP de alguma funcionalidade presente na rotina em Python, a mesma pode também enviar os dados demandados ao CLP.

Figura 14 – Estrutura geral da solução proposta.



Fonte: Produção do próprio autor.

3.3 Monitoramento de desempenho

Para monitorar o desempenho da malha de controle, o índice de avaliação de desempenho a se analisar é a integral do erro absoluto (IAE, do inglês, *integral of absolute error*). O IAE é dependente do tamanho da mudança de referência r_0 aplicada ao processo. Para que os IAEs possam ser avaliados independentemente da amplitude de r_0 , é proposto o IAE normalizado IAE_n , determinado pela equação 3.1.

$$\text{IAE}_n = \frac{\text{IAE}}{|r_0|} \quad (3.1)$$

Nesse trabalho para o monitoramento de desempenho é proposto um teste de hipótese sobre a média do IAE_n . Inicialmente, na etapa de treinamento, um conjunto de amostras $\text{IAE}_n[k]$, com $k = 1, \dots, L$ da população é utilizada para cálculo da média da população μ_0 . Após isso, inicia-se a etapa de teste, onde uma janela deslizante de valores de IAE_n são utilizados para cálculo da média móvel $\overline{\text{IAE}_n}$, utilizando N amostras para tal. A cada novo valor de IAE_n obtido, $\overline{\text{IAE}_n}$ é calculado utilizando $N - 1$ valores passados e o valor atual, e o teste de hipótese mostrado em 3.2 é realizado, onde H_0 é a hipótese nula e H_1 é a hipótese alternativa.

$$\begin{aligned} H_0 : \overline{\text{IAE}_n} &= \mu_0 \\ H_1 : \overline{\text{IAE}_n} &> \mu_0 \end{aligned} \quad (3.2)$$

Deve-se ressaltar que o teste está sujeito a erros, e um dos parâmetros que estão relacionados a taxa de erros é o coeficiente de confiança α , parâmetro a ser escolhido, que indica a porcentagem de amostras de $\overline{\text{IAE}_n}$ que são estatisticamente iguais a μ_0 , mas foram dadas como diferentes pelo teste de hipótese. Esse erro é denominado de erro do tipo I.

Como a variância da população $\overline{\text{IAE}_n}$ não é conhecida, utiliza-se a distribuição T de Student para modelar o comportamento da população. Para a distribuição T, a variância da população real σ^2 é aproximada da variância (S^2) do conjunto de amostras de $\text{IAE}_n[k]$, com $k = 1, \dots, L$ utilizadas para treino.

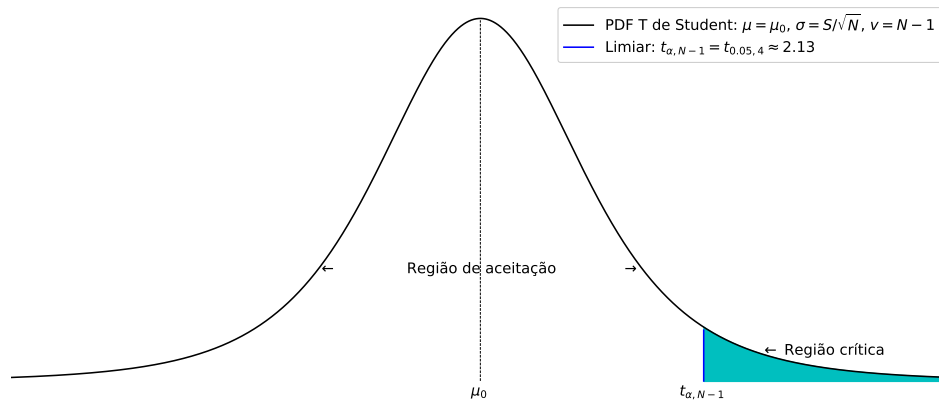
Baseado na escolha de α e no número de amostras N utilizadas para cálculo de $\overline{\text{IAE}_n}$, é possível calcular até que ponto $\overline{\text{IAE}_n}$ pode se distanciar de μ_0 sem que haja rejeição de H_0 . A métrica utilizada para isso utilizando a distribuição T é chamada de $t_{\alpha,v}$, sendo v o número de graus de liberdade, geralmente dado por $N - 1$.

Ademais, t_0 definem a estatística de teste para o valor de $\overline{\text{IAE}}_n$ obtido, dado por

$$t_0 = \frac{\overline{\text{IAE}}_n - \mu_0}{\sqrt{S^2/N}}. \quad (3.3)$$

A Figura 15 ilustra a função de densidade de probabilidade (PDF, do inglês *probability density function*) normalidade de uma população exemplo de $\overline{\text{IAE}}_n$.

Figura 15 – PDF de população de médias com distribuição T de Student ($\alpha = 5\%$ e $N = 5$)



Fonte: Produção do próprio autor.

Caso $t_0 < t_{\alpha, v}$, t_0 encontra-se na região de aceitação. Portanto, há falha em rejeitar a hipótese nula H_0 , e, logo, estatisticamente a média $\overline{\text{IAE}}_n$ é igual a μ_0 , e a malha de controle apresenta desempenho normal. Entretanto, se $t_0 > t_{\alpha, v}$, t_0 se encontra na região crítica. Assim, rejeita-se a hipótese nula H_0 , e assim, existem evidências suficientes para considerar que $\overline{\text{IAE}}_n$ maior que μ_0 , considerando assim o desempenho como normal.

Para trabalhar em termos de $\overline{\text{IAE}}_n$, a inequação do teste de hipótese caso H_0 seja rejeitado pode ser reescrita como

$$\begin{aligned} t_0 &> t_{\alpha, v} \\ \frac{\overline{\text{IAE}}_n - \mu_0}{\sqrt{S^2/N}} &> t_{\alpha, v} \\ \overline{\text{IAE}}_n &> \mu_0 + t_{\alpha, v} \sqrt{\frac{S^2}{N}}. \end{aligned} \quad (3.4)$$

A mesma equação é utilizada para falha em rejeitar H_0 , sendo alterado apenas o sinal da inequação. Dessa forma, pode-se avaliar o desempenho a partir do $\overline{\text{IAE}}_n$. A parte direita da inequação é denominada de limiar de monitoramento IAE_L no presente trabalho.

$$\text{IAE}_L = \mu_0 + t_{\alpha,v} \sqrt{\frac{S^2}{N}}. \quad (3.5)$$

Portanto, a partir da média μ_0 e do desvio padrão S^2 da população, a partir de um conjunto de amostras representativas de tamanho L , e do conhecimento do coeficiente de confiança desejado α , e do número de amostras N utilizadas no cálculo de $\overline{\text{IAE}}_n$, é possível calcular um limiar estatístico para $\overline{\text{IAE}}_n$.

Para efeito de comparação com a metodologia proposta no presente trabalho, o monitoramento de desempenho de Swanda e Seborg (1999) foi implementado. Considera-se o sistema efetivamente representado por um modelo de primeira ordem mais tempo morto (FOPDT) dado por

$$G(s) = \frac{K e^{-\theta_a s}}{\tau_a s + 1}, \quad (3.6)$$

onde K é o ganho do modelo, θ_s o tempo morto aparente do sistema e τ_a a constante de tempo. A partir do método de sintonia IMC, obtém-se a função de transferência do controlador $G_c(s)$ mostrada na equação 3.7.

$$G_c(s) = \frac{\tau_a s + 1}{K(\tau_c + \theta_a)s}, \quad (3.7)$$

que corresponde a um controlador PI com os parâmetros igual a

$$K_p = \frac{\tau_a}{K(\tau_c + \theta_a)} \quad T_i = \tau_a, \quad (3.8)$$

onde τ_c é a constante de tempo de malha fechada desejada. Para o sucesso da sintonia, Rivera, Morari e Skogestad (1986) e Fruehauf, Chien e Lauritsen (1994) comprovam que o τ_c escolhido deve ser maior ou igual ao tempo morto aparente θ_a do sistema.

O índice de avaliação de desempenho de malha de controle em Swanda e Seborg (1999) é o IAE_d , o IAE adimensional, calculado conforme

$$\text{IAE}_d = \frac{\text{IAE}}{|r_0|\theta_a}, \quad (3.9)$$

onde r_0 é o tamanho da mudança de referência aplicada.

Segundo os autores, é possível calcular um valor esperado para o IAE_d , denominado de IAE_e .

$$\text{IAE}_e = 1 + \frac{\lambda}{\theta_a} \quad (3.10)$$

Como o IAE_n , IAE normalizado proposto pelo presente trabalho, não utiliza θ_a para normalização, o valor esperado de IAE_n pode ser calculado conforme

$$\text{IAE}_s = \theta_a \text{IAE}_e = \theta_a + \lambda, \quad (3.11)$$

o que permite uma comparação das metodologias utilizando os mesmos valores (IAE_n).

3.4 Resintonia

Variadas falhas no processo e em instrumentos industriais podem reduzir o desempenho da malha de controle. Para algumas delas, os parâmetros do modelo tendem a ser alterados. Assim, uma possível metodologia é a estimação do modelo em falha para resintonia do controlador. Sabe-se que o método de sintonia IMC promete, a partir de um sistema modelado por um modelo FOPDT, entregar uma resposta em malha fechada com constante de tempo igual a λ .

Sendo assim, para recuperar o desempenho, uma das possíveis metodologias a serem usadas para a resintonia pode ser sintonizar o controlador via método IMC (ver Seção 2.4) utilizando a mesma constante de tempo desejada em malha fechada λ para desempenho normal e os parâmetros do modelo do processo em baixo desempenho. Espera-se que ao sintonizar o controlador para um mesmo λ , τ_c se aproxime de λ novamente, como em normalidade.

Entretanto, algumas falhas no processo que geram redução de desempenho podem afetar pouco os parâmetros do modelo do processo. Devido a isso, uma resintonia utilizando o modelo obtido em falha não é justificada. Para esses casos, Jelali (2012) propôs uma

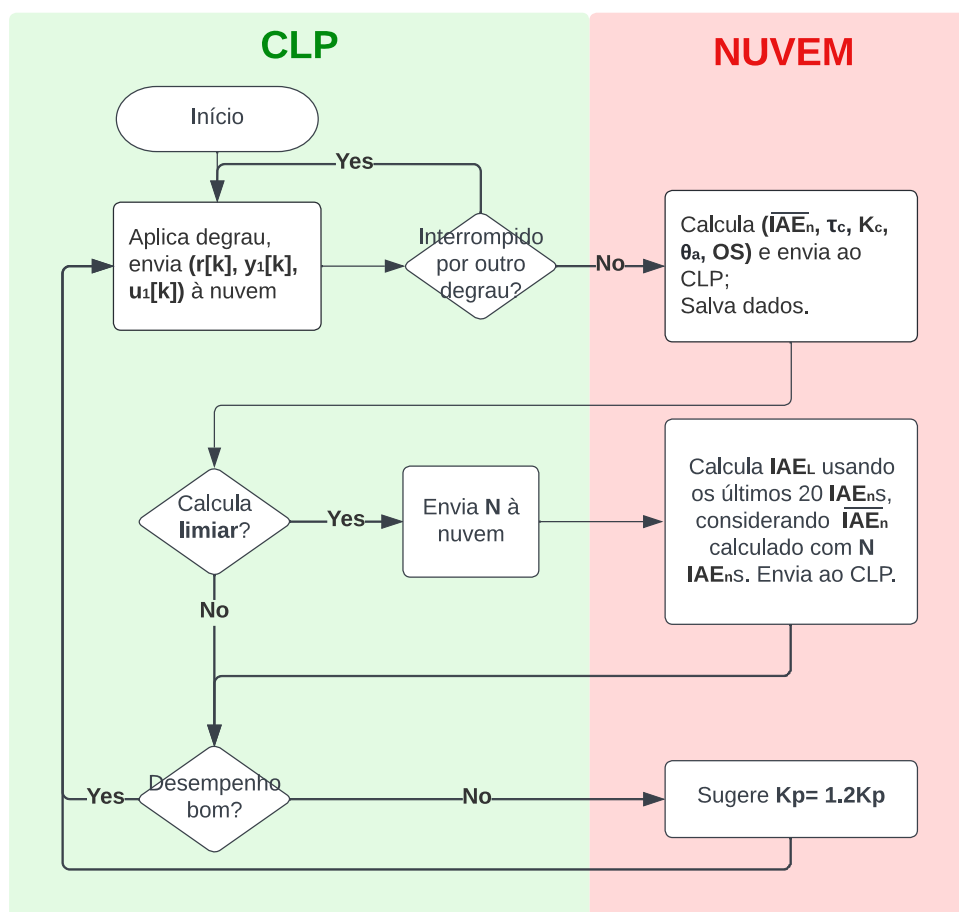
resintonia iterativa. A partir de uma mudança gradual nos parâmetros do controlador, o desempenho é monitorado até que retorne ao patamar desejado.

Logo, para validação da metodologia utilizando o método IMC, um conjunto de modelos de normalidade devem ser comparados estatisticamente com um conjunto de modelos de falha. Caso sejam estatisticamente diferentes, tal metodologia pode ser utilizada. Contudo, se os modelos forem estatisticamente iguais, possivelmente o desempenho não retornará aos valores desejados. Sendo assim, faz-se necessária a utilização da metodologia de Jelali (2012).

3.5 Monitoramento local e em nuvem

Nesta seção serão explicados todos os procedimentos necessários para a integração do CLP e a nuvem. O fluxograma presente na Figura 16 resume o funcionamento da solução e pode ser utilizado para auxílio do entendimento do conteúdo a ser explicado nessa seção.

Figura 16 – Fluxograma da solução proposta.



Fonte: Produção do próprio autor.

3.5.1 Mastertool

O primeiro passo para a implementação do monitoramento local e em nuvem é o envio de determinados conjuntos de dados à nuvem durante mudanças de referência via protocolo MQTT. O intervalo de tempo Δt para envio de $(r[k], u_1[k], y_1[k])$ é escolhido baseado na constante de tempo em malha fechada τ_c do processo, informada pelo operador, a partir de $\Delta t = t_r + 10\tau_c$, onde t_r é o instante de aplicação da mudança de referência. O parâmetro t_r é utilizado para que sejam enviados à nuvem dados em regime antes da mudança de referência, já que, caso contrário, pode-se haver problemas na estimação do modelo em malha fechada.

Para que a nuvem possa estimar os modelos em malha fechada e calcular o IAE, é necessário que a mesma tenha conhecimento do tempo de amostragem T_s . Ademais, o intervalo de amostras utilizadas para o cálculo do IAE é baseada no tempo de estabelecimento t_s . Considera-se que $t_s = \theta_a + 5\tau_c$ para sistemas FOPDT, sendo θ_a o tempo morto aparente do sistema. Como $5\tau_c \gg \theta_a$ utiliza-se, para simplificação $t_s = 5\tau_c$. Sendo assim, para cálculo do IAE, também é necessário que a nuvem conheça a constante de tempo de malha fechada τ_c . Ambos T_s e τ_c são publicados, portanto, à nuvem.

Além disso, é possível o envio de *flags* pelo Mastertool à nuvem para a requisição de novos treinamentos do monitoramento de desempenho e para sugestões de resintonia. Para novos treinamentos, o envio do tamanho da janela deslizante a ser utilizada é realizado. Já para sugestões de resintonia, a constante de tempo de malha fechada desejada λ é enviada à nuvem.

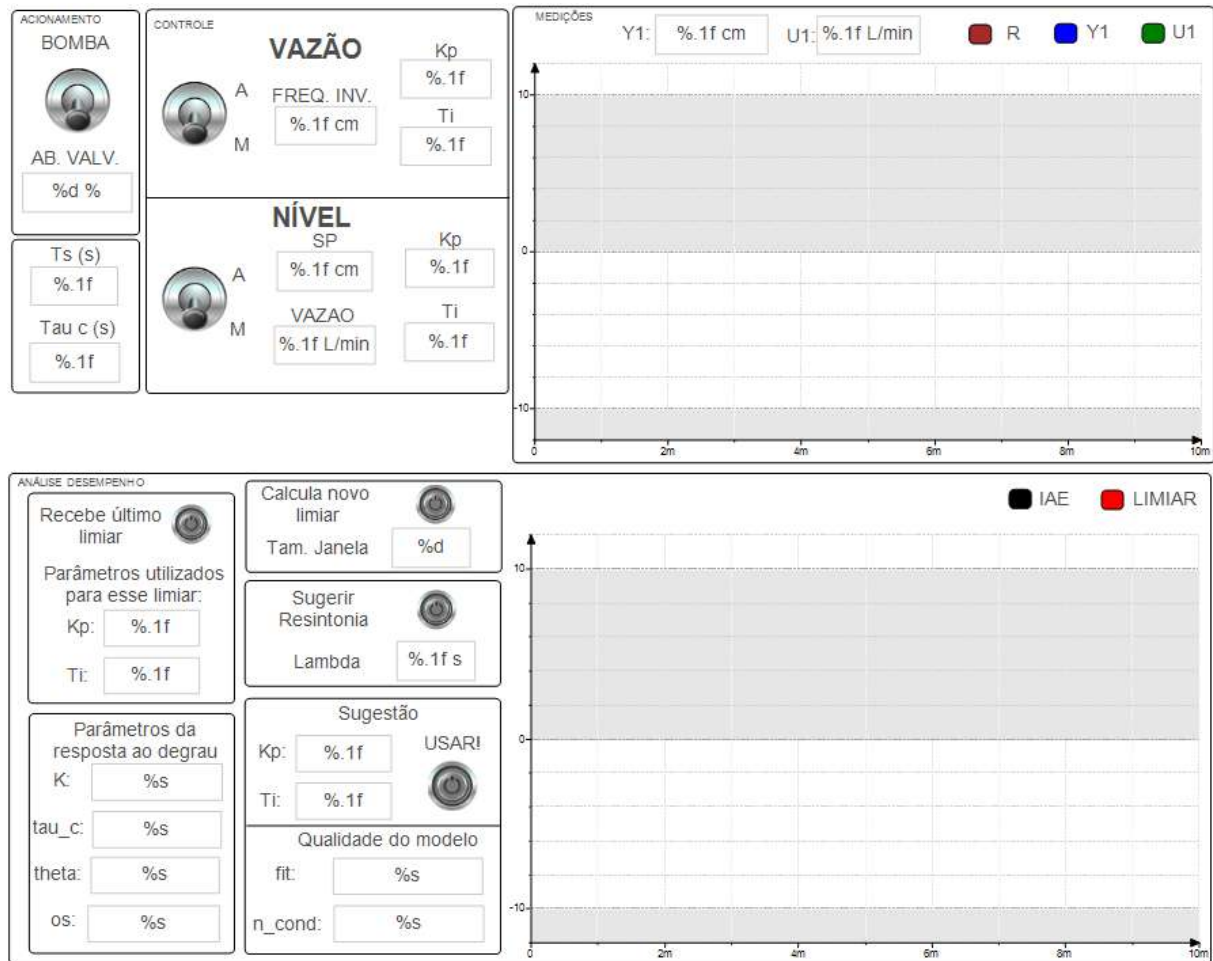
Por outro lado, a inscrição nos diversos tópicos utilizados para envio de dados da nuvem ao CLP é necessária. Os parâmetros do modelo de malha fechada: ganho, constante de tempo, atraso, sobre-elevação, *fit* e número de condição; o ganho proporcional e o tempo integral do controlador resintonizado; e o IAE são os tópicos inscritos pelo CLP.

A Figura 17 mostra a tela de operação criada, onde as variáveis recebidas são visualizadas e os parâmetros determinados podem ser enviados à nuvem.

3.5.2 Nuvem

No trabalho presente, foi utilizada uma nuvem do tipo SaaS, o intermediador usado para a implementação do protocolo MQTT; e IaaS, a máquina virtual em nuvem onde ele foi instalado e executado e a rotina em Python será executada. O sistema operacional Ubuntu foi utilizado. A provedora selecionada para o serviço foi a DigitalOcean, por ter apresentado a melhor relação custo-benefício para esta aplicação. Este provedor também

Figura 17 – Tela de operação criada.



Fonte: Produção do próprio autor.

foi escolhido em (ARÉVALO; DIPRASETYA; SCHWUNG, 2018).

O acesso ao terminal da máquina foi realizado pelo SSH nativo do Windows. A partir do acesso ao terminal o intermediador mosquito foi instalado. Para que o processo do intermediador não fosse finalizado ao fechar o terminal, foi utilizado o comando `setsid` seguido do comando para execução do mosquito.

Em nuvem, a rotina de programação em Python foi criada e executada para auxiliar o CLP no monitoramento de desempenho das malhas de controle. O algoritmo em Python é basicamente um cliente MQTT, que recebe *flags* de comando do CLP e executa determinadas funções. Ele é responsável por estimar modelos do sistema, calcular novos ganhos para os controladores, armazenar os dados, e enviar as informações necessárias ao CLP para auxiliar o operador em suas tarefas.

O cliente MQTT é gerado a partir da biblioteca paho-mqtt. Ele funciona de forma passiva, interrogando os tópicos assinados continuamente, e respondendo ao CLP apenas quando solicitado, ao receber determinadas *flags*. Esse processo de recebimento de mensagens é realizado pela função principal do programa, que, baseada no tópico da mensagem recebida, realiza determinadas tarefas.

A cada mudança de referência os dados $(r[k], u_1[k], y_1[k])$ são enviados à nuvem e a rotina armazena-os em vetores. Ao final do envio dos dados, o CLP envia uma *flag* que carrega o número N de IAE_{*n*}s utilizados para cálculo de $\overline{\text{IAE}}_n$. Ao recebe-lá, os dados são armazenados e enviados para processamento.

Inicialmente são calculados alguns parâmetros da resposta do sistema: o IAE_{*n*} (ver Equação 2.27 e 3.1), o ganho do modelo de malha fechada K_c (ver Equação 2.16), a constante de tempo do modelo de malha fechada τ_c (ver Equação 2.15), o sobressinal da resposta ao degrau *OS* (ver Equação 2.28) e o tempo morto aparente do processo θ_a . Este último pode ser calculado de diversas formas. Uma maneira muito comum de definir θ_a é o tempo para que a saída alcance uma porcentagem ζ do valor em regime, conforme

$$\theta_a = t(\zeta y_f), \quad (3.12)$$

sendo ζ um parâmetro a definir. Nota-se a partir da Equação 2.24 que a escolha de θ_a influencia diretamente na sintonia do controlador. Além disso, o IAE recém calculado, juntamente com os últimos $N - 1$ IAEs são utilizados para o cálculo de $\overline{\text{IAE}}_n$. Por fim, os parâmetros $[\text{IAE}_n, K_c, \tau_c, \theta, OS]$ são enviados ao CLP e $X = [\text{IAE}, K_c, \tau_c, \theta, OS]$ são armazenados em nuvem.

Além dos parâmetros da resposta ao degrau, a cada mudança de referência, o modelo de malha aberta do processo é estimado em malha fechada. Esse processo é realizado via método dos mínimos quadrados, conforme mostrado na subseção 2.3.1. A equação final onde os parâmetros do modelo discreto são obtidos é mostrado na Equação 2.9. Os índices de avaliação do modelo, *fit* (ver Equação 2.10) e o número de condição κ (ver Equação 2.11) são então calculados (ver subseção 2.3.2). Ao final, os dados $M = [K, \tau, fit, \kappa]$ são salvos na máquina virtual na nuvem.

Caso o operador deseje calcular um novo limiar IAE_{*L*} (ver Equação 3.5), ele envia uma determinada *flag* para a nuvem, contendo o número de IAE_{*n*}s N utilizados para o cálculo de $\overline{\text{IAE}}_n$. Na nuvem, a nova média μ_0 e desvio padrão S^2 da população são calculados usando os últimos L valores de IAE_{*n*}. Além disso, é necessário a escolha do coeficiente

de confiança α . Baseado em testes empíricos para o processo trabalhado, α igual a 1% apresentou bons resultados, que serão mostrados no capítulo a seguir. O novo valor de IAE_L é calculado a partir da detenção dos parâmetros e variáveis obtidos, e, ao final, é armazenado em nuvem e enviado ao CLP.

Tratando-se de resintonia, a qualquer momento o operador pode requerer uma sugestão de sintonia da nuvem. Para isso, ele envia uma *flag* à nuvem com a constante de tempo em malha fechada desejada λ . Em nuvem, a mediana de K , τ , θ_a , fit e κ dos últimos cinco modelos salvos são calculados. A partir das medianas \tilde{K} , $\tilde{\tau}$ e $\tilde{\theta}_a$ os parâmetros do controlador PI podem ser computados, conforme a Equação 2.24. Ao final do processo, os dados $T = [\tilde{K}, \tilde{\tau}, \tilde{\theta}_a, \tilde{\kappa}, \tilde{fit}, K_p, T_i, \lambda]$ são armazenados em nuvem e os dados $[K_p, T_i, \tilde{\kappa}, \tilde{fit}]$ são enviados ao CLP. O operador, ao analisá-los, pode decidir aceitar ou não a nova sugestão.

4 RESULTADOS E DISCUSSÕES

Neste capítulo são mostrados os resultados do trabalho proposto. Inicialmente, serão apresentadas a modelagem do sistema em malha aberta, as sintonias iniciais do controlador para diferentes valores de λ e a comparação entre a estimação de modelos em malha aberta e malha fechada. Após isso, todas as etapas de um monitoramento de desempenho de rotina serão apresentadas. Uma falha será utilizada para redução de desempenho e novas sugestões de sintonia serão aceitas. Os resultados serão expostos e discutidos.

4.1 Modelagem e Sintonia

Para a sintonia inicial do sistema, foram dados 5 degraus em malha aberta. Para cada degrau, os parâmetros do modelo calculado são mostrados no *boxplot* mostrados na Figura 18. A estimação do tempo morto θ_a foi realizada conforme a Equação 3.12, utilizando ζ igual a 2%.

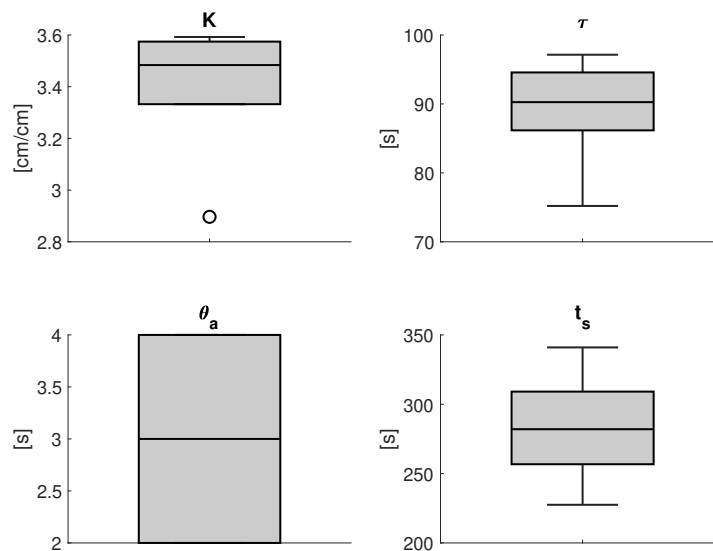


Figura 18 – Parâmetros dos modelos FOPDT obtidos a partir de respostas ao degrau em malha aberta.

Fonte: Produção do próprio autor.

Depois de calculados os modelos para cada degrau, a mediana do ganho \tilde{K} , da constante de tempo em malha aberta $\tilde{\tau}$ e do tempo morto aparente $\tilde{\theta}_a$ foram utilizados para estimação do modelo FOPDT da malha aberta do processo (ver Equação 4.1).

$$G(s) = \frac{\tilde{K}}{\tilde{\tau}s + 1} e^{-\tilde{\theta}_a s} = \frac{3,48}{90,26s + 1} e^{-3s} \quad (4.1)$$

As respostas ao degrau do sistema para as cinco mudanças de referência e para o modelo $G(s)$ são mostradas na Figura 19. Nota-se que a utilização da mediana é justificada pelo fato de que em uma das curvas (ver Figura 7) o ganho K e a constante de tempo τ do Degrau 1 menor que os demais. Isso é mostrado também na Figura 18, devido ao *outlier* em K e a amostra de menor valor de τ destoante das demais. Sendo assim, caso a média houvesse sido escolhida como medida de posição, K e τ do Degrau 1 influenciariam em $G(s)$.

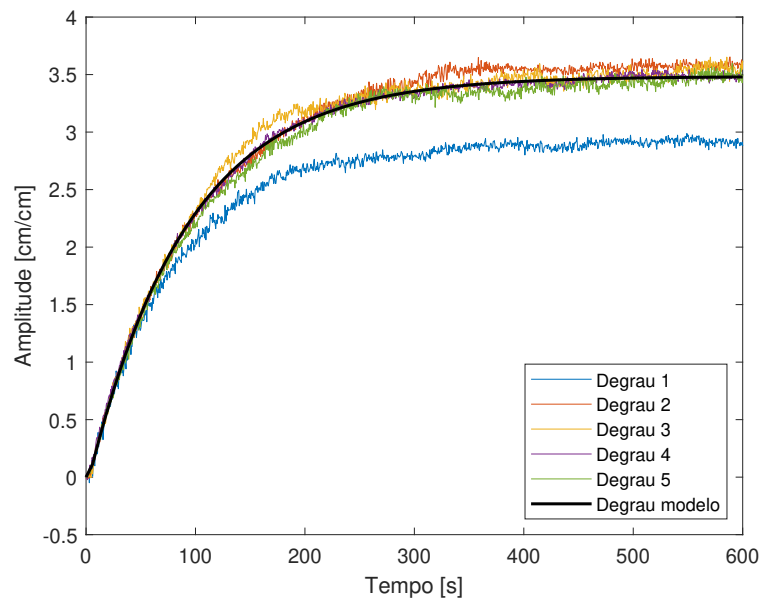


Figura 19 – Respostas ao degrau em malha aberta: dados vs. modelo obtido.

Fonte: Produção do próprio autor.

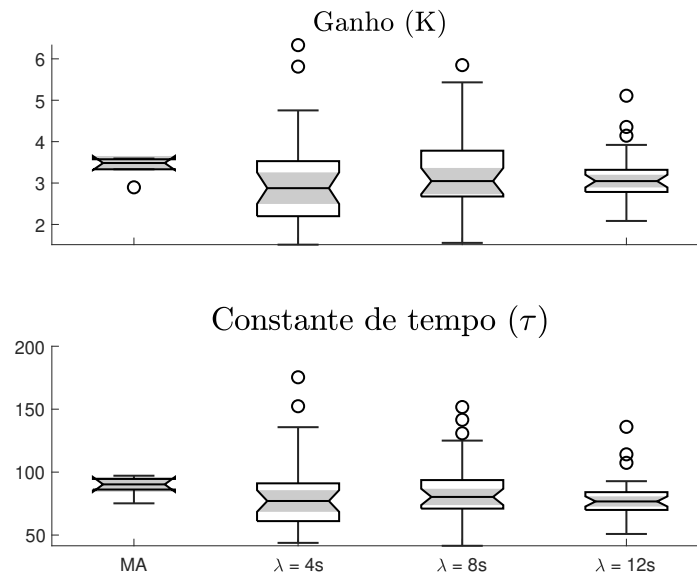
A partir do modelo $G(s)$, três controladores PI foram sintonizados utilizando o método de sintonia IMC para $\lambda = [4, 8, 12]$ (ver Equação 2.24). O resultado das três sintonias é mostrado na Tabela 2.

Tabela 2 – Controlador PI para diferentes valores de λ .

	$\lambda = 4s$	$\lambda = 8s$	$\lambda = 12s$
\mathbf{K}_p	3,70	2,36	1,72
\mathbf{T}_i	90,26	90,26	90,26

Fonte: Produção do próprio autor.

Após a sintonia em malha fechada, 30 degraus foram aplicados no processo para cada controlador ($\lambda = [4, 8, 12]$) e uma comparação entre os parâmetros K e τ dos modelos de malha aberta e fechada foi realizada (ver Figura 20).

Figura 20 – Parâmetros dos modelos em malha aberta vs. malha fechada para diferentes valores de λ .

Fonte: Produção do próprio autor.

Note que, apesar de possuírem parâmetros estatisticamente diferentes, é possível verificar semelhança entre os modelos obtidos em malha fechada em relação ao modelo obtido em malha aberta $G(s)$. Sendo assim, por experiência os modelos obtidos em malha fechada poderiam ser utilizados para sintonia caso necessário. A Figura 21 mostra as respostas ao degrau para os quatro casos utilizando a mediana das respostas aos cinco degraus para cada um destes casos. Nesta figura fica mais fácil observar as diferenças em semelhanças entre as constantes de tempo e ganho estático em cada caso.

4.2 Monitoramento de Desempenho

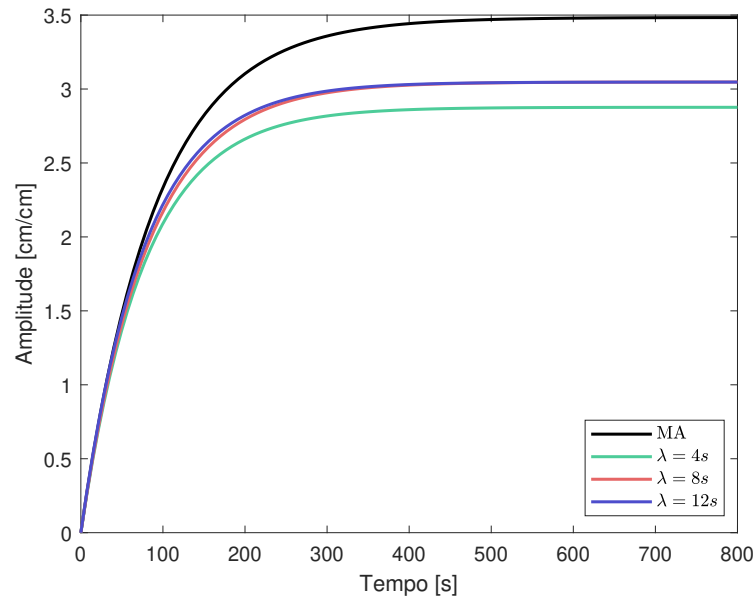
Nesta seção serão apresentadas todas as etapas de um monitoramento de desempenho de rotina. Uma falha será utilizada para redução de desempenho e novas sintonias sugeridas pela nuvem serão aceitas. Os resultados serão expostos e discutidos.

4.2.1 Treinamento e teste em normalidade

O passo inicial para viabilização do monitoramento de desempenho é o treinamento, etapa onde o limiar IAE_L é calculado a partir de

$$IAE_L = \mu_0 + t_{\alpha,v} \sqrt{\frac{S^2}{N}}. \quad (4.2)$$

Figura 21 – Respostas ao degrau do modelo obtido em malha aberta vs. malha fechada para diferentes valores de λ .



Fonte: Produção do próprio autor.

Foi utilizado o coeficiente de confiança α igual a 1% e número de graus de liberdade v igual a 4 (número de amostras para cálculo de \overline{IAE}_n menos um). Além disso, o número de IAE_n s utilizados para cálculo das medidas representativas da população - média μ_0 e variância S^2 - foi igual a 20. Assim, 20 degraus foram aplicados para treinamento, e os resultados de IAE_L para os três valores de λ são mostrados na Tabela 3.

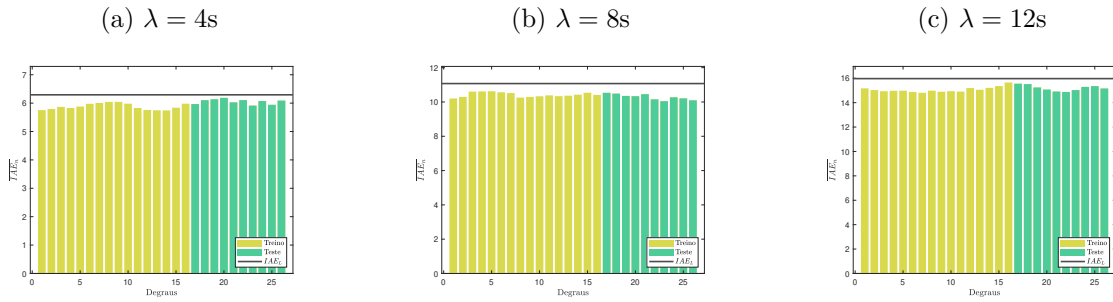
Tabela 3 – IAE_L para os 3 valores de λ .

	$\lambda = 4s$	$\lambda = 8s$	$\lambda = 12s$
IAE_L	6,29	11,07	15,97

Fonte: Produção do próprio autor.

O resultado do treinamento e do teste para normalidade é mostrado na Figura 22. As barras amarelas correspondem aos dados de treinamento e as barras verdes aos dados de teste. Pode-se ver 16 \overline{IAE}_n s de treino para 20 mudanças de referência, uma vez que as primeiras 5 mudanças de *setpoint* são necessárias para iniciar o treinamento. Como uma janela deslizante de valores de IAE_n é usada para calcular sua média, a barra número 17 contém 4 IAE_n s calculados dos últimos 4 degraus de treinamento e um IAE_n do primeiro degrau de teste e assim por diante. Nota-se que diferentes valores de λ geram diferentes limiares, o que é esperado, pois o IAE depende da constante de tempo em malha fechada.

Figura 22 – Treinamento e teste do monitoramento de desempenho para diferentes valores de λ .



Fonte: Produção do próprio autor.

4.2.2 Teste em falha

Uma falha foi introduzida a partir do fechamento de 30% da válvula pneumática c3 (ver Figura 11). O monitoramento do IAE_n é realizado a cada mudança na referência e é mostrado nas Figuras 23, 24 e 25 para os diferentes valores de λ . Além disso, nas respectivas figuras, o monitoramento de desempenho utilizando a metodologia de Swanda e Seborg (1999) também foi realizado, para efeito comparativo. Nas figuras da metodologia proposta no presente trabalho, os primeiros 6 valores de \overline{IAE}_n (barras verdes) correspondem à operação de teste em normalidade (Figura 22) e os últimos 15 valores correspondem a médias do IAE_n após a inserção da falha. Nas figuras que apresentam os resultados da metodologia de Swanda e Seborg (1999), as barras azuis representam as amostras de IAE_n para teste em normalidade, e as barras vermelhas em falha. O limiar IAE_s calculado por Swanda e Seborg (1999) é dado por

$$IAE_s = \theta_a + \lambda. \tag{4.3}$$

Como visto na Equação 4.1, o tempo morto aparente do processo θ_a é igual a 3s. Sendo assim, IAE_s s para os três valores de λ são mostrados na Tabela 4.

Tabela 4 – IAE_s para os 3 valores de λ .

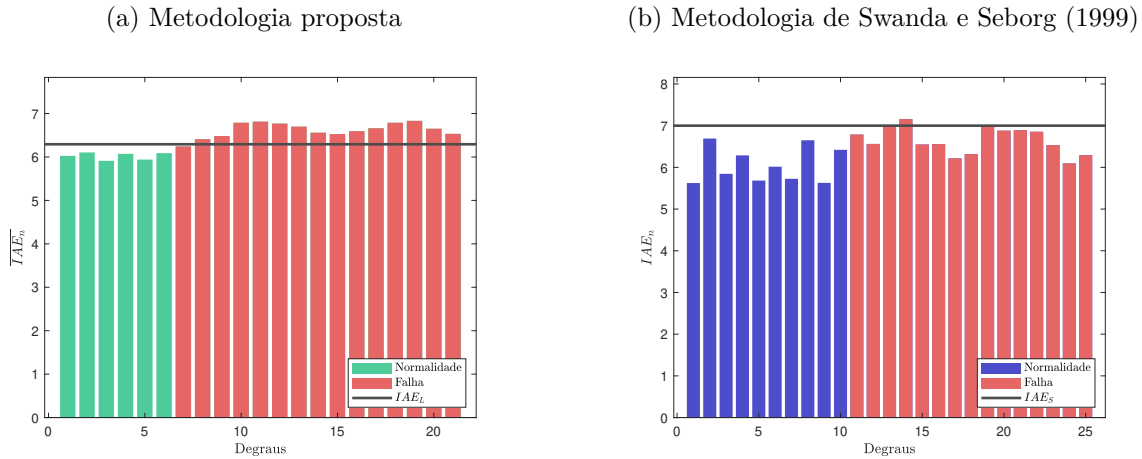
	$\lambda = 4s$	$\lambda = 8s$	$\lambda = 12s$
IAE_s	7	11	15

Fonte: Produção do próprio autor.

É importante destacar que o teste proposto por Swanda e Seborg (1999) é feito sobre cada valor de IAE_n medido, e não sobre uma média de valores, o que torna-o mais suscetível às variabilidades inerentes ao processo.

Para λ igual a 4s, percebe-se que o segundo \overline{IAE}_n em falha ultrapassou IAE_L . Utilizando a metodologia do Swanda e Seborg (1999), percebe-se que apesar de identificar assertivamente as amostras de normalidade, houveram três falsos negativos, isto é, amostras de falha (barras vermelhas) que foram identificadas como normalidade.

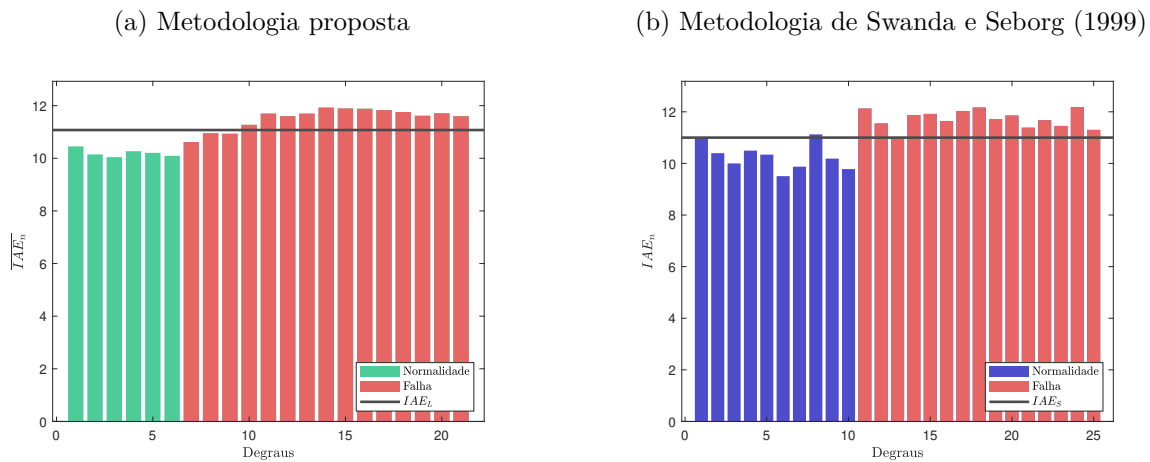
Figura 23 – Monitoramento de desempenho para situação de falha com sintonia realizada para $\lambda = 4s$.



Fonte: Produção do próprio autor.

Tratando-se de λ igual a 8s, a metodologia proposta no presente trabalho foi capaz de detectar o baixo desempenho após o quarto degrau em falha. \overline{IAE}_n nesse momento contém quatro amostras de falha e uma amostra de normalidade. Para a metodologia de Swanda e Seborg (1999), a partir do primeiro degrau aplicado em situação de falha foi possível detectar o baixo desempenho. Todavia, houveram dois erros do tipo I.

Figura 24 – Monitoramento de desempenho para situação de falha com sintonia realizada para $\lambda = 8s$.

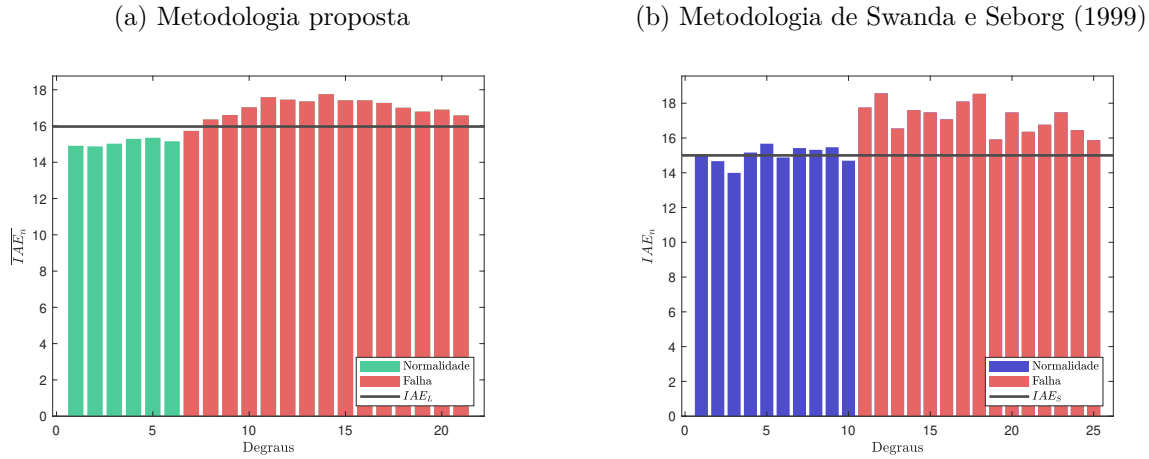


Fonte: Produção do próprio autor.

Para λ igual a 12s, utilizando a metodologia proposta foi possível identificar o baixo desempenho da malha de controle a partir do segundo degrau aplicado em situação

de falha. Para a metodologia do Swanda e Seborg (1999), nota-se que, apesar de ter identificado com sucesso todas os IAE_n s em falha, houveram cinco erros do tipo I para os 10 degraus em normalidade.

Figura 25 – Monitoramento de desempenho para situação de falha com sintonia realizada para $\lambda = 12$ s.



Fonte: Produção do próprio autor.

Como o limiar do monitoramento de Swanda e Seborg (1999) é dependente do tempo morto aparente θ_a (ver Equação 4.3), e como θ_a não é um parâmetro absoluto, ou seja, apresenta diversas formas de estimação, faz-se necessário avaliar ambas metodologias para diferentes valores de θ_a . Para os testes realizados nas Figuras 23, 24 e 25 o tempo morto foi escolhido como o tempo para a saída atingir 2% do valor de regime ($\zeta = 2\%$). Para efeito de comparação, ζ será alterado para 5%.

Quando um novo tempo morto aparente θ_a é estimado, a constante de tempo de malha aberta τ é alterada, já que o tempo gasto para alcançar 63,3% do valor em regime (T) é a soma do tempo morto aparente com a constante de tempo de malha aberta. Considerando a utilização do mesmo degrau para a estimação de modelos com θ_a s diferentes, T se mantém constante, e logo, o aumento de θ_a gera uma diminuição de τ , e vice-versa. Devido a isso, como de acordo com o método IMC para modelos FOPDT

$$K_p = \frac{\tau}{K(\lambda + \theta_a)}, \quad (4.4)$$

$$T_i = \tau,$$

um aumento de θ_a , por exemplo, gera uma diminuição no ganho proporcional K_p e no tempo integral T_i . Como o K_p em geral é o parâmetro fortemente responsável pela agressividade do controlador, θ_a apresenta uma relação inversa com a agressividade do controlador. De forma geral, salvo situações onde o sobressinal da resposta ao degrau se torna muito

elevado, a agressividade do controlador apresenta uma relação direta com o desempenho da malha de controle a ser controlada por ele. Logo, θ_a apresenta uma relação inversa com o IAE_n , já que o desempenho apresenta uma relação inversa com o IAE_n . Sendo assim, um aumento de θ_a , por exemplo, gera um aumento nos valores de IAE_n . Isso gerará um limiar IAE_L maior, devido ao aumento nos valores de IAE_n usados para treinamento, e um IAE_s maior, já que θ_a será maior.

Os mesmos degraus em malha aberta mostrados na Figura 19 foram utilizados. A mediana dos parâmetros de seus modelos geraram o modelo dado por

$$G(s) = \frac{3,48}{87,26s + 1} e^{-6s}, \quad (4.5)$$

onde θ_a é igual a 6, o dobro do valor anterior (3).

Para este modelo, o controlador foi sintonizado utilizando o método IMC para $\lambda = 8s$. Os parâmetros obtidos, comparados com o controlador PI sintonizado anteriormente para θ_a igual a 6s, são mostrados na Tabela 5.

Tabela 5 – IAE_s para os 3 valores de λ .

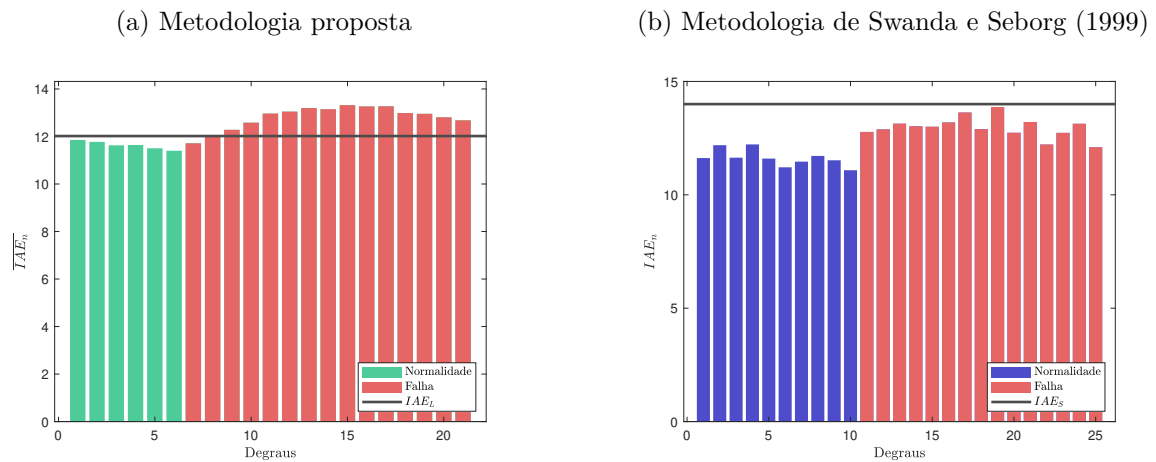
	$\theta_a = 3s$	$\theta_a = 6s$
K_p	2,36	1,81
T_i	90,26	87,26s

Fonte: Produção do próprio autor.

A Figura 26 comprova o comportamento esperado de ambos monitoramentos de desempenho para θ_a igual a 6s. Nota-se que a mudança de θ_a não alterou a qualidade do monitoramento de desempenho proposto pelo presente trabalho, que detectou a falha a partir do segundo degrau aplicado. É perceptível, entretanto, que tal mudança afetou o monitoramento de desempenho de Swanda e Seborg (1999). O mau desempenho da malha de controle operando o processo em falha não foi detectado em nenhum degrau (barras vermelhas).

A metodologia estatística proposta para monitorar o desempenho é uma solução para a variabilidade dos índices de desempenho de malhas de controle. Nota-se que, apesar da utilização do IAE normalizado pelo tamanho da mudança de referência, o resultado da metodologia de Swanda e Seborg (1999) apresentou dados de normalidade que foram considerados falha (falsos positivos) e dados de falha considerados como normalidade (falsos negativos), mesmo para λ igual a 8, onde esta apresentou melhores resultados. Além

Figura 26 – Monitoramento de desempenho para situação de falha com sintonia realizada para $\lambda = 8s$ e $\theta_a = 6s$.



Fonte: Produção do próprio autor.

disso, apesar de diminuir a variabilidade, o monitoramento da média retardada a detecção de mudanças de comportamento. Entretanto, como o maior interesse de boa parte dos processos industriais é na redução de falsos positivos e negativos, isso pode não ser um problema. Contudo, cabe uma avaliação individual para cada processo, para ajuste do número de valores utilizados no cálculo da média da métrica de desempenho utilizada.

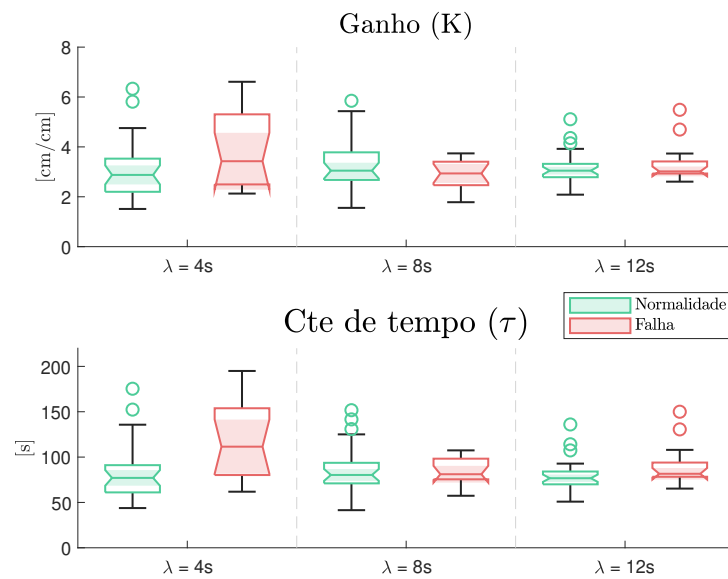
Outro ponto a ser ressaltado é a capacidade de generalização da solução proposta. É perceptível que a solução apresentou bons resultados independentemente do λ utilizado na sintonia do controlador e do tempo morto aparente θ_a estimado, ao contrário dos resultados do trabalho de Swanda e Seborg (1999), onde ambos afetaram a qualidade do monitoramento.

4.3 Resintonia

Uma vez detectado o baixo desempenho, uma das alternativas para recuperá-lo é fazer a resintonia dos controladores, usando para isto o modelo em falha obtido, utilizando o mesmo λ . A Figura 27 mostra os parâmetros dos modelos de normalidade e falha para 15 degraus, para os três valores de λ analisados. Pode-se concluir, com 95% de confiança, que os parâmetros do modelo em malha fechada para normalidade e falha são estatisticamente iguais, já que os entalhes dos *boxplots* se sobrepõem para um mesmo parâmetro e valor de λ .

Além disso, por experiência, a partir da análise das respostas ao degrau médias para os três valores de λ em normalidade e falha (ver Figura 28), é possível notar que os modelos são similares. Nela, as respostas ao degraus foram normalizadas e a média de

Figura 27 – Parâmetros dos modelos obtidos em normalidade e falha.



Fonte: Produção do próprio autor.

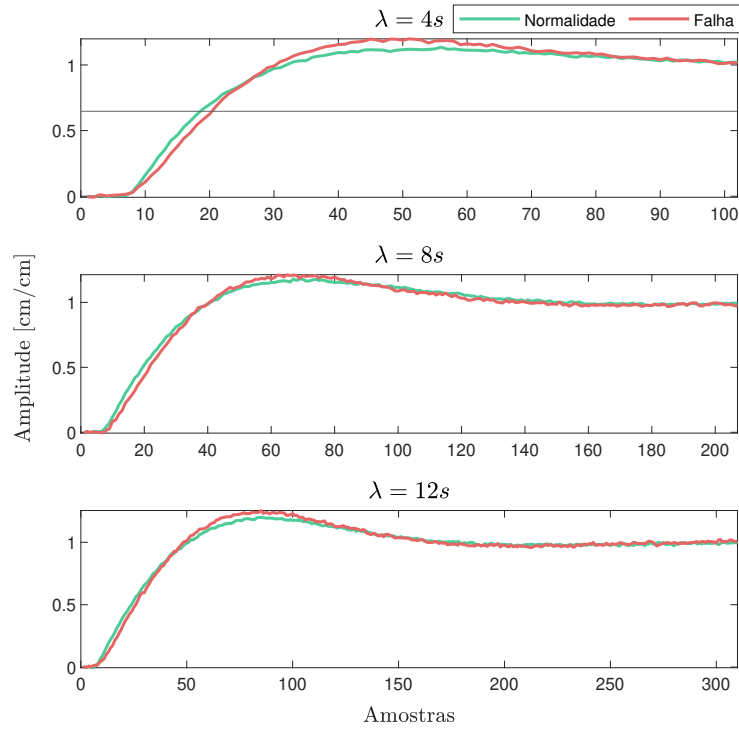
seus pontos para cada instante de tempo foi plotada. Assim, conclui-se que, apesar de afetar o desempenho, a falha inserida na malha de controle não afeta significativamente o modelo do processo, sendo necessária a utilização da metodologia de Jelali (2012) para a recuperação do desempenho. De fato, testes foram realizados e a melhora de desempenho necessária não foi obtida.

Jelali (2012) propôs diferentes mudanças iterativas no parâmetros do controlador PI para a recuperação de desempenho. Uma delas é realizada apenas no ganho proporcional K_p , aumentando-o de 20% a 50% em cada iteração. No presente trabalho, foram realizados aumentos consecutivos de 20%, até que \overline{IAE}_n retornasse fosse menor que o limiar IAE_L .

As Figuras 29, 30 e 31 mostram a recuperação de desempenho para os três valores de λ (4s, 8s e 12s, respectivamente). Para λ igual a 4s, a primeira tentativa não conseguiu recuperar o desempenho totalmente (barras amarelas), embora seja notada alguma melhora. Devido a isso, portanto, foi realizada o segundo ajuste no valor de K_p , que, por sua vez, recuperou completamente o desempenho (barras azuis).

O mesmo ocorreu para λ igual a 8s. O primeiro ajuste de K_p para recuperação de desempenho não foi suficiente (barras amarelas). Sendo assim, um reajuste foi necessário para que o desempenho retornasse ao patamar de normalidade (barras azuis com valores próximos às barras verdes).

Figura 28 – Respostas ao degrau médias para diferentes valores de λ : normalidade vs. falha.

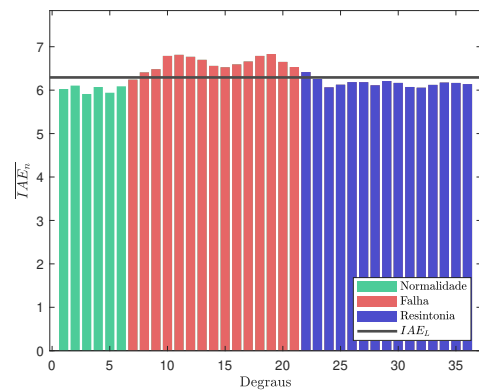
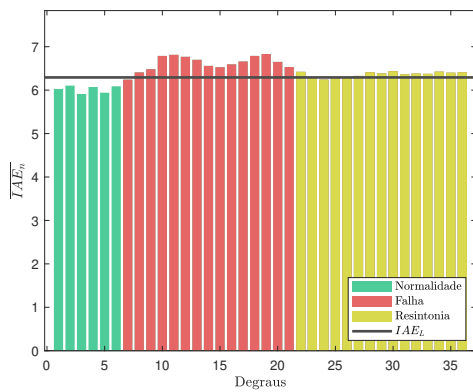


Fonte: Produção do próprio autor.

Figura 29 – Resintonia para $\lambda = 4s$.

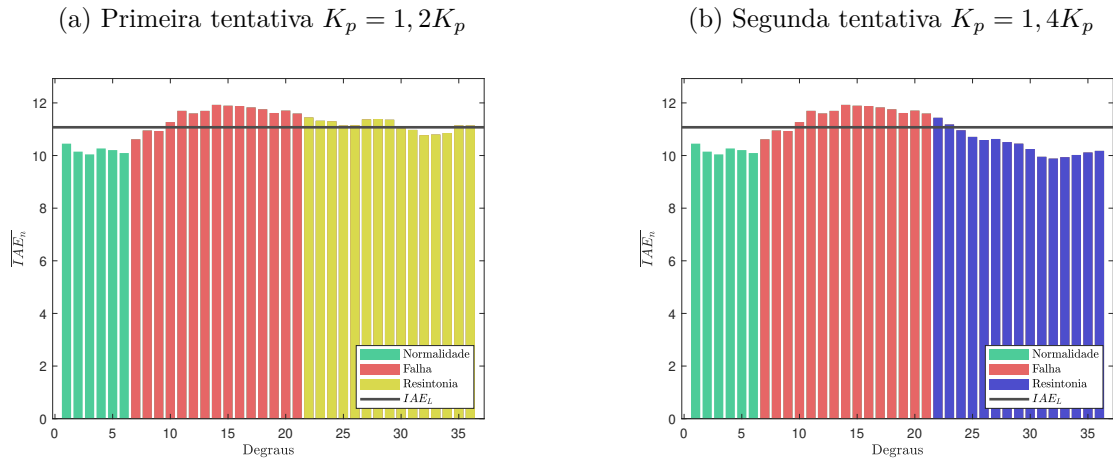
(a) Primeira tentativa $K_p = 1, 2K_p$

(b) Segunda tentativa $K_p = 1, 4K_p$



Fonte: Produção do próprio autor.

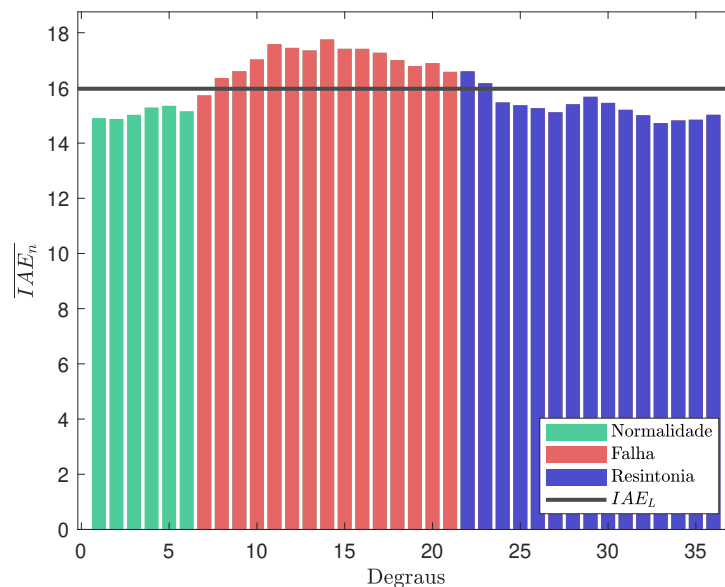
Figura 30 – Resintonia para $\lambda = 8s$.



Fonte: Produção do próprio autor.

Já para λ igual a 12s, apenas uma iteração foi necessária. O desempenho foi plenamente recuperado ao multiplicar-se K_p por 1, 2.

Figura 31 – Resintonia para $\lambda = 12s$. Primeira tentativa: $K_p = 1, 2K_p$



Fonte: Produção do próprio autor.

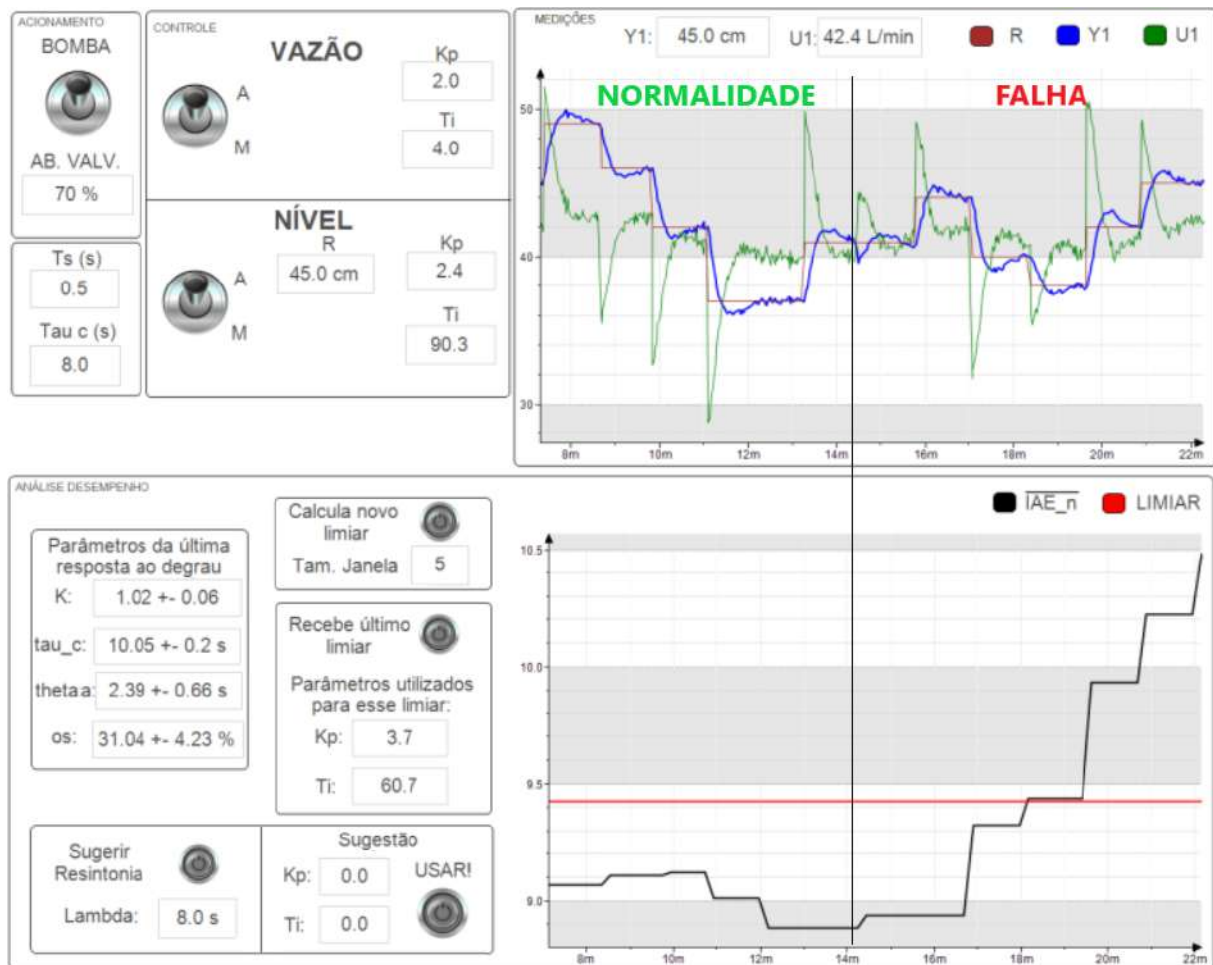
A proposta de resintonia exposta no presente trabalho realizou com sucesso a recuperação de desempenho da malha de controle para diferentes controladores sintonizados inicialmente (valores de λ). Percebe-se também que para alguns controladores a proposta necessita de mais iterações para recuperar o desempenho, sendo necessário mais degraus aplicados e conseqüentemente mais tempo de operação. Entretanto, sabe-se que diversas malhas de controle após inicialmente sintonizadas geram prejuízos financeiros por apresentarem

mau desempenho e não serem monitoradas (STARR; PETERSEN; BAUER, 2016), o que valida a vantagem de utilização da solução proposta.

4.4 Interface para operação local e em nuvem

Nessa seção será mostrada uma operação de rotina utilizando a tela de operação criada no Mastertool (ver Figuras 17 e 32).

Figura 32 – Tela de operação



Fonte: Produção do próprio autor.

Na seção de Acionamento é possível ativar a bomba centrífuga e modificar a abertura da válvula pneumática c3 (ver Figura 11). Na seção de controle, é possível alternar entre os modos automático e manual para as malhas de vazão e nível, alterar os ganhos de seus controladores PI, e modificar a referência de nível $r[k]$. Caso a malha de nível esteja em modo manual, é possível alterar a referência da malha de vazão $u_1[k]$. Por fim, se ambas as malhas estejam em modo manual, é possível alterar a frequência do inversor da bomba.

Abaixo da seção de acionamento é possível modificar o tempo de amostragem T_s e a constante de tempo de malha fechada τ_c , variáveis necessárias para o cálculo dos parâmetros da resposta ao degrau. Na região superior direita da tela, é possível analisar a referência de nível $r[k]$, o nível $y_1[k]$ e o sinal de controle $u_1[k]$. Uma janela de 15 minutos é utilizada, que pode ser alterada para diferentes aplicações.

A parte inferior da tela é responsável pelo monitoramento de desempenho. Após o envio de cada resposta ao degrau à nuvem, os parâmetros da resposta são recebidos. Além disso, é possível calcular um novo limiar IAE_L , indicando o tamanho da janela utilizada para cálculo de $\overline{IAE_n}$. Caso tenha ocorrido pausa na operação, queda de energia ou reinício do CLP, é possível recuperar o último limiar calculado em nuvem, juntamente com os parâmetros do controlador utilizados no cálculo desse limiar.

Se for notado baixo desempenho da malha de nível, é possível requerer uma nova sintonia, que será recebida na aba de sugestão. Ao clicar em "USAR!" o operador concorda com a sugestão proposta e modifica os ganhos atuais do controlador de nível $C_1(z^{-1})$.

No instante 15min (linha preta vertical) a abertura de c3 é reduzida para 70%. Percebe-se que a partir do terceiro degrau do processo em falha é possível detectar a falha a partir do gráfico de monitoramento de desempenho.

Por fim, na rotina em nuvem é possível alterar parâmetros do sistema não disponíveis para modificação no CLP. São eles: o coeficiente de confiança α , o número de constantes de tempo τ_c utilizadas para o cômputo de IAE_n , a quantidade de IAE_n s utilizados para cálculo da média μ_0 e da variância S^2 da população, e o parâmetro ζ utilizado no cálculo do tempo morto aparente θ_a . Assim, tais parâmetros podem ser escolhidos de acordo com o processo a se monitorar.

5 CONCLUSÃO

Este trabalho apresentou uma proposta para a integração entre sistemas SCADA e plataformas de nuvem. Ademais, uma metodologia para monitoramento do desempenho foi também discutida e apresentada. O objetivo foi obter uma integração confiável para transmissão de dados entre o CLP e a nuvem, e também uma metodologia robusta para o monitoramento de desempenho, pouco sensível às variabilidades inerentes ao processo, com rápida capacidade de detecção de anormalidades e passível de recuperação do desempenho comprometido.

A partir dos resultados apresentados é possível notar que os objetivos inicialmente almejados foram alcançados. Inicialmente, a comunicação entre o CLP e a nuvem apenas durante a mudança de referências foi realizada, o que gerou maior segurança caso comparado ao envio irrestrito de dados via Internet. É válido destacar que, ainda assim, há o problema relacionado à cibersegurança, já que os dados dessa forma ainda saem do domínio da rede da empresa. Devido a isso, o uso de computação em nuvem interna pode ser considerado. Em suma, uma tela de operação pode ser criada para interação entre o CLP e a nuvem. Nela o operador pode realizar a operação da malha local, alterando referências e parâmetros do controlador, como também pode interagir com a nuvem, recebendo valores de IAE_n e requisitando novos limiares para a carta de controle responsável pelo monitoramento de desempenho.

Adicionalmente, a metodologia estatística para o monitoramento de desempenho utilizando testes de hipótese para análise do IAE normalizado foi implementada, o que, a partir da análise de médias de amostras da população, gerou a robustez desejada. Por fim, ao utilizar a metodologia proposta por Jelali (2012), foi possível recuperar o desempenho da malha mesmo quando a mudança da dinâmica do sistema que causou o mau desempenho não afeta significativamente o modelo. É interessante ressaltar que tal metodologia é flexível quanto ao grau de aumento do ganho proporcional e se é gerado aumento ou redução de tal parâmetro, já que, eventualmente, pode ser necessária uma redução do ganho proporcional para recuperação do desempenho.

Ademais, uma comparação entre os resultados da metodologia para análise de desempenho proposta por Swanda e Seborg (1999) e do presente trabalho foi feita. Foi possível notar que a metodologia proposta apresentou robustez em relação aos diferentes níveis de agressividade do controlador, além de não depender da estimação de parâmetros do modelo, o que não ocorreu para a metodologia de Swanda e Seborg (1999). Além disso, foi possível notar que, por utilizar uma média móvel para avaliação do desempenho, a detecção de mau desempenho pela metodologia proposta pelo presente trabalho foi mais lenta, demorando

cerca de 2 a 3 degraus ao se utilizar uma média móvel de 5 amostras. Entretanto, como a periodicidade dessas análises é grande (dias ou semanas), geralmente, isso não apresenta problema. Tratando-se de desempenho, em boa parte das soluções um baixo número de falsos positivos é mais importante que uma detecção mais rápida.

É possível perceber, devido a isso, que parâmetros como o número de amostras para cálculo da média móvel devem ser avaliados e escolhidos para diferentes tipos de processo e objetivo final da aplicação. Ao se demandar uma detecção mais rápida porém com maior variabilidade, pode-se diminuir o número de amostras para cálculo da média, por exemplo.

Não obstante, a solução proposta pode ser aprimorada, para gerar, além da detecção de mau desempenho, as possíveis causas do problema. Estudos apresentados por Jelali (2012) mostraram comportamentos comuns para diferentes tipos de distúrbios que geram mau desempenho, e possíveis formas de diagnóstico destes. Além disso, o trabalho presente apresentou apenas solução para o problema de controle servo. Futuramente, a abrangência da solução para problemas de controle regulatório, provocado por distúrbios de carga, pode ser realizada.

Em síntese, portanto, é possível concluir que os resultados indicam que a metodologia proposta pelo presente trabalho é promissora para a aplicação em sistemas SCADA, sendo passível de generalização para diferentes tipos de processos industriais e índices de avaliação de desempenho.

REFERÊNCIAS

- ALI, A. M.; MOHAMED, E.-A.; YACOUT, S.; SHABAN, Y. Cloud computing based unsupervised fault diagnosis system in the context of industry 4.0. Gestão & Produção, SciELO Brasil, v. 27, 2020. Citado na página 34.
- ARÉVALO, F.; DIPRASETYA, M. R.; SCHWUNG, A. A cloud-based architecture for condition monitoring based on machine learning. In: IEEE. 2018 IEEE 16th International Conference on Industrial Informatics (INDIN). [S.l.], 2018. p. 163–168. Citado na página 47.
- ÅSTRÖM, K. J.; HÄGGLUND, T.; ASTROM, K. J. Advanced PID control. [S.l.]: ISA-The Instrumentation, Systems, and Automation Society Research Triangle Park, 2006. v. 461. Citado 4 vezes nas páginas 26, 30, 31 e 32.
- BAUER, M.; HORCH, A.; XIE, L.; JELALI, M.; THORNHILL, N. The current state of control loop performance monitoring—a survey of application in industry. Journal of Process Control, Elsevier, v. 38, p. 1–10, 2016. Citado 2 vezes nas páginas 13 e 29.
- BAUER, M.; SCHLAKE, J. C. Changes to the automation architecture: Impact of technology on control systems algorithms. In: IEEE. 2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA). [S.l.], 2017. p. 1–8. Citado na página 19.
- BORASE, R. P.; MAGHADE, D.; SONDKAR, S.; PAWAR, S. A review of pid control, tuning methods and applications. International Journal of Dynamics and Control, Springer, v. 9, n. 2, p. 818–827, 2021. Citado na página 26.
- CAPACI, R. Bacci di; SCALI, C. A cloud-based monitoring system for performance assessment of industrial plants. Industrial & Engineering Chemistry Research, ACS Publications, v. 59, n. 6, p. 2341–2352, 2020. Citado 3 vezes nas páginas 13, 19 e 34.
- CHOUDHURY, M. S.; SHAH, S. L.; THORNHILL, N. F. Diagnosis of poor control-loop performance using higher-order statistics. Automatica, Elsevier, v. 40, n. 10, p. 1719–1728, 2004. Citado na página 34.
- CLARK, D. The iot of automation—a totally new way to look at control and automation in the process industries. In: Proc. LCCC Workshop Process Control. [S.l.: s.n.], 2016. Citado na página 19.
- DESBOROUGH, L.; MILLER, R. Increasing customer value of industrial control performance monitoring-honeywell’s experience. In: NEW YORK; AMERICAN INSTITUTE OF CHEMICAL ENGINEERS; 1998. AICHE symposium series. [S.l.], 2002. p. 169–189. Citado na página 34.
- DIJKSTRA, E. W. Self-stabilizing systems in spite of distributed control. Communications of the ACM, ACM New York, NY, USA, v. 17, n. 11, p. 643–644, 1974. Citado na página 19.

Eclipse Foundation. mosquitto. 2009. Acessado em: 14 de maio de 2021. Disponível em: <<https://mosquitto.org/>>. Citado na página 17.

FOLEY, M. W.; JULIEN, R. H.; COPELAND, B. R. A comparison of pid controller tuning methods. The Canadian Journal of Chemical Engineering, Wiley Online Library, v. 83, n. 4, p. 712–722, 2005. Citado na página 26.

FORSSELL, U.; LJUNG, L. Closed-loop identification revisited. Automatica, Elsevier, v. 35, n. 7, p. 1215–1241, 1999. Citado na página 20.

FRUEHAUF, P. S.; CHIEN, I.-L.; LAURITSEN, M. D. Simplified imc-pid tuning rules. ISA Transactions, Elsevier, v. 33, n. 1, p. 43–59, 1994. Citado na página 43.

GAO, X.; YANG, F.; SHANG, C.; HUANG, D. A review of control loop monitoring and diagnosis: Prospects of controller maintenance in big data era. Chinese Journal of Chemical Engineering, Elsevier, v. 24, n. 8, p. 952–962, 2016. Citado na página 34.

GARCIA, C. E.; MORARI, M. Internal model control. a unifying review and some new results. Industrial & Engineering Chemistry Process Design and Development, ACS Publications, v. 21, n. 2, p. 308–323, 1982. Citado na página 26.

GARCIA, C. E.; MORARI, M. Internal model control. 2. design procedure for multivariable systems. Industrial & Engineering Chemistry Process Design and Development, ACS Publications, v. 24, n. 2, p. 472–484, 1985. Citado na página 26.

GARCIA, C. E.; MORARI, M. Internal model control. 3. multivariable control law computation and tuning guidelines. Industrial & Engineering Chemistry Process Design and Development, ACS Publications, v. 24, n. 2, p. 484–494, 1985. Citado na página 26.

GAUSS, C. F. Least squares. [S.l.]: Werke, 1809. Citado na página 22.

GIVEHCHI, O.; IMTIAZ, J.; TRSEK, H.; JASPERNEITE, J. Control-as-a-service from the cloud: A case study for using virtualized plcs. In: IEEE. 2014 10th IEEE Workshop on Factory Communication Systems (WFCS 2014). [S.l.], 2014. p. 1–4. Citado na página 19.

GONG, C.; LIU, J.; ZHANG, Q.; CHEN, H.; GONG, Z. The characteristics of cloud computing. In: IEEE. 2010 39th International Conference on Parallel Processing Workshops. [S.l.], 2010. p. 275–279. Citado na página 13.

GRELEWICZ, P.; KHUAT, T. T.; CZECZOT, J.; KLOPOT, T.; GABRYS, B. Application of machine learning to performance assessment for a class of pid-based control systems. arXiv preprint arXiv:2101.02939, 2021. Citado na página 13.

HÄGGLUND, T. A control-loop performance monitor. Control Engineering Practice, Elsevier, v. 3, n. 11, p. 1543–1551, 1995. Citado na página 34.

HARRIS, T. J. Assessment of control loop performance. The Canadian Journal of Chemical Engineering, Wiley Online Library, v. 67, n. 5, p. 856–861, 1989. Citado na página 31.

HUANG, B. Bayesian methods for control loop monitoring and diagnosis. Journal of process control, Elsevier, v. 18, n. 9, p. 829–838, 2008. Citado na página 34.

IBM. Qualities of service provided by an MQTT client. 2021. Acessado em: 14 de maio de 2021. Disponível em: <<https://www.ibm.com/docs/en/ibm-mq/8.0?topic=concepts-quality-service>>. Citado na página 16.

IBM; EUROTECH. MQTT V3.1 Protocol Specification. s.d. Acessado em: 14 de maio de 2021. Disponível em: <<https://public.dhe.ibm.com/software/dw/webservices/ws-mqtt/mqtt-v3r1.html>>. Citado na página 15.

JELALI, M. An overview of control performance assessment technology and industrial applications. Control engineering practice, Elsevier, v. 14, n. 5, p. 441–466, 2006. Citado na página 30.

JELALI, M. Control performance management in industrial automation: assessment, diagnosis and improvement of control loop performance. Springer Science & Business Media, 2012. Citado 6 vezes nas páginas 13, 44, 45, 59, 64 e 65.

KASHYAP, M.; SHARMA, V.; GUPTA, N. Taking mqtt and nodemcu to iot: communication in internet of things. Procedia computer science, Elsevier, v. 132, p. 1611–1618, 2018. Citado na página 15.

LEGENDRE, A. M. Nouvelles méthodes pour la détermination des orbites des comètes; par AM Legendre... [S.l.]: chez Firmin Didot, libraire pour lew mathematiques, la marine, l . . . , 1806. Citado na página 22.

LYNCH, C.; DUMONT, G. A. Control loop performance monitoring. IEEE transactions on control systems technology, IEEE, v. 4, n. 2, p. 185–192, 1996. Citado na página 34.

MANDLOI, R.; SHAH, P. Methods for closed loop system identification in industry. Journal of Chemical and Pharmaceutical Research, v. 7, n. 1, p. 892–896, 2015. Citado na página 21.

MERCADÉ, D. Comparison of different Internet of Things platforms. Dissertação (B.S. thesis) — Universitat Politècnica de Catalunya, 2018. Citado na página 18.

MISHRA, B.; MISHRA, B. Evaluating and Analyzing MQTT Brokers with Stress-testing. [S.l.]: Article, 2020. Citado na página 17.

MORARI, M.; ZAFIRIOU, E. Robust process control. [S.l.]: Morari, 1989. Citado na página 26.

PROFANTER, S.; TEKAT, A.; DOROFEEV, K.; RICKERT, M.; KNOLL, A. Opc ua versus ros, dds, and mqtt: performance evaluation of industry 4.0 protocols. In: IEEE. 2019 IEEE International Conference on Industrial Technology (ICIT). [S.l.], 2019. p. 955–962. Citado na página 18.

PUTHAL, D.; SAHOO, B. P.; MISHRA, S.; SWAIN, S. Cloud computing features, issues, and challenges: a big picture. In: IEEE. 2015 International Conference on Computational Intelligence and Networks. [S.l.], 2015. p. 116–123. Citado na página 13.

QIN, S. J. Control performance monitoring—a review and assessment. Computers & Chemical Engineering, Elsevier, v. 23, n. 2, p. 173–186, 1998. Citado na página 34.

RIVERA, D. E.; MORARI, M.; SKOGESTAD, S. Internal model control: Pid controller design. Industrial & engineering chemistry process design and development, ACS Publications, v. 25, n. 1, p. 252–265, 1986. Citado na página 43.

ROCHA, M. S.; SESTITO, G. S.; DIAS, A. L.; TURCATO, A. C.; BRANDÃO, D.; FERRARI, P. On the performance of opc ua and mqtt for data exchange between industrial plants and cloud servers. ACTA IMEKO, v. 8, n. 2, p. 80–87, 2019. Citado 2 vezes nas páginas 17 e 18.

SHAHNI, F.; YU, W.; YOUNG, B. Rapid estimation of pid minimum variance. ISA transactions, Elsevier, v. 86, p. 227–237, 2019. Citado na página 34.

SHARDT, Y. A. Statistics for Chemical and Process Engineers. [S.l.]: Springer, 2015. Citado na página 22.

SKOGESTAD, S.; POSTLETHWAITE, I. Multivariable feedback control: analysis and design. [S.l.]: Citeseer, 2007. v. 2. Citado na página 34.

SRIVASTAVA, P.; KHAN, R. A review paper on cloud computing. International Journal of Advanced Research in Computer Science and Software Engineering, v. 8, n. 6, p. 17–20, 2018. Citado na página 19.

STARR, K. D.; PETERSEN, H.; BAUER, M. Control loop performance monitoring—abb’s experience over two decades. IFAC-PapersOnLine, Elsevier, v. 49, n. 7, p. 526–532, 2016. Citado 4 vezes nas páginas 13, 29, 34 e 62.

SWANDA, A. P.; SEBORG, D. E. Controller performance assessment based on setpoint response data. In: IEEE. Proceedings of the 1999 American Control Conference (Cat. No. 99CH36251). [S.l.], 1999. v. 6, p. 3863–3867. Citado 10 vezes nas páginas 31, 32, 33, 43, 54, 55, 56, 57, 58 e 64.

TOLDINAS, J.; LOZINSKIS, B.; BARANAUSKAS, E.; DOBROVOLSKIS, A. Mqtt quality of service versus energy consumption. In: IEEE. 2019 23rd International Conference Electronics. [S.l.], 2019. p. 1–4. Citado 2 vezes nas páginas 16 e 17.

VERONESI, M.; VISIOLI, A. Performance assessment and retuning of pid controllers. Industrial & engineering chemistry research, ACS Publications, v. 48, n. 5, p. 2616–2623, 2009. Citado 2 vezes nas páginas 31 e 32.

VISIOLI, A. Practical PID control. [S.l.]: Springer Science & Business Media, 2006. Citado 2 vezes nas páginas 26 e 28.

WANG, L.; LASZEWSKI, G. V.; YOUNGE, A.; HE, X.; KUNZE, M.; TAO, J.; FU, C. Cloud computing: a perspective study. New generation computing, Springer, v. 28, n. 2, p. 137–146, 2010. Citado na página 18.

YU, Z.; WANG, J.; HUANG, B.; BI, Z. Performance assessment of pid control loops subject to setpoint changes. Journal of Process Control, Elsevier, v. 21, n. 8, p. 1164–1171, 2011. Citado 2 vezes nas páginas 31 e 32.

YUAN, M. Getting to know MQTT. 2017. Acessado em: 14 de maio de 2021. Disponível em: <<https://developer.ibm.com/br/technologies/iot/articleds/iot-mqtt-why-good-for-iot>>. Citado na página 15.

ZHAN, C.; LI, S.; YANG, Y. Improved process monitoring based on global–local manifold analysis and statistical local approach for industrial process. Journal of Process Control, Elsevier, v. 75, p. 107–119, 2019. Citado na página 34.