COMPARAÇÃO DE CONTROLE PREDITIVO E ADAPTATIVO NA CLIMATIZAÇÃO PREDIAL COM ESTIMATIVA NEURO-VISUAL DA CARGA TÉRMICA

HEYDER ANTONIO SILVA DE ARAUJO

TRABALHO DE CONCLUSÃO DE CURSO EM ENGENHARIA ELÉTRICA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

HEYDER ANTONIO SILVA DE ARAUJO

ORIENTADOR: ADOLFO BAUCHSPIESS

TRABALHO DE CONCLUSÃO DE CURSO EM
ENGENHARIA ELÉTRICA

BRASÍLIA/DF: DEZEMBRO - 2020

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

COMPARAÇÃO DE CONTROLE PREDITIVO E ADAPTATIVO NA CLIMATIZAÇÃO PREDIAL COM ESTIMATIVA NEURO-VISUAL DA CARGA TÉRMICA

HEYDER ANTONIO SILVA DE ARAUJO

TRABALHO DE CONCLUSÃO DE CURSO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO ELÉTRICISTA.

APROVADA POR:

_____

Prof. Dr. Adolfo Bauchspiess – ENE/Universidade de Brasília
Orientador

_____

Prof. Dr. Eduardo Stockler Tognetti – ENE/Universidade de Brasília
Membro Interno

_____

Prof. Dr. Renato Coral Sampaio – GAMA/Universidade de Brasília
Membro Interno

BRASÍLIA, 08 DE DEZEMBRO DE 2020.

FICHA CATALOGRÁFICA

ARAUJO, HEYDER
COMPARAÇÃO DE CONTROLE PREDITIVO E ADAPTATIVO NA CLIMATIZAÇÃO PREDIAL COM ESTIMATIVA NEURO-VISUAL DA CARGA TÉRMICA [Distrito Federal] 2020.
xiv, 95p., 210 x 297 mm (ENE/FT/UnB, Engenheiro Elétricista, Engenharia Elétrica, 2020).
Trabalho de Conclusão de Curso – Universidade de Brasília, Faculdade de Tecnologia.
Departamento de Engenharia Elétrica

1. Racionalização de energia	2.Controle Adaptativo
3. Controle Preditivo	4. Identificação de Sistemas
I. ENE/FT/UnB	II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

ARAUJO, H. (2020). COMPARAÇÃO DE CONTROLE PREDITIVO E ADAPTATIVO NA CLIMATIZAÇÃO PREDIAL COM ESTIMATIVA NEURO-VISUAL DA CARGA TÉRMICA . Trabalho de Conclusão de Curso em Engenharia Elétrica, Publicação , Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 95p.

CESSÃO DE DIREITOS

AUTOR: Heyder Antonio Silva de Araujo
TÍTULO: COMPARAÇÃO DE CONTROLE PREDITIVO E ADAPTATIVO NA CLIMATIZAÇÃO PREDIAL COM ESTIMATIVA NEURO-VISUAL DA CARGA TÉRMICA .
GRAU: Engenheiro Elétricista ANO: 2020

Heyder Antonio Silva de Araujo

Departamento de Engenharia Elétrica (ENE) - FT
Universidade de Brasília (UnB)
Campus Darcy Ribeiro
CEP 70919-970 - Brasília - DF - Brasil

AGRADECIMENTOS

Heyder Antonio Silva de Araujo

Agradeço à minha família, aos meus pais Helda e Antonio, a minha irmã Tarciane, à minha tia Hilna, aos meus primos Williames, Wellingson e Wellington por todo o apoio por toda minha vida sem os quais nada disso seria possível. Agradeço à minha colega de trabalho Marcella por todos os bons momentos e pela amizade durante o trabalho. Agradeço aos meus companheiros de curso José Oniram, João Vitor, Antonio, Rodrigo Sabino, Rodrigo Fernandes, Rodrigo Castro, Tiago Lima, Luan Calaça, George Brindeiro, Thiago Barra, Filipe Ribeiro e Leonardo Cunha por todos os momentos de companheirismo durante esta jornada. Aos meus amigos Jackson Vasconcelos, Carolina Umetsu, Cassiana Umetsu, Vitor Tolentino, Athos Ribeiro e Marcus Matias por todos os anos de amizade. Agradeço ao professor Adolfo, pelo fantástico trabalho de orientação e pela tranquilidade durante o desenvolvimento deste trabalho.

RESUMO

Existe uma grande preocupação no que diz respeito ao consumo excessivo de energia elétrica. Dentro deste contexto, estudos têm sido feitos no intuito de tornar mais eficiente o uso deste tipo de energia. Geralmente, sistemas térmicos são os que representam uma grande parcela do consumo energético residencial e comercial.

É proposta, deste projeto, otimizar o uso da energia em sistemas térmicos. Este trabalho é parte de uma linha de pesquisa que visa tornar mais eficiente o consumo de energia elétrica em sistemas condicionadores de ar, considerando o conforto térmico da instalação. O presente estudo está focado no consumo de energia de um ambiente predial. A Instalação utilizada foi a sala de reuniões do Laboratório de Robótica e Automação (LARA), onde a variável controlada é a temperatura da sala. A perturbação considerada é formada pela contribuição das diferenças de temperatura entre o ambiente externo ao prédio e das salas ao redor do ambiente em estudo além da carga térmica proveniente da ocupação do ambiente por pessoas.

Essa carga térmica proveniente da ocupação depende da quantidade de pessoas presentes no ambiente. A determinação da ocupação do ambiente é feita através da utilização de uma rede neural convolucional capaz de detectar a presença de pessoas em uma imagem capturada em uma câmera.

Quanto ao controle, a comunicação entre os nós da rede de controle será feita pela internet através da plataforma ThingSpeak implementado em um Raspiberry PI 4B e em módulos NodeMCU's. A medição da temperatura é tomada por sensores do tipo DS18B20 para a temperatura da sala e DHT22 para os outros ambientes, e o processamento de dados com o auxílio do MATLAB.A identificação da função de transferência será feita utilizando a toolbox de identificação de sistemas do MATLAB.

A eficiência energética do controlador preditivo foi comparada ao controlador adaptativo e aos tradicionais Liga-Desliga e PI implementados em estudos passados.

Palavras-chave: Racionalização de energia,Controle Adaptativo,Controle Preditivo,Identificação de Sistemas.

ABSTRACT

There is been a great concern regarding the use of energy. Within this context, studies have been made in order to make the use of such energy more efficient. Thermal systems are usually the ones which represent the largest amount of energy expenditure in building environments.

The propose of this project is to optimize the use of energy in thermal systems. This project is part of a research line that aims a more efficient use of electrical energy in air conditioning systems, considering comfort. Such study is focused on the consumption of energy in building environments: the used environment was the meeting room from the Laboratory of Robotics and Automation (LARA) where the control variable is the room temperature. The considered perturbation is formed by the contribution of the temperature differential between the external environment to the building temperature, the surroundings rooms temperature and the thermal load from the people occupying the room.

This thermal load from the people occupying the room is a function of the number of people in the environment. The determination of the environment occupation is done through the use of a convolutional neural network capable of detecting the presence of people in an image captured on a camera.

As for the control, the communication between the control network nodes will be made through the internet through the ThingSpeak platform implemented in a Raspiberry PI 4B and in NodeMCU's modules. The temperature measurement is taken by sensors of the type DS18B20 for room temperature and DHT22 for other environments, and data processing with the aid of MATLAB. The transfer function will be identified using the MATLAB systems identification toolbox.

The energy efficiency of the predictive controller has been compared to the adaptive controller and the traditional On-Off and PI implemented in past studies.

Keywords: Energy Racionalization, Adaptive Control, Predictive Control, System Identification.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ACRÔNIMOS E ABREVIA CÕES

AC Alternate Current. 1

ADC Analogto Digital Converter. 1

API Application Programming Interface. 1

ARMAX Autoregressive Moving Average Model With Exogenous Inputs. 1

ASHRAE Sociedade Americana de Engenheiros de Aquecimento, Refrigeração e Ar Condicionado. 1–3

CARMA *Controller Auto-Regressive Moving-Average*. 1, 40

CHR Chien, Hrone e Reswick. 1, 33

CNN Rede Neural Convolucional. 1, 29

GPC Controle Preditivo Generalizado. 1

IDE Integrated Development Environment. 1

IoT Internet of Things. 1, 47, 50

LARA Laboratório de Automação e Robótica. xi, 1, 6, 52

ML Aprendizado de Máquina. 1, 29

MOE matriz de observabilidade extendida. 1

MPC controle preditivo baseado em modelo. xi, 1, 37–39, 54

MQ Mínimos Quadrados. 1, 10, 18, 22, 54, 61

PMV Predicted Mean Vote. 1, 3

PPD Predicted Percentage of Dissatisfied. 1, 3

PWM Pulse-Widht Modulation. 1

RAM Random Access Memory. 1

SISO Single Input Single Output. 1

SVD decomposição em valores singulares. 1, 12–14, 19–21

TPU Tensor Processing Units. 1, 30

USB Universal Serial Bus. 1

1 INTRODUÇÃO

1.1 MOTIVAÇÃO

A racionalização do consumo de energia elétrica tem sido uma preocupação. Grande parte do consumo vem de aparelhos que aquecem ambientes ou que os resfriam, como chuveiros elétricos e aparelhos de ar-condicionado. Todos estes equipamentos têm em comum o objetivo de propiciar o conforto térmico. Países tropicais demandam o uso de aparelhos de ar-condicionado praticamente ao longo de todo ano.

O LARA está conduzindo um projeto de automação predial no contexto de ambientes inteligentes. Uma das áreas de estudo é o conforto térmico, que envolve o controle de temperatura e umidade em uma sala de reuniões e seus ambientes adjacentes.

A automação predial é um ramo em que o uso da tecnologia e o incentivo a pesquisas têm aumentado com o passar dos anos. Várias são as áreas de atuação, que vão desde segurança preditiva (contra assaltos, incêndios), até automatização de portas, janelas e demais aparelhos eletrônicos, como o ar-condicionado. O uso de computadores é algo corriqueiro neste tipo de trabalho, e é bastante comum organizá-los em uma rede (cabeada ou não).

No que tange ao campo de ambientes prediais, redes sem fio permitem o uso de sensores, equipamentos interligados por redes wireless e unidades de processamento associadas. As escolhas dos módulos e do padrão de comunicação são de fundamental importância. Para este trabalho foi utilizado um servidor na nuvem para armazenamento de dados.

Controlar a temperatura de todos os pontos de um prédio é um trabalho excessivamente dispendioso e demorado. Muitas vezes, os controladores usados no campo da automação predial são simples, dimensionados empiricamente. Uma vez que o modelo não é bem conhecido e está exposto a perturbações, a identificação da dinâmica da planta em estudo é uma forma de obter uma caracterização que permite projetar controladores baseados em modelos. Devemos, então, nos preocupar com a escolha das variáveis, assim como com a quantidade suficiente de sinais para estabelecer uma lógica de controle satisfatória.

O MPC é um conjunto de métodos de controle que abarca o conceito de predição e obtenção do sinal de controle através da minimização de uma determinada função objetivo e considera o erro futuro e as restrições nas variáveis de processo. O MPC tem mostrado muito êxito na indústria de controle de processos. Isso porque essa técnica permite lidar com diferentes casos. Desde sistemas SISO (única entrada e única saída) até sistemas multivariaveis. Também permite a inclusão de ações de realimentação e pré-alimentação, podem ser incluídas restrições de entrada e saída na formulação da lei de controle e pode também compensar

intrinsecamente os tempos mortos do processo.

1.2 CONFORTO TÉRMICO

O conforto térmico é uma grandeza subjetiva, que tenta quantificar a sensação humana de frio ou calor. Existem vários fatores físicos, fisiológicos e psicológicos envolvidos: estrutura corporal, resposta metabólica, diferenças de percepção, resposta a estímulos sensoriais.

Os estudos em conforto térmico visam, principalmente, analisar e estabelecer as condições necessárias para a avaliação e concepção de um ambiente térmico adequado às atividades e à ocupação humana, bem como estabelecer métodos e princípios para uma detalha da análise térmica de um ambiente[1]. Segundo a Sociedade Americana de Engenheiros de Aquecimento, Refrigeração e Ar Condicionado (ASHRAE), o conforto térmico é "a condição da mente que expressa satisfação com o ambiente térmico".

1.2.1 Termorregulação humana

O ser humano é um animal homeotérmico, isto é, precisa manter sua temperatura constante, e por isso possui um mecanismo termorregulador que controla as suas variações térmicas. Nesse sentido, o corpo humano é aproximado a uma máquina térmica, e como tal, precisa de um mínimo de calor para operar, além de também produzi-lo. O calor necessário para desempenharmos nossas atividades se dá pela metabolização dos alimentos ingeridos. Em relação ao calor produzido, uma parte deste é fundamental para o funcionamento fisiológico, enquanto a outra parcela deve ser dissipada para que não ocorra um superaquecimento do corpo humano.

A figura 1.1 exemplifica quais os tipos de trocas térmicas que podem haver entre o corpo humano e o ambiente.

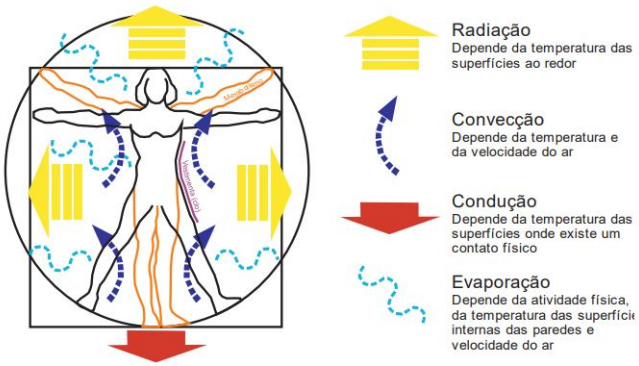

Figura 1.1 – Fisiologia humana e suas trocas térmicas.[1]

1.2.2 Índice PMV

Os estudos oriundos dessa área geraram vários modelos matemáticos e, portanto diversos índices que quantificassem a "agradabilidade" de um ambiente. Desta forma, criou-se um padrão: o índice Predicted Mean Vote (PMV). As normas ISO7730 (1994) e ASHRAE Standard 55 (2005) usam este índice e são as mais aceitas no meio acadêmico.

São considerados quatros parâmetros físicos: umidade, temperatura, temperatura radiante média e velocidade do ar. Além dessas condições ambientais, dois parâmetros individuais são levados em consideração, que são o nível de atividade da pessoa (sentada, andando, conversando) e as vestimentas.

O índice varia, em escala, de 3 a -3, em que o valor 3 equivale a um ambiente muito quente, e o valor -3, a um ambiente muito frio. O valor 0 corresponde ao equilíbrio ideal em se tratando de conforto térmico.

Associa-se ao PMV um índice de insatisfação com o ambiente, o PPD(Predicted Percentage of Dissatisfied)[3]. A figura 1.2 explicita essa relação.

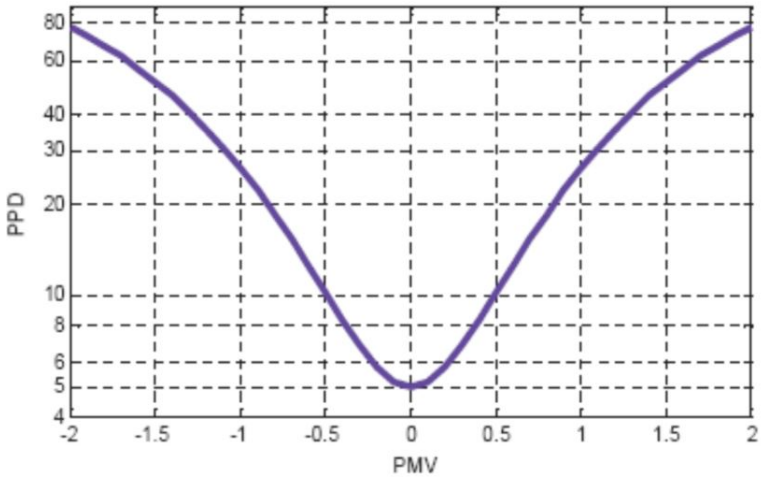

Figura 1.2 – Gráfico de relação do PMV com o PPD.

O trabalho foi guiado por este índice para avaliar quais são as variáveis importantes, e como devemos tratar a presença de pessoas em um ambiente predial.

Em [3] é mostrado que a temperatura é a variável essencial para o conforto térmico. Já o ser humano possui trocas de calor complexas, como podemos ver pela figura 1.1, e a sua presença é tratada como uma perturbação para a temperatura ambiente.

1.3 OBJETIVOS

Os objetivos deste trabalho podem ser divididos em duas categorias:objetivos gerais e específicos.

1.3.1 Objetivos Gerais

A proposta deste trabalho é a comparação do desempenho dos controladores adaptativos e preditivos em relação às estratégias classicas de controle.Os tempos de comparação serão a velocidade de resposta do sistema(em quanto tempo o sistema alcança a temperatura de referência, o quão próximo da referência o controlador consegue manter a temperatura em torno da referência e a eficiência energética dos controladores.

Para que os controladores tenham uma melhor resposta é preciso estimar as pertubações presentes no ambiente, portanto, é necessário o desenvolvimento de um sistema capaz de estimar a carga térmica devido ao número de ocupantes do ambiente e a carga térmica devido ao fluxo de calor proveniente dos ambientes ao redor da sala de reuniões.

1.3.2 Objetivos Específicos

Para alcançar os objetivos gerais propostos neste trabalho será necessário:

- Instrumentação do processo;
- Instalação dos sensores de temperatura e atuadores no ambiente;
- Desenvolvimento de um sistema capaz de contar o número de pessoas presentes.
- Identificar os parâmetros e o modelo matemático do ambiente;
- Simular os controladores com os modelos encontrados;
- implementar a malha de controle;
- comparação entre os resultados obtidos e os resultados utilizando controladores mais tradicionais como o PI e o liga-desliga.

1.4 APRESENTAÇÃO DO MANUSCRITO

Este trabalho está organizado da seguinte forma: no capítulo 2 faremos a revisão bibliográfica sobre modelos e algoritmos de identificação, controle preditivo, controle adaptativo,

controle PID, controle Liga-Desliga, redes neurais convolucionais e detecção de objetos. No capítulo 3, serão detalhados os materiais usados na instrumentação do projeto, tais como sensor, os módulos de comunicação e microcontrolador. No capítulo 4, será descrito como procedemos com a comunicação sem fio e o fluxo de dados para a implementação do controlador preditivo. A seguir, no capítulo 5, serão apresentados os resultados da identificação, o desempenho de cada tipo de controle empregado também será apresentado neste capítulo, sendo feitas as devidas comparações de desempenho em termos de economia energética. No capítulo 6 é feita a conclusão5 a partir dos resultados obtidos, salientando-se as diferenças entre os ambientes de estudo e outras peculiaridades dos experimentos e uma lista de sugestões para trabalhos futuros.

2 REVISÃO BIBILIOGRÁFICA

2.1 AMBIENTE DE ESTUDO

O laboratório LARA pode ser separado em seis ambientes, como pode ser visto na figura 2.1 . Um deles é a sala de reuniões, que também funciona como ambiente de estudo (figura 2.2). Nela, está instalado o sistema de ar condicionado que usaremos. A figura 2.3 mostra a localização do aparelho de ar condicionado na sala de reuniões e na figura 2.4 podemos ver o interior do quadro de acionamento.

Figura 2.1 – Planta baixa do LARA.

Figura 2.2 – Vista da sala de reuniões do LARA

Figura 2.3 – Ar condicionado da sala de reunião (painel de controle, unidade evaporativa).

Figura 2.4 – Interior do quadro de acionamento.

Para facilitar a captação de dados, o ambiente ao lado da sala de reunião foi dividido em duas áreas (áreas1 e 2). É importante ressaltar que, na sala de reuniões, há um bom isolamento térmico do piso, do teto e da parede que a separa da área 3. O ambiente de estudo é separado das áreas 1 e 2 por divisória simples e vidro.

É muito raro um grupo de pessoas estar dividindo um ambiente e estas não interagirem entre si. Muitas pessoas visitam a sala para fazer reunião – o objetivo principal – ou para estudar. Em ambos os casos, fica óbvio que haverá muita movimentação. Isso, ao ser incluído no modelo de sinal, é tratado como uma perturbação. Como exemplo, temos a figura 2.5, em que a linha em vermelho retrata o efeito da perturbação devido à presença de pessoas interferindo nos ciclos do controlador.

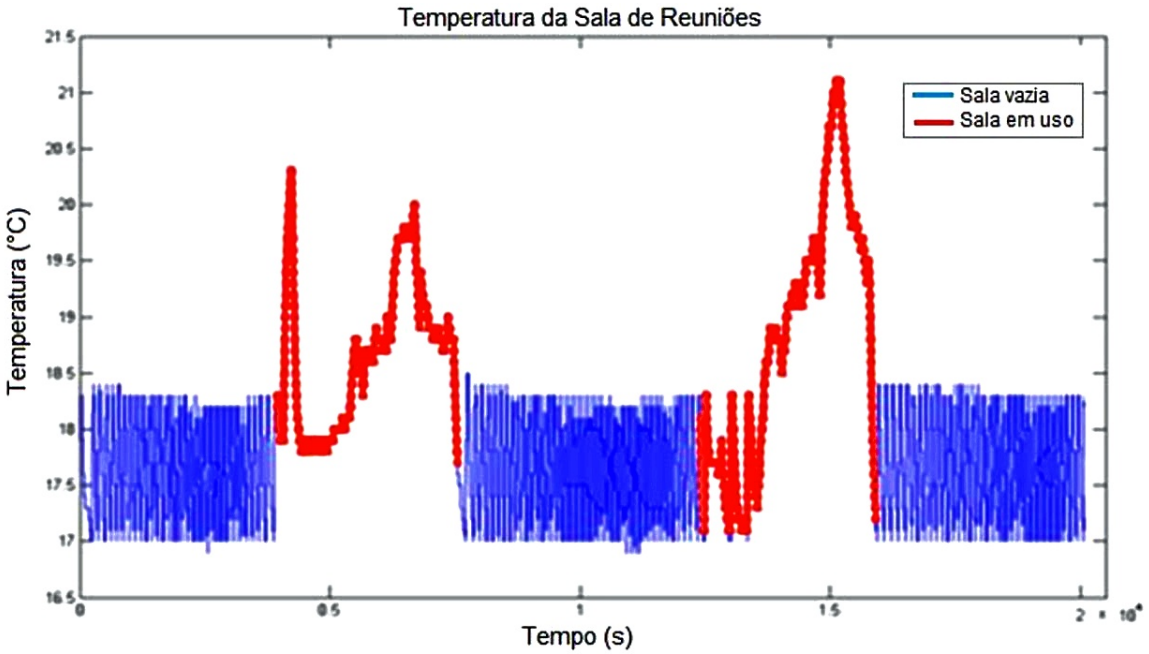

Figura 2.5 – Em vermelho, perturbações devido ao trânsito de pessoas

2.2 MODELAGEM MATEMÁTICA

Na atualidade, cada vez mais, o trabalho de um engenheiro consiste na obtenção de modelos matemáticos dos processos estudados[4]. Projetos convencionais de sistemas de controle para sistemas dinâmicos, como por exemplo, processos químicos catalíticos, satélites espaciais, braços robóticos e veículos aéreos dependem da disponibilidade de modelos que representem e descrevam alguns dos fenômenos dinâmicos envolvidos[5].

Na modelagem matemática de sistemas, existem basicamente duas formas de obtenção de modelos: a identificação de sistemas e a modelagem pela física do processo[6].A identificação de sistemas é uma forma empírica de obtenção de modelos, baseada em experimentos e coleta de dados, enquanto a modelagem pela física do processo é um procedimento baseado no conhecimento sobre os fenômenos envolvidos. Sabe-se que o conhecimento das relações físicas do processo pode exigir um esforço significativo e específico para cada sistema, o que muitas vezes é difícil. Por outro lado, a identificação de sistemas, exige pouco ou praticamente nenhum conhecimento específico do processo. Por essas razões, têm sido de grande interesse estudar essa área.

2.2.1 Técnicas de Modelagem

Há várias formas de classificar técnicas de modelagem. Um delas agrupa os métodos em três categorias denominadas modelagem caixa branca, caixa preta e caixa cinza.

Caixa Branca:é necessário conhecer o sistema em detalhes, bem como as leis físicas que

descrevem o sistema a ser modelado. Devido ao conhecimento e ao tempo necessários para modelar um sistema partindo do equacionamento dos fenômenos envolvidos, nem sempre é viável seguir esse procedimento;

Caixa preta:geralmente, possui-se pouco ou nenhum conhecimento prévio do sistema, por isso, esse tipo de modelagem também é denominado modelagem empírica. Aqui, são levantas as relações de causa e efeito entre entradas e saídas, pois não se conhecem as equações envolvidas no funcionamento de determinado sistema, ou até são conhecidas, mas seria impraticável, por limitações de tempo e demais recursos, levantar tais equações e estimar seus respectivos parâmetros.Dentre os métodos de modelagem caixa-preta para sistemas lineares, pode-se citar dois métodos de identificação: os métodos de predição de erro e os métodos de identificação por subespaços[5];

Caixa cinza:uma alternativa entre a modelagem puramente física e a puramente empírica, usa informação auxiliar, que não se encontra no conjunto de dados utilizados durante a identificação. O tipo de informação auxiliar e a forma com que ela é usada varia muito entre diversas técnicas disponíveis, cujo desenvolvimento é um dos grandes desafios na identificação de sistemas[6].

Para este trabalho, a modelagem caixa cinza foi escolhida, a priori.

2.2.2 Algoritmos

Deve-se definir qual o método (algoritmo) a ser utilizado, que variam entre determinístico e estocástico.

Métodos determinísticos:não tratam o ruído, ainda que se aceite o fato de que os dados estejam contaminados. Esse método só apresenta bons resultados quando a relação sinal/-ruído é suficientemente alta.

Métodos estocásticos:utilizam recursos adequados para levar em consideração o ruído e, dessa forma, reduzir seus efeitos sobre o modelo identificado.

Definir como será a captação de dados também influencia o algoritmo a ser usado.

Batelada:quando toda a massa de dados está disponível antes de começar a estimar os parâmetros – o problema numérico será resolvido de uma vez só.

Estimação recursiva:outra situação recorrente é de os dados serem medidos e disponibilizados sequencialmente. É claro que eles poderiam ser armazenados até ter uma quantidade suficiente, mas é possível utilizá-los enquanto a coleta é feita. É útil, principalmente, quando os parâmetros do processo variam lentamente ou quando a solução em batelada é dispendiosa, como a execução do algoritmo em tempo real, por exemplo.[6]

2.2.3 Identificação Utilizando o Métodos de Subespaços

Considere o modelodo sistema na forma de espaço de estados[4]:

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) + w(t) \\ y(t) &= Cx(t) + Du(t) + \nu(t) \end{aligned} \tag{2.1}$$

Assumimos que a saída $y(t)$ é um vetor coluna de dimensão p, equanto a saída $u(t)$ é um vetor coluna de dimensão m. A ordem do sistema, ou seja, a dimensão de $x(t)$ é n. Assumindo que a representação em espaço de estados possui realização mínima, a relação de entrada e saída pode ser descrita como

$$\begin{aligned} \tilde{x}(t+1) &= T^{-1}AT\tilde{x}(t) + T^{-1}Bu(t) + \tilde{w}(t) \\ y(t) &= CT\tilde{x}(t) + Du(t) + \nu(t) \end{aligned} \tag{2.2}$$

para qualquer matriz T inversilvel. Isso corresponde a uma mudança de base $\tilde{x}(t) = T^{-1}x(t)$ no estado de espaço.

Considerando as seguintes observações:

- Se $\hat{A}$ e $\hat{C}$ são conhecidas é um simples problema de Mínimos Quadrados (MQ) estimar B e D apartir de:

$$y(t) = \hat{C}\left(qI - \hat{A}\right)^{-1} Bu(t) + Du(t) + \nu(t) \tag{2.3}$$

  usando o preditor

$$\hat{y}(t|B,D) = \hat{C}\left(qI - \hat{A}\right)^{-1} Bu(t) + Du(t) \tag{2.4}$$

- Se a matriz de observabilidade(extendida) do sistema

$$O_r = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{r-1} \end{bmatrix} \tag{2.5}$$

  é conhecida, então, é simples determinar C e A. Utilize o primeiro bloco de linha de O_r é a propriedade de translado, respectivamente.

- A matriz de observabilidade extendida pode ser estimada consistentemente apartir dos dados de entrada e saída por um algoritmo parecido com minimos quadrados(projeção)

2.2.3.1 Estimando B e D

Para $\hat{A}$ e $\hat{C}$ fixas e dadas, a estrutura do modelo (2.4) é linear em B e D. O preditor é formado inteiramente por entradas passadas, então, é uma estrutura de *modelo de erro de saída*. Se o sistema opera em malha aberta, podemos consistentemente estimar B e D mesmo se o ruido em 2.3 for colorido[1].

Escrevendo o preditor em (2.4) na forma padrão da regressão linear:

$$\hat{y}(t) = \varphi(t)\theta = \varphi(t) \begin{bmatrix} Vec(B) \\ Vec(D) \end{bmatrix} \tag{2.6}$$

Com uma matriz $\varphi(t)$ com dimensão $p \times (mn + mp)$[4]. Onde "Vec" é um operador que constroi um vetor apartir de uma matriz colocando suas colunas em cima umas das outras.

Faça $r = (k-1)n + j$. Para encontrar a $r - est$ coluna de $\varphi(t)$ tal que $r \leq mn$, que corresponda ao $r - est$ elemento de θ, isto é, o elemento $B_{j,k}$, diferenciamos 2.6 em relação a este elemento e obtemos[4]:

$$\varphi_r(t) = \hat{C}\left(qI - A\right)^{-1} E_j u_k(t)$$

Onde E_j é o vetor coluna com o $j - est$ elemento igual a 1 e os outros iguais a 0. As colunas para $r > nm$ são lidadas de maneira similar.

Se desejado, o estado inicial $x_0 = x(0)$ pode ser estimado de maneira analoga, já que o preditor levando os valores iniciais em conta é:

$$\hat{y}(t|B, D, x_0) = \hat{C}\left(qI - \hat{A}\right)^{-1} x_0\delta(t) + \hat{C}\left(qI - \hat{A}\right)^{-1} Bu(t) + Du(t) \tag{2.7}$$

O preditor em (2.7) também é linear para x_0.Aqui $\delta(t)$ é o pulso unitário no instante $t = 0$.

Uma Maneira de melhorar as estimações é determinar a média do ruido e filtrar os dados de acordo com essa média.

2.2.3.2 Encontrando A e C apartir da Matriz de Observabilidade Extendida

Suponha que uma matriz G de dimensão $pr \times n^*$ é dada e relacionada com a matriz de observabilidade extendida do sistema. Temos que determinar A e C apartir de G considerando

[1] Ruido com media não nula

o incremento do complexidade.

1. **Sabendo a ordem do sistema**:
   Suponha que:

$$G = O_r$$

   Para que $n^* = n$. Para encontrar C temos que imediatamente:

$$\hat{C} = O_r(1 : p, 1 : n) \tag{2.8}$$

   A equação 2.8 utiliza a notação do Matlab, logo, $M(s : l, j : k)$ é a matriz obtida extraindo as linhas $s, s + 1, ..., l$ e as colunas $j, j + 1, ..., k$ da matriz M.

   Similarmente, podemos encontrar $\hat{A}$ da equação 2.9:

$$O_r(p + 1 : pr, 1 : n) = O_r(1 : p(r - 1), 1 : n)\hat{A} \tag{2.9}$$

   Que é facilmente visto apartir de (2.5). Assumindo a condição de observabilidade, O_{r-1} tem posto n, então $\hat{A}$ pode ser unicamente determinada.

2. **Ordem do sistema desconhecida**:
   Suponha que a ordem do sistema é desconhecida e que n^*, número de colunas de G, é somente o limite superior da ordem. Logo:

$$G = O_r\tilde{T} \tag{2.10}$$

   Para uma matriz $\tilde{T}$ desconhecida mas de posto completo $n \times n^*$, onde n tambem é desconhecido. Uma maneira direta de lidar com isso seria determinar o posto, apagando as últimas $n^* - n$ colunas de G. Uma maneira mais geral e numéricamente concreta de reduzir o espaço coluna é usar a decomposição em valores singulares (SVD):

$$G = USV^T = U \begin{bmatrix} \sigma_1 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & 0 & \dots & 0 \\ 0 & 0 & \sigma_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \sigma_{n^*} \\ 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix} V^T \tag{2.11}$$

Onde U e V são matrizes ortonormais[2] de dimensões $pr \times pr$ e $n^* \times n^*$, respectivamente. S é uma matriz $pr \times n^*$ com valores singulares de G na diagonal e zeros em qualquer outra posição. Se G possui posto n, somente os primeiros n valores singulares σ_k serão diferentes de zero. Isso significa que podemos reescrever:

$$G = USV^T = U_1 S_1 V_1^T \tag{2.12}$$

Onde U_1 é uma matriz $pr \times n$ contendo as primeiras n colunas de U, S_1 é a parte superior esquerda $n \times n$ de S e V_1 consiste das primeiras n colunas de V($V_1^T V_1 = I$). De (2.10) podemos ver que $O_r \tilde{T} = U_1 S_1 V_1^T$. Multiplicando isso por V_1 pela direita, temos:

$$O_r \tilde{T} V_1 = O_r T = U_1 S_1 \tag{2.13}$$

para alguma matriz inversível $T = \tilde{T} V_1$. Agora temos uma situação semelhante a $\tilde{O}_r = O_r T$ que conhecemos a matriz de observabilidade até uma matriz inversível T ou equivalente.Conhecemos a matriz de observabilidade em alguma base no espaço de estados. Consequentemente podemos usar que $\hat{O}_r = U_1 S_1$ ou $\hat{O}_r = U_1$ ou qualquer que possa ser escrita como:

$$\hat{O}_r = U_1 R \tag{2.14}$$

para alguma matriz inversível R em (2.8) e (2.9) para determinar a matriz $\hat{C}$ $p \times n$ e a matriz $\hat{A}$ $n \times n$.

3. **Utiliando a Estimação do Ruido da Matriz de Observabilidade Extendida**
   Agora assumindo que dada a matriz G $pr \times n^*$ é uma matriz com ruido estimado da matriz de observabilidade verdadeira

$$G = O_r \tilde{T} + E_N \tag{2.15}$$

onde E_N é pequena e tende a zero quando $E \to \infty$.O posto de O_r é desconhecido, enquanto a matriz do ruido E_N é provável que seja de posto completo. É razoavel proceder como acima e fazer uma SVD em G:

$$G = USV^T \tag{2.16}$$

Devido ao ruido, S terá, tipicamente, todos os valores singulares $\sigma_k, k = 1, ..., \min(n^*, pr)$ diferentes de zero.Os primeiro n serão suportados por O_r, equanto os restantes serão sustentados de E_N. Se o ruido é pequeno, podemos esperar que os primeiros valores serão significadamente maiores do que os últimos. Por isso, determine $\hat{n}$ como o número de valores singulares que são significadamente maiores que zero. Em seguida

[2] $U^T U = I, V^T V = I$

matenha esses e substitua os outros em S por zeros e proceda como em (2.12) para determinar U_1 e S_1. Então use $\hat{O}_r$ em (2.14) para determinar $\hat{A}$ e $\hat{C}$ como antes. Entretanto, devido ao ruido, $\hat{O}_r$ não estará exatamente na forma da estrutura em (2.9), então esse sistema de equações deverá ser resolvido no sentido de Mínimos Quadrados.

4. **Usando Matrizes de Ponderação no SVD**
   Para mais flexibilidade nos poderiamos pré e pos multiplicar G assim ficamos com $\hat{G} = W_1 G W_2$ antes de fazermos a SVD

$$\hat{G} = W_1 G W_2 = U S V_T \approx U_1 S_1 V_1^T \tag{2.17}$$

   e então em vez de (2.14) use

$$\hat{O}_r = W_1^{-1} U_1 R \tag{2.18}$$

   para determinar $\hat{C}$ e $\hat{A}$ em (2.8) e (2.9). Onde R é uma matriz arbitrária, que irá determinar as bases para a representação do estado. A pós multiplicação por $W2$ corresponde a uma mudança de base no espaço de estados e a pré multiplicação por W_1 é eliminada em (2.18), então no caso sem ruido $E = 0$, essas ponderações são sem consequências. Entretanto quando o ruido é presente , eles possuem uma importante influência no espaço medido por U_1 e aí em diante na qualidade das estimções de $\hat{C}$ e $\hat{A}$. Podemos observar que a pós-multiplicação W_2 por uma matriz ortonormal não afeta a matriz U_1 na decomposição.

2.2.3.3 Estimando A Matriz de Observabilidade Extendida

1. **A expressão básica**
   De (2.1), temos que

$$\begin{aligned} y(t+k) &= Cx(t+k) + Du(t+k) + \nu(t+k) \\ &= CAx(t+k-1) + CBu(t+k-1) + Cw(t+k-1) \\ &+ Du(t+k) + \nu(t+K) \\ &= ... \end{aligned}$$

$$\begin{aligned} y(t+k) = CA^k x(t) + CA^{k-1} Bu(t) + CA^{k-2} Bu(t+1) + ... \\ + CBu(t+k-1) + Du(t+k) \\ + CA^{k-1} w(t) + CA^{k-2} w(t+1) + ... + Cw(t+k-1) + \nu(t+k) \end{aligned} \tag{2.19}$$

De (2.19) podemos extrair os vetores:

$$Y_r(t) = \begin{bmatrix} y(t) \\ y(t+1) \\ \vdots \\ y(t+r-1) \end{bmatrix}, U_r(t) = \begin{bmatrix} u(t) \\ u(t+1) \\ \vdots \\ u(t+r-1) \end{bmatrix} \tag{2.20}$$

e rescrevendo (2.19) como:

$$Y_r(t) = O_r x(t) + S_r U_r(t) + V(t) \tag{2.21}$$

com:

$$S_r = \begin{bmatrix} D & 0 & \dots & 0 & 0 \\ CB & D & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ CA^{k-2}B & CA^{k-3} & \dots & CB & D \end{bmatrix}$$

e o $k - est$ bloco componente de $V(t)$:

$$\begin{aligned} V_k(t) = {} & CA^{k-2}w(t) + CA^{k-3}w(t+1) + \dots \\ & + Cw(t+k-2) + \nu(t+k-1) \end{aligned} \tag{2.22}$$

Devemos usar (2.21) para estimar O_r, ou melhor, a matriz $O_R T$ para alguma matriz desconhecida T.A idéia é correlacionar os dois lados de (2.21) com quantidades que eliminem o termo U_R e que faça a influência do ruido de V desaparecer assintoticamente, para isso temos as medidas $y(t), u(t), t = 1, \dots, N + r - 1$ disponíveis.Para facilitar a visualização é melhor descrever as operações de correlação como multiplicações de matrizes. Para isso introduzimos:

$$\begin{aligned} Y &= \begin{bmatrix} Y_r(1) & Y_r(2) & \dots & Y_r(N) \end{bmatrix} \\ X &= \begin{bmatrix} x(1) & x(2) & \dots & x(N) \end{bmatrix} \\ U &= \begin{bmatrix} U_r(1) & U_r(2) & \dots & U_r(N) \end{bmatrix} \\ V &= \begin{bmatrix} V(1) & V(2) & \dots & V(N) \end{bmatrix} \end{aligned} \tag{2.23}$$

Essas quantidades depedem de r e N, mas para facilitar a leitura vamos suprimir essas dependência. Podemos escrever (2.21) como:

$$Y = O_r X + S_r U + V \tag{2.24}$$

Observação: Defina o vetor $\hat{Y}$ preditor k-passo a frente como

$$\hat{Y} = O_r \hat{X} \tag{2.25}$$

onde $\hat{X}$ é feito apartie dos estados preditos(Filtro de Kalman) $\hat{x}(t|t+1)$, isto é, a melhor estimação de $x(t)$ baseada nas entradas e saídas passada

2. **Removendo o termo** U
   Forme a matriz $N \times N$:

$$\Pi_{U^T}^{\perp} = I - U^T \left(UU^T\right)^{-1} U \tag{2.26}$$

Se UU^T é singular, use a pseudoinversa no lugar. Essa matriz faz a projeção ortogonal em relação a matriz U, isto é:

$$U\Pi_{U^T}^{\perp} = U - UU^T \left(UU^T\right)^{-1} U = 0$$

Logo, multiplicando (2.21) pela direita por $\Pi_{U^T}^{\perp}$ irá eliminar o termo com U.

$$Y\Pi_{U^T}^{\perp} = O_r X \Pi_{U^T}^{\perp} + V\Pi_{U^T}^{\perp} \tag{2.27}$$

3. **Removendo o termo ruidoso**
   Defina a matriz de dimensão $s \times N (s \geq n)$:

$$\Phi = \begin{bmatrix} \varphi_S(1) & \varphi_S(2) & \dots & \varphi_S(N) \end{bmatrix} \tag{2.28}$$

onde $\varphi_S(t)$ é um vetor indefinido. Multiplique (2.27) pela direita por Φ^T e normalize por N:

$$G = \frac{1}{N} Y \Pi_{U^T}^{\perp} \Phi^T = O_r \frac{1}{N} X \Pi_{U^T}^{\perp} \Phi^T + \frac{1}{N} V \Pi_{U^T}^{\perp} \Phi^T \stackrel{def}{=} O_r \tilde{T_N} + V_N \tag{2.29}$$

Onde $\tilde{T}$ é uma matriz $n \times s$.Suponha que podemos encontrar $\varphi_s(t)$ tal que:

$$\lim_{N\to\infty} V_N = \lim_{N\to\infty} \frac{1}{N} V \Pi_{U^T}^{\perp} \Phi^T = 0 \tag{2.30a}$$

$$\lim_{N\to\infty} \tilde{T}_N = \lim_{N\to\infty} \frac{1}{N} \Pi_{U^T}^{\perp} \Phi^T = \tilde{T}, \qquad \text{tem posto completo n} \tag{2.30b}$$

Então (2.29) pode ser lida como:

$$\begin{aligned} G &= \frac{1}{N} Y \Pi_{U^T}^{\perp} \Phi^T = O_r \tilde{T} + E_N \\ E_N &= O_r(\tilde{T}_N - \tilde{T}) + V_N \to 0, \text{quando}, N \to \infty \end{aligned} \tag{2.31}$$

A matriz G $pr \times s$ pode ser vista como um estimador de ruido (2.15) e podem ser sujeitas a tratamento (2.16)-(2.18) para estimar A e C.

Agora temos de encontrar uma maneira de obter as condições em (2.30). Usando a expressão (2.26) para $\Pi_{U^T}^{\perp}$ e escrevendo as multiplicações por matrizes como somas:

$$\begin{aligned} \frac{1}{N} V \Pi_{U^T}^{\perp} \Phi^T = \frac{1}{N} \sum_{t=1}^{N} V(t) \varphi_S^T(t) - \frac{1}{N} \sum_{t=1}^{N} V(t) U_r^T(t) \\ \times \left[\frac{1}{N} \sum_{t=1}^{N} U_r(t) U_r^T(t) \right]^{-1} \frac{1}{N} \sum_{t=1}^{N} U_r(t) \varphi_S^T(t) \end{aligned} \tag{2.32}$$

Sobre condições amenas, a lei dos grandes números de estados que a soma de amostras converge para seus respectivos valores

$$\lim_{N \to \infty} \frac{1}{N} V \Pi_{U^T}^{\perp} \Phi^T = \bar{E} V(t) \Phi_S^T(t) - \bar{E} V(t) U_r^T(t) R_u^{-1} \bar{E} U_r(t) \Phi_S^T(t) \tag{2.33}$$

onde

$$R_u = \bar{E} U_r(t) U_r^T(t)$$

é a matriz de covariancia $r \times r$ das entradas. Considerando que as u foram produzidas em malha aberta, logo independente dos termos de ruido em V. Então $EV(t)U_r^T(t) = 0$. Assumindo também que R é inversível.Então o segundo termo de (2.33) será zero(se a pseudo-inversa for usada em (2.26), isso ainda será verdade mesmo se R_u for não inversível). Para que o primeiro termo seja zero, será preciso que $V(t)$ e $\varphi_S(t)$ não sejam correlatos.Já que $V(t)$, de acordo com (2.22), é formada de termos de ruido branco[3] de termos futuros, qualquer escolha de $\varphi_S(t)$ construida de dados do passado

[3]Ruido com média 0

irá satizfazer (2.30a).Uma escolha típica seria:

$$\varphi_s(t) = \begin{bmatrix} y(t-1) \\ \vdots \\ y(t-S_1) \\ u(t-1) \\ \vdots \\ u(t-S_2) \end{bmatrix} \tag{2.34}$$

Agora para (2.30b), podemos encontrar argumento semelhante que:

$$\tilde{T} = \bar{E}x(t)\varphi_S^T(t) - \bar{E}x(t)U_r^T(t)R_u^{-1}\bar{E}U_r(t)\varphi_S^T(t) \tag{2.35}$$

Logo, resumindo, formando $G = \frac{1}{N}V\Pi_{U^T}^{\perp}\Phi^T$ com Φ definida por (2.34) e (2.28) dá a propriedade (2.31) o que nos permite determinar A e C via (2.17) e (2.18) e (2.8) e (2.9).

2.2.3.4 **Encontrando os Estados e Estimando as Estatisticas do Ruido**

Em (2.24) construimos uma relação entre as saídas futuras e os estados.Agora vamos estimar a predição k-passos a frente:

$$Y_r(t) = \Theta\varphi_S(t) + \Gamma U_r(t) + E(t) \tag{2.36}$$

Negligenciando todos os t para facilitar a leitura, temos:

$$Y = \Theta\Phi + \Gamma U + E \tag{2.37}$$

A estimação por MQ dos parâmetros é:

$$\begin{bmatrix} \hat{\Theta} & \hat{\Gamma} \end{bmatrix} = \begin{bmatrix} Y\Phi^T & YU^T \end{bmatrix} \begin{bmatrix} \Phi\Phi^T & \Phi U^T \\ U\Phi^T & UU^T \end{bmatrix}^{-1} \tag{2.38}$$

Usando o lema para inversão de bolocos de matrizes[4] temos:

$$\hat{\Theta} = Y\Pi_{U^T}^{\perp}\Phi^T\left(\Phi\Pi_{U^T}^{\perp}\Phi^T\right)^{-1} \tag{2.39}$$

Com isso podemos escrever a matriz das saídas preditas como:

$$\begin{aligned}\hat{Y} &= \begin{bmatrix}\hat{Y}_r(1) & \dots & \hat{Y}_r(N)\end{bmatrix} \\ &= Y\Pi_{U^T}^{\perp}\Phi^T\left(\Phi\Pi_{U^T}^{\perp}\Phi^T\right)^{-1}\Phi\end{aligned} \tag{2.40}$$

O que reconhecemos como $\hat{G}$ em (2.17), com G como em (2.29) e as ponderações $W_1 = I$ e $W_2 = \left(\Phi\Pi_{U^T}^{\perp}\Phi^T\right)^{-1}\Phi$. Fazendo a SVD e deletando os valores singulares pequenos, obtemos:

$$\hat{Y} \approx U_1 S_1 V_1^T \tag{2.41}$$

Onde V_1^T é uma matriz $n \times N$. Sabemos de (2.18) que U_1 é relacionada à matriz de observabilidade por alguma matriz R invercível tal que $U_1 = O_r R^{-1}$. Introduzindo $\hat{X} = R^{-1}S_1V_1$, então (2.41) pode ser rescrita como:

$$\hat{Y} = O_r^{-1}S_1V_1^T = O_rX \tag{2.42}$$

Comparando com (2.25), vemos que $\hat{X}$ deve ser a matriz de estados estimada corretamente se $\hat{Y}$ é a matriz das saídas preditas verdadeira.As saídas preditas verdadeiras são, entretanto, obtidas somente quando a ordem de S_i em (2.34) tendem ao infinito. Nesse caso teremos encontrado os estados estimados verdadeiros, na base do espaço de estados em questão, a partir da SVD. Podemos escrever também:

$$\begin{aligned}\hat{X} = L\hat{Y} &= \begin{bmatrix}\hat{x}(1) & \dots & \hat{x}(N)\end{bmatrix} \\ L &= R^{-1}U_1^T\end{aligned} \tag{2.43}$$

[4]O Lema da inverção de matrizes em blocos diz:

$$\begin{bmatrix}A & D\\ C & B\end{bmatrix}^{-1} = \begin{bmatrix}\Delta^{-1} & -\Delta DB^{-1}\\ -B^{-1}C\Delta^{-1} & B^{-1}+B^{-1}C\Delta^{-1}DB^{-1}\end{bmatrix}$$

onde

$$\Delta = A - DB^{-1}C$$

Como $U_1^T U_1 = I$ das propriedades da SVD. Com os estados dados, podemos estimar o processo e o ruido medido como

$$\begin{aligned} w(t) &= \hat{x}(t+1) - \hat{A}\hat{x}(t) - \hat{B}u(t) \\ \nu(t) &= y(t) - \hat{C}\hat{x}(t) - \hat{D}u(t) \end{aligned} \tag{2.44}$$

e estimar suas matrizes de covariância de maneira direta. Aqui $\hat{A}, \hat{B}, \hat{C}, \hat{D}$ são matrizes do sistema estimadas obtida como descrito acima.

2.2.3.5 **Resumo e as Famílias de Algoritmos do Método de Subespaços**

O algoritmo completo pode ser sumarizado como:

1. Dos dados de entradas e saídas forme:

$$G = \frac{1}{N} Y \Pi_{U^T}^{\perp} \Phi^T \tag{2.45}$$

2. Selecione as matrizes de ponderação $W_1 (rp \times rp$ e inversível) e W_2 $((ps_1 + ms_2) \times \alpha)$ e faça a SVD

$$\hat{G} = W_1 G W_2 = U S V^T \approx U_1 S_1 V_1^T \tag{2.46}$$

3. Selecione uma matriz R de posto completo e apartir dessa defina a matriz $\hat{O}_r = W_1^{-1} U_1 R$ $(rp \times n)$ e resolva

$$\hat{C} = O_r(1:p, 1:n) \tag{2.47a}$$

$$O_r(p+1:pr, 1:n) = O_r(1:p(r-1), 1:n)\hat{A} \tag{2.47b}$$

   para $\hat{A}$ e $\hat{C}$.

4. Estime $\hat{B}$, $\hat{D}$ e $\hat{x}_0$ apartir do problema de regreção linear:

$$\begin{aligned} \underset{B,D,x_0}{\arg\min} \frac{1}{N} \sum_{t=1}^{N} \| y(t) - \hat{C}(qI - A)^{-1} B u(t) - D u(t) \\ - \hat{C}(qI - A)^{-1} x_0 \delta(t) \|^2 \end{aligned} \tag{2.48}$$

5. Se o modelo do ruido é buscado, forme $\hat{X}$ como em (2.43) e estime as contribuições do ruido como em (2.44).

Implementação Numérica: A implementação numérica mais eficiente ocorre quando se aplica uma fatorização QR na matriz de dados $\begin{bmatrix} U^T & \Phi^T & y^T \end{bmatrix}^T = LQ$, onde L é uma triz triângular inferior quadrada de dimensões $(pr + mr + s)$ e Q é uma matriz ortonormal $((pr + mr + s) \times N)$.Então a fatorização SVD crucial U_1 em (2.46) pode ser encontrada inteiramente apartir da parte L desta fatoração, ou seja, de uma "matriz menor"[4].

As famílias de métodos contem algumas variáveis de *design*. Algoritmos diferentes correspondem às diferentes escolhas destas variáveis. Estas variáveis são:

- A escolha do vetor de correlação $\varphi_s(t)$ como em (2.34).A maioria dos algoritmos escolhe construir $\varphi_s(t)$ apartir das entradas e saídas passadas com $s_1 = s_2$. Então o escalar s se torna a variável de *design*.
- O escalar r que é o horizonte de predição máximo usado.Muitos algoritmos usam $r = s$, mas não por uma razão particular para essa escolha.
- as matrizes de ponderação W_1 e W_2, talvez as escolhas mais importantes[4].Os algoritimos existentes utilizam as seguuintes escolhas:
  - MOESP, Verhaengen(1994): $W_1 = I$ e $W_2 = \left(\frac{1}{N}\Phi\Pi^{\perp}_{U^T}\Phi^T\right)^{-1}\Phi\Pi^{\perp}_{U^T}$;
  - N4SID,Van Overschee e DeMoor(1994): $W_1 = I$ e $W_2 = \left(\frac{1}{N}\Phi\Pi^{\perp}_{U^T}\Phi^T\right)^{-1}\Phi$;
  - IVM, Viberg(1995): $W_1 = \left(\frac{1}{N}Y\Pi^{\perp}_{U^T}Y\right)^{-\frac{1}{2}}$ e $W_2 = \left(\frac{1}{N}\Phi\Phi^T\right)^{-\frac{1}{2}}$;
  - CVA, Larimore(1990): $W_1 = \left(\frac{1}{N}Y\Pi^{\perp}_{U^T}Y\right)^{-\frac{1}{2}}$ e $W_2 = \left(\frac{1}{N}\Phi\Pi^{\perp}_{U^T}\Phi^T\right)^{-\frac{1}{2}}$.
- A matriz R no passo 3. As escolhas típicas são $R = I$, $R = S_1$ ou $R = S_1^{1/2}$.

2.2.4 Identificação Recursiva

Um algoritmo recursivo de identificação, por definição, estima os parâmetros de um modelo em tempo real. Essa abordagem é útil quando o processo evolui lentamente.

$$y(k) = \psi_k^T(k-1)\theta_k + \xi(k) \tag{2.49}$$

Na equação 2.49 temos a saída $y(k)$ mensurada no instante k como a relação entre a matriz de regressores ψ_k^T baseada nas k - 1 medidas anteriores, o vetor de parâmetros estimados θ_k e o resíduo $\xi(k)$, parcela do modelo que engloba imprecisões (diferenças entre a estrutura presumida e a real) e ruído.

Usando a implementação recursiva do método dos MQ tradicional,temos o vetor de parâmetros de mínimos quadrados descrito pela equação 2.50 , a seguir:

$$\theta_k = \left[\sum_{i=1}^{k} \psi(i-1)\psi^T(i-i)\right]^{-1} \left[\sum_{i=1}^{k} \psi(i-1)y(i)\right] \tag{2.50}$$

Para garantir que θ_k seja uma boa estimativa, existem duas restrições importantes: a covariância de $\theta_\mathbf{k}$ deve ser mínima, e $\theta_\mathbf{k}$ não pode ser polarizado, ou seja, $E[\theta_k] - \theta = 0$.

A matriz $P_k = cov[\theta_k] \in \Re^{n\theta x n\theta}$ denota a matriz de covariância obtida até a interação k.Vale lembrar que, no algoritmo MQ, é interessante expressar as grandezas em termos de estimações passadas. Assim, segundo [6]:

$$P_k = \left[\sum_{i=1}^{k} \psi(i-1)\psi^T(i-1)\right]^{-1}$$

$$P_k^{-1} = \left[\sum_{i=1}^{k} \psi(i-1)\psi^T(i-1)\right] + \psi(i-1)\psi^T(i-1)$$

$$P_k^{-1} = P_{k-1}^{-1} + \psi(i-1)\psi^T(i-1) \tag{2.51}$$

Alterando a parcela $\psi(k-1)$ para ψ_k, a equação 2.50 pode ser reescrita como:

$$\theta_k = P_k \left[\sum_{i=1}^{k} \psi(i-1)y(i) + \psi(k-1)y(k)\right] \tag{2.52}$$

Ou ainda, como:

$$\left[\sum_{i=1}^{k-1} \psi(i-1)\psi^T(i-1)\right]\theta_{k-1} = \left[\sum_{i=1}^{k-1} \psi(i-1)y(i)\right] \tag{2.53}$$

O lado esquerdo da equação 2.53 pode ser representado por $P_{k-1}^{-1}\theta_{k-1}$. Substituindo este resultado na equação 2.52, temos:

$$\theta_k = P_k \left[P_{k-1}^{-1}\theta_{k-1} + \psi(k-1)y(k)\right]$$

$$= P_k \left[\left(P_{k-1}^{-1} - \psi(k-1)\psi^T(k-1)\right)\theta_{k-1} + \psi(k-1)y(k)\right]$$

$$= \theta_{k-1} - P_k\psi(k-1)\psi^T(k-1)\theta_{k-1} + P_k\psi(k-1)y(k)$$

$$= \theta_{k-1} - P_k\psi(k-1)\left[y(k) - \psi^T(k-1)\theta_{k-1}\right]$$

$$\boldsymbol{\theta_k = \theta_{k-1} - K_k \left[y(k) - \psi^T(k-1)\theta_{k-1} \right]} \tag{2.54}$$

Onde,

$$\boldsymbol{K_k = P_k \psi(k-1)}$$

A outra premissa para um bom estimador, relativa a não polarização dos dados, é dimensionada pela matriz $\boldsymbol{K_k}$.

Para usarmos a recursividade do algoritmo, é preciso calcular $\boldsymbol{P_k}$.O uso direto da equação 2.51 mostra que é preciso calcular a inversa de uma matriz, o que é dispendioso, mesmo usando um computador. Aplicando o lema da inversão[5], é possível encontrar uma equação para P_k melhorada:

$$P_k = P_{k-1} - \frac{P_{k-1}\psi(k-1)\psi^T(k-1)P_{k-1}}{\psi^T(k-1)P_{k-1}\psi(k-1)+1} \tag{2.55}$$

Desta forma, K_k pode ser reescrita como:

$$K_k = \frac{P_{k-1}\psi(k-1)}{\psi^T(k-1)P_{k-1}\psi(k-1)+1} \tag{2.56}$$

Existe, porém, uma característica a ser levada em conta: o estimador trata todas as medições de forma idêntica, isto é, o peso associado à primeira medida w_1 é o mesmo que o da centésima, w_{100}, e este é o mesmo da milésima medida, w_{1000}.Para um sistema invariante no tempo, isto é razoável, caso contrário, não o é. É preciso, portanto, inserir plasticidade ao algoritmo, ou seja, ponderar de maneira diferenciada as observações. Uma forma de obter isto é utilizando um decaimento exponencial (2.57) , a seguir:

$$\begin{cases} w_k(k) = 1 \\ w_i(k) = \lambda w_i(k-1), i < k \end{cases} \tag{2.57}$$

O fator λ representa a razão entre pesos consecutivos para uma mesma massa de dados e é conhecido como fator de esquecimento.

O estimador recursivo de mínimos quadrados com fator de esquecimento λ é mostrado

[5]Considere que A e C são matrizes quadradas arbitrárias para as quais existe a inversa, e B é uma terceira matriz tal que BCB^T tem a mesma dimensão de A. Então o chamado lema de inversão de matrizes é dado na forma:
$\left(A^{-1}BCB^T\right) = A - AB\left(B^TAB + C^{-1}\right)^{-1}B^TA$[7]

na equação 2.58 [6]:

$$\begin{cases} K_k = \frac{P_{k-1}\psi_k}{\psi_k^T P_{k-1}\psi_k + \lambda} \\ \theta_k = \theta_{k-1} + K_k[y(k) - \xi_k^T \theta_{k-1}] \\ P_k = \frac{1}{\lambda}(P_{k-1} - \frac{P_{k-1}\psi_k\psi_k^T P_{k-1}}{\psi_k^T P_{k-1}\psi_k + \lambda}) \end{cases} \tag{2.58}$$

2.2.5 Modelo da parede

Um processo térmico tem como características a lentidão, a variação no tempo dos seus parâmetros e a necessidade de se trabalhar com múltiplas entradas e saídas. Desta forma, é interessante trabalhar com um modelo matemático simplificado, tanto para o tratamento de dados, quanto para definir quais variáveis são mais importantes.

O modelo de transmissão de calor através de uma parede é mostrada na figura 2.6:

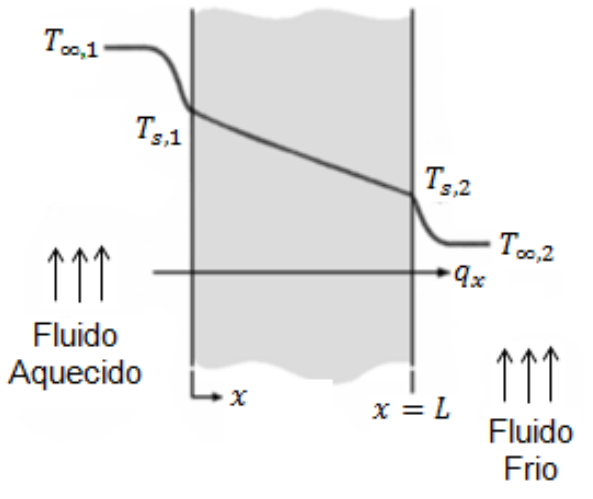

Figura 2.6 – Esquemático das trocas de calor por uma parede

Onde:

$T_{\infty,i} \rightarrow$ temperatura estacionária no ambiente i;

$T_{S,i} \rightarrow$ temperatura na superfície i;

$x \rightarrow$ posição ao longo da parede;

$q_x \rightarrow$ vetor de transferência de calor pela parede em função da posição x.

A parede de uma sala separa dois fluidos a temperaturas distintas. O processo de trocas de calor se dá em três fases: convecção do calor do fluido à $T_{\infty,1}$ para a superfície da parede à $T_{S,1}$,condução através da parede pelo comprimento L, e, novamente, convecção da outra superfície da parede com temperatura $T_{s,2}$ para o fluido de temperatura $T_{\infty,2}$.

Temos, pela lei de Fourier, que a transferência de calor por condução é:

$$q_x = -kA\frac{dT}{dA} = \frac{kA}{L}\left(T_{S,1} - T_{S,2}\right) \tag{2.59}$$

Onde,

$A \rightarrow$ área da parede;

$L \rightarrow$ espessura da parede, e;

$k \rightarrow$ constante de condutividade térmica $\left[\frac{\mathrm{W}}{\mathrm{mK}}\right]$.

Já o fluxo de calor é dado por:

$$q_x^{"} = \frac{q_x}{A} = \frac{k}{L}(T_{S,1} - T_{S,2}) \tag{2.60}$$

As equações (2.59) e (2.60) mostram que tanto a transferência quanto o fluxo de calor são constantes e independentes da posição ao longo da parede.

A transferência de calor pelas paredes deve ser investigada. Iniciaremos com um modelo simples, $2R1C$, modelo de primeira ordem, com características lineares. É bom lembrar que, para a consideração de linearidade, definiremos um ponto de operação do sistema. Nesse modelo, a parede é modelada por uma capacitância térmica interna e dois elementos de condutividade térmica R.

Existe um conceito amplamente usado em modelagem de ambientes denominado resistência térmica. Trata-se de recursos matemáticos baseados na analogia entre a difusão do calor e a carga elétrica (potencial elétrico e de transmissão de calor, corrente e transferência de calor), de modo a descrever um sistema térmico como um circuito elétrico. Da mesma maneira que se associa uma resistência elétrica com a condução de eletricidade, associa-se uma resistência térmica com à condução de calor. A resistência elétrica, pela lei de Ohm, é definida como a razão entre tensão (diferença de potencial elétrico) e corrente. A resistência térmica por condução é, portanto, a razão entre a diferença de temperatura entre dois pontos e a transferência de calor – veja a equação (2.61), logo abaixo:

$$R_{t,cond} = \frac{T_{S,1} - TS,2}{q_x} = \frac{L}{kA} \tag{2.61}$$

É preciso definir a resistência térmica por convecção. Usaremos a lei de resfriamento de Newton, para definir a transferência de calor, ou seja:

$$q = hA(T_S - T_\infty) \tag{2.62}$$

Onde h é o coeficiente de transferência de calor constante que depende de propriedades físicas do fluido.

Assim:

$$R_{t,conv} = \frac{T_S - T_\infty}{q} \tag{2.63}$$

Considerando que pelo princípio da conservação de energia, assim como na lei de nós de Kirchhoff, a transferência de calor ao longo da parede é constante, ou seja:

$$q_x = \frac{T_{\infty,1} - T_{S,1}}{1 \backslash h_1 A} = \frac{T_{S,1} - T_{S,2}}{L \backslash K A} = \frac{T_{S,1} - T_{\infty,2}}{1 \backslash h_2 A} \tag{2.64}$$

Além disso, em termos de diferença de temperatura, $T_{\infty,1} - T_{\infty,2}$, podemos definir uma resistência total R_{tot}, a qual será a soma de todas as resistência térmicas na malha, uma vez que elas estão em série. A equação (2.65) expressa o valor de R_{tot} :

$$R_{tot} = \frac{1}{h_1 A} + \frac{L}{KA} + \frac{1}{h_2 A} \tag{2.65}$$

A figura 2.7 a seguir ilustra o conceito de resistência térmica aplicado à modelagem da parede:

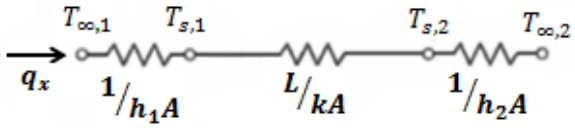

Figura 2.7 – Malha representativa da parede.

A equação (2.66) abaixo descreve a transferência de calor entre as salas vizinhas[2]:

$$\frac{T_o}{T_{iv}}(s) = \frac{1}{sRC_v + 1} \tag{2.66}$$

Vemos que se trata de um sistema dinâmico, cuja função de transferência é de 1ª ordem, e a constante de tempo é $\tau = RC_v$, no caso, representa o tempo necessário para atingir o equilíbrio térmico entre duas vizinhanças. O resfriamento promovido pelo ar condicionado também tem a mesma estrutura da equação (2.66), e a constante de tempo associada retrata o efeito de resfriamento da sala.

As características físicas da parede, do piso e do teto são diferentes entre si e relevantes para o problema. Para efeitos de simplificação, considera-se que o modelo $2R1C$ na figura 2.8 é suficiente para representar a dinâmica dominante.

O ponto T_{iv} retrata um ponto fictício no meio da parede. A diferença entre T_v e T_o define

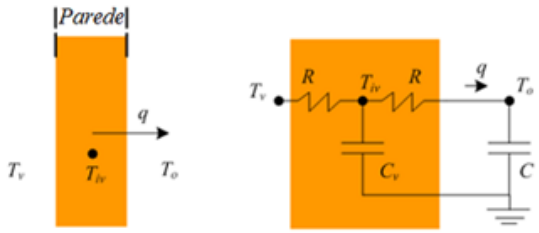

Figura 2.8 – Modelo 2R1C da parede[2].

o vetor $\vec{q}$, definindo o fluxo de calor entre duas vizinhanças. Os parâmetros C_v e R são intrínsecos das características da parede, tais como material, suas dimensões e seu formato.

A figura 2.9 mostra a relação entre varáveis de saída e de entrada, um rascunho para a identificação, baseada na superposição das contribuições térmicas descritas anteriormente.

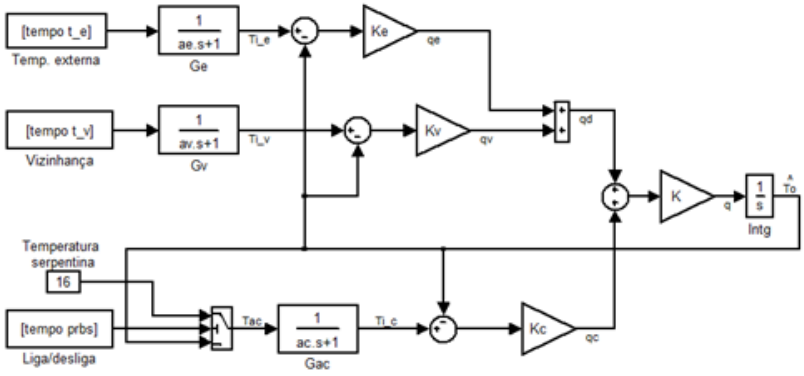

Figura 2.9 – Diagrama de entradas e saídas simplificado, relativo à sala de reunião [3], em que o significado e dedução de cada termo podem ser encontrados na respectiva obra.

2.3 CARGA TÉMICA

O calor sensível é o calor fornecido a um corpo que não provoca mudança de fase, ou seja, provoca somente mudança de temperatura. Já o calor latente está relacionado a mudança de fase.

A quantidade de calor sensível e latente que deve ser retirada ou colocada em um recinto visando atingir as condições de conforto é chamada de carga térmica. Essa carga térmica pode ter diferentes fontes, como por exemplo:

- Condução;
- Insolação;
- Dutos;

- Pessoas e
- Ventilação.

2.3.1 Carga térmica devido à presença de pessoas

Os seres humanos emitem tanto calor latente como calor sensível. Conforme a atividade que o indivíduo esteja realizando, a quantidade de calor emitida varia.

Além da atividade, outros fatores também influenciam a troca de calor entre as pessoas e o meio. Uma pessoa possui uma temperatura média de 36,5 ºC. Dependendo da temperatura externa, a quantidade de calor latente e sensível é maior. Isso ocorre devido ao mecanismo termostático que atua sobre o metabolismo para manter a temperatura do corpo aproximadamente constante, independente das variações nas condições externas.

De acordo com a NBR 6401 (Instalações centrais de ar-condicionado para conforto - Parâmetros básicos de projeto) em sua tabela 12, o calor sensível gasto por uma pessoa em um ambiente universitário e de escritório para uma temperatura de 21 ºC é de 71 kcal/h. Essa quantidade de calor deve ser levada em conta no controle do ar-condicionado, a fim de reduzir ou eliminar a influência do número de pessoas no sistema térmico.

Existem diversas soluções para determinar a presença de pessoas em um ambiente e assim determinar a carga térmica devido ao número de pessoas.Entre as soluções está a detecção apartir do imagens do ambiente capturadas por câmeras e tratadas por algoritmos de visão computacional.

2.3.2 Visão Computacional

Visão computacional é o processo que utiliza máquinas para entender e analisar conjuntos de imagens (tanto vídeos quanto imagens estáticas). Algoritmos de processamento de imagens estão disponíveis, em várias formas, desde a década de 1960, mas avanços recentes nos algoritmos de Machine Learning assim como em armazenamento de dados, capacidade computacional, barateamento de câmeras de alta qualidade proporcionaram grandes melhorias em como podemos desenvolver softwares especializados em visão computacional.

A realização das tarefas típicas da visão computacional possui dois núcleos básicos:

- **Classificação de objetos:**o modelo é treinado com dataset de objetos específicos e o modelo treinado classifica novos objetos pertencendo, ou não, às categorias treinadas.
- **Detecção de objetos:**o modelo identifica características específicas de um objeto.

Avanços recentes nas ferramentas utilizadas levaram a algoritmos cada vez mais eficien-

tes como as Redes Neurais Convolucionais que serão tratadas nas próximas seções.

2.3.2.1 Redes Neurais Convolucionais

Uma Rede Neural Convolucional (CNN) é um algoritmo de aprendizado profundo que tem como entrada uma imagem, designa importância (pesos(w) e viés(b)) para vários aspectos e/ou objetos presentes em na imagem e é capaz de diferenciá-los. A etapa de pre-processamento requerida em uma CNN é muito menor se comparada com outros algoritmos de classificação.Enquanto em métodos mais antigos os filtros aplicados nas imagens são calculados de forma manual, CNN's, com treino suficiente, possuem a habilidade de aprender os parâmetros desses filtros[8].

A arquitetura básica de uma CNN é mostrada na figura 2.10.

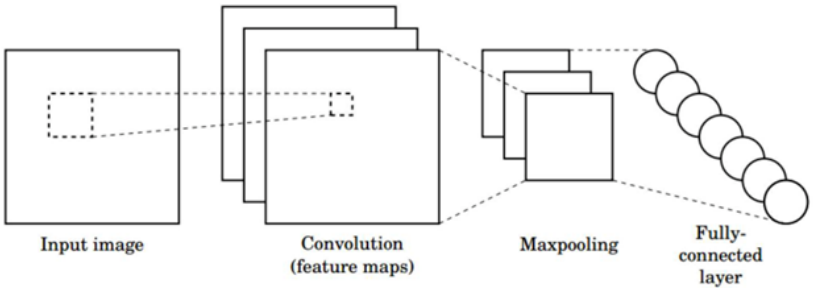

Figura 2.10 – Arquitetura básica de uma CNN.

A arquitetura utilizada para o fim de classificação pode ser vista na figura 2.11 :

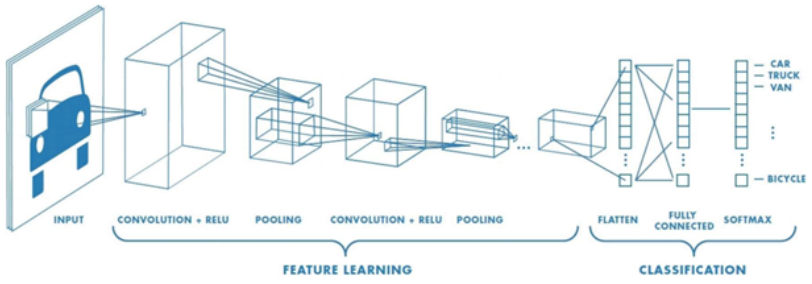

Figura 2.11 – Arquitetura CNN para classificação.

Para a implementação dos modelos de CNN's existem diversas ferramentas. Neste trabalho se optou por utilizarmos a API implementada em *TensorFlow*.

2.3.2.2 TensorFlow

TensorFlow é uma plataforma de código aberto para Aprendizado de Máquina (ML). Possui um ecossistema compreensivo e flexível de ferramentas que permite a implementação dos mais diversos algoritmos de ML.

Uma das características mais interessantes do TensorFlow é sua capacidade de rapidamente gerar um produto ou serviço a partir do modelo preditivo treinado, eliminando a necessidade de reimplementar o modelo.

O TensorFlow é multiplataforma e pode ser usado no Windows, MacOS e Linux. Ele é executado em quase tudo: CPUs e GPUs – incluindo plataformas móveis e integradas – e até mesmo Tensor Processing Units (TPU), que são hardware especializado para fazer a matemática de um tensor (um objeto matemático multidimensional). O TensorFlow é projetado para ser escalável em vários computadores, bem como várias CPUs e GPUs dentro de máquinas individuais. Embora a implementação de código aberto original não tivesse recursos distribuídos após a liberação, a partir da versão 0.8.0 a funcionalidade de execução distribuída ficou disponível como parte da biblioteca TensorFlow. Embora esta API distribuída seja um pouco pesada, é incrivelmente poderosa. A maioria das outras bibliotecas de aprendizagem de máquina não possui esses recursos e é importante observar que a compatibilidade nativa com determinados gerenciadores de cluster está sendo trabalhada.

2.4 CONTROLADORES

Um sistema de controle é definido por Murray et al. (2003) como uma peça onde uma variável medida é usada para alterar o sistema por meio de computação e atuação. De maneira simples, a automação usa a realimentação como uma ferramenta de gerenciamento de incertezas, sejam elas paramétricas, de condições de funcionamento, de limites práticos ou advindas de alteração de sianis externos (ruído, perturbação) não controlados. Justamente esta visão, de sistemas de controle como uma maneira de conferir robustez frente a uma incerteza, explica o porquê do controle realimentado estar disseminado por todas as tecnologias do mundo moderno.

2.4.1 Controlador Liga-desliga(On-Off)

Sendo uma das técnicas mais clássicas de controle, este tipo de controlador se caracteriza por uma resposta rápida e uma implementação simples. Leva em conta o erro entre a referência e a saída do sistema para ajustar o sinal de controle[9].

Este tipo de controlador não garante que o sinal se mantenha na referência, mas sim numa faixa em torno do valor desejado, de modo que o valor médio corresponda a referência desejada[10].

Neste tipo de ação o controlador é modelado por um relé conforme mostra a figura 2.12:

O sinal de controle $u(t)$ pode assumir apenas dois valores, conforme o erro seja positivo

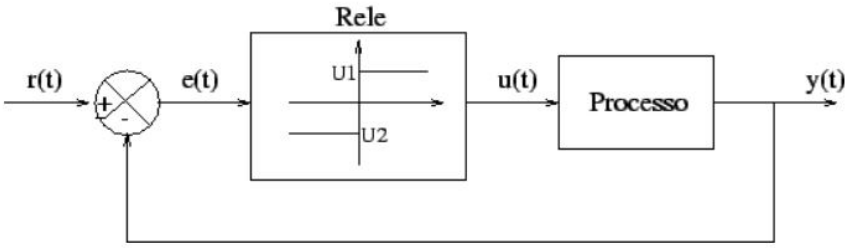

Figura 2.12 – Malha de um controlador Liga-desliga.

ou negativo. Em outras palavras tem-se:

$$u(t) = \begin{cases} U_1 \text{ se } e(t) > 0 \\ U_2 \text{ se } e(t) < 0 \end{cases} \tag{2.67}$$

Este tipo de função pode ser implementada como um simples comparador ou mesmo um relé físico. Note que neste caso teríamos uma inconsistência em zero e, na presença de ruídos, teríamos chaveamentos espúrios quando o sinal $e(t)$ for próximo de zero[11].

Para evitar este tipo de problema, utiliza-se na prática o que chamamos de controlador liga-desliga com histerese mostrado na figura 2.13.

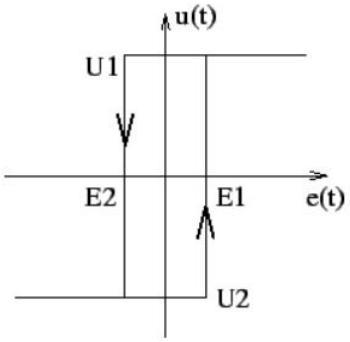

Figura 2.13 – Histerese

Com este tipo de controlador temos o seguinte comportamento:

Se $u(t) = U_1$, é necessário que o valor de $e(t)$ desça abaixo de $-E_2(t)$ para que haja um chaveamento para U_2. Se $u(t) = U_2$, é necessário que o valor de $e(t)$ ultrapasse o valor de E_1 para que haja um chaveamento para U_1.

O gráfico da figura (2.14) mostra a curva de resposta em malha fechada e o respectivo sinal de controle para um sistema com controlador liga-desliga com histerese. Note que, em regime permanente, a saída do sistema apresenta uma oscilação em torno do valor de referência. Este fato denota a baixa precisão obtida com este tipo de controlador. A amplitude e a frequência da oscilação são funções do intervalo $[E_1, E_2]$. A determinação do intervalo $[E_1, E_2]$ deve ser feito levando-se em consideração a precisão desejada, os níveis de ruído e

a vida útil dos componentes.

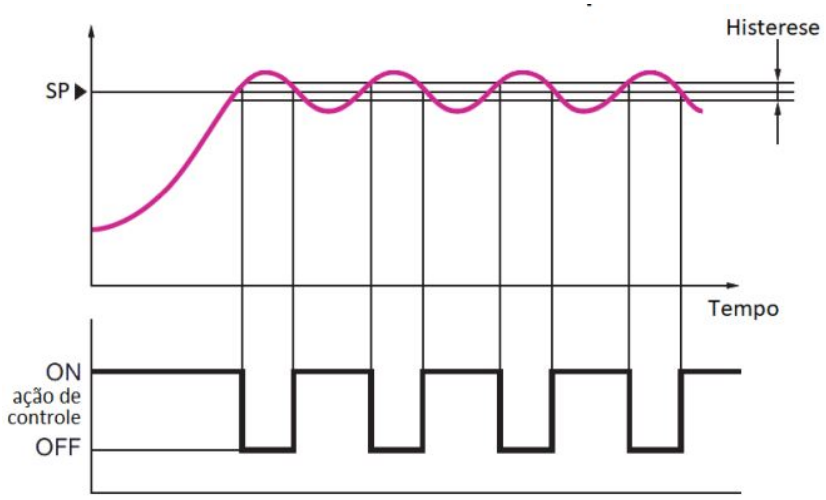

Figura 2.14 – Resposta ao controle com histerese.

A ação de controle liga-desliga pode assim ser considerada a ação de controle mais simples e mais econômica. Entretanto, este tipo de ação possui limitações no que diz respeito ao comportamento dinâmico e em regime permanente do sistema em malha fechada. Suas aplicações restringem-se a sistemas onde não é necessário precisão nem um bom desempenho dinâmico.

2.4.2 Controlador PI

A principal função da ação integral é fazer com que processos do tipo 0 sigam, com erro nulo, um sinal de referência do tipo salto. Entretanto, a ação integral se aplicada isoladamente tende a piorar a estabilidade relativa do sistema. Para contrabalançar este fato, a ação integral é em geral utilizada em conjunto com a ação proporcional constituindo-se o controlador PI, cujo sinal de controle é dado por:

$$u(t) = K_p \left(e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau \right) \tag{2.68}$$

Onde $u(t)$ representa a saída do controle, K_p é o ganho proporcional, T_i é o ganho integral e $e(t)$ é o erro da variável(erro definido pela diferença entre o valor do setpoint com a variável de processo).

Aplicando a transformada de Laplace em (2.68), temos:

$$G_{PI}(s) = \frac{U(s)}{E(s)} = K_p \left(1 + \frac{1}{sT_i} \right) \tag{2.69}$$

Diante disso, é necessário determinar os parâmetros do controlador (K_p, T_i) para obter-se

uma resposta satisfatória que atenda as necessidades do processo.

Existem diversos métodos de sintonia para o controlador PI, neste trabalho foi adotado o método Chien, Hrone e Reswick (CHR).

2.4.2.1 Método CHR

O método de Chien-Hrones-Reswick (K. L. Chien, J. A. Hrones e J. B. Reswick, 1952) propõe regras de ajuste diferentes para problemas do tipo servo (mudanças no setpoint) e para problemas do tipo regulador (perturbações externas com setpoint constante). Também são propostos dois critérios de desempenho, sendo eles a resposta mais rápida sem sobrevalor (CHR - 0%) e a resposta mais rápida com 20% do sobrevalor (CHR-20%)[12].

Tabela 2.1 – Sintonia empírica de controlador P, PI, PID – modelo CHM.

Tipo de Controlador	Com *overshoot* de 0%			Com *overshoot* de 20%		
	Kp	*Ti*	*TD*	*Kp*	*Ti*	*TD*
P	*0.3a*	-	-	*0.7a*	-	-
PI	*0.35a*	*1.2 T*	-	*0.6a*	*T*	-
PID	*0.6a*	*T*	*0.5 L*	*0.95a*	*1.4 T*	*0.47 T*

Onde:

L é o atraso de transporte;

T é a constante de tempo e;

a é a inclinação de uma reta assintótica à curva de saída do controlador ($a = \frac{KL}{T}$).

2.4.3 Controlador Adaptativo

Um controlador adaptativo[13] é aquele que pode modificar seu comportamento de acordo com a variação da dinâmica do processo. Este tipo de controlador é caracterizado por possuir um mecanismo de ajuste que faz com que a lei de controle se adapte a estas variações, como pode ser visto na figura 2.15.

O diagrama presente na figura 2.15 possui uma estrutura complexa. Cada ponto pode ser realizado de várias formas distintas. Nota-se que controle adaptativo é mais um conceito do que uma técnica propriamente dita. Para fins de simplificação, pode-se dizer que existem duas abordagens típicas: controle direto e indireto.

No controle indireto, primeiramente os parâmetros do processo são estimados de forma recursiva e com estes parâmetros são calculados os parâmetros do controlador, como visto na figura 2.15. No controle direto, o processo é re-parametrizado, isto é, o estimador é utilizado para obter os parâmetros do controlador a partir das medidas de entrada e saída do sistema,

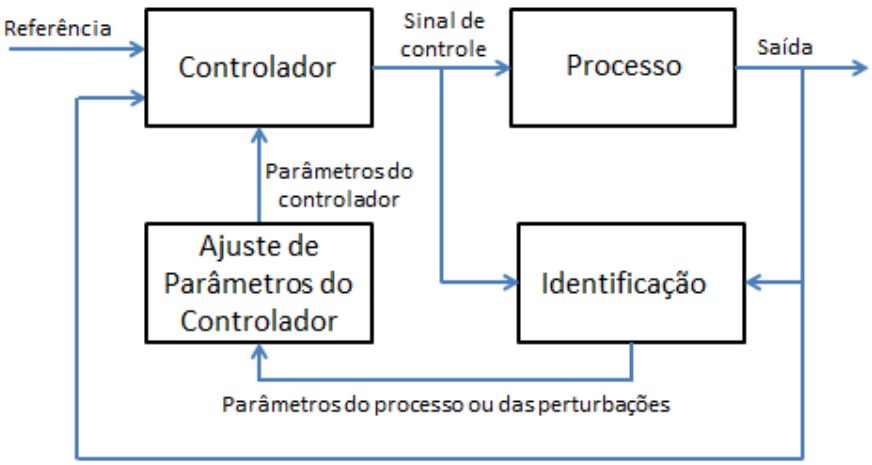

Figura 2.15 – Esquemático geral de controle adaptativo

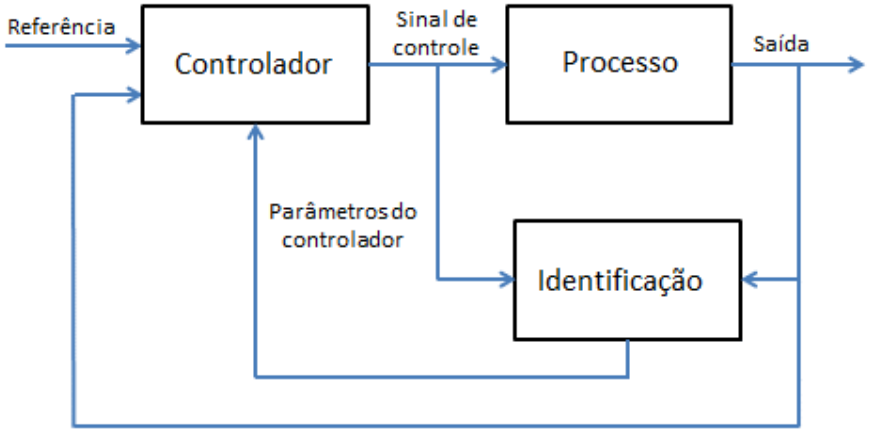

Figura 2.16 – Esquemático de controle adaptativo direto.

conforme pode ser visto na figura 2.16. Em ambos os algoritmos, existe a ideia básica de identificar alguns parâmetros que estejam relacionados ou ao processo, ou às especificações para o sistema em malha fechada, e, preferencialmente, aos dois. Vê-se que a identificação assume um papel importante.

Existem várias técnicas para o projeto de um controlador adaptativo: modelo de referência (MRAC), reguladores auto ajustáveis (RST), controle dual, escalonamento de ganho, entre outros. Neste trabalho, foi utilizada a técnica de reguladores auto ajustáveis, detalhada logo a seguir.

Assumindo o modelo do processo como ARMAX(do inglês *autoregressive moving average with exogenous input*)[6] SISO (single-input, single-output):

$$A(q)y(k) = B(q)u(k) + C(q)\nu(k) \tag{2.70}$$

ou,alternativamente,

Com $C(q) = B(q)$, temos:

$$y(k) = \frac{B(q)}{A(q)}u(k) + \frac{B(q)}{A(q)}\nu(k) \tag{2.71}$$

Onde A e B são polinômios, q é o operador de atraso, e ν é uma perturbação ao sistema.

Deseja-se encontrar um controlador como o mostrado na figura 2.17[13]:

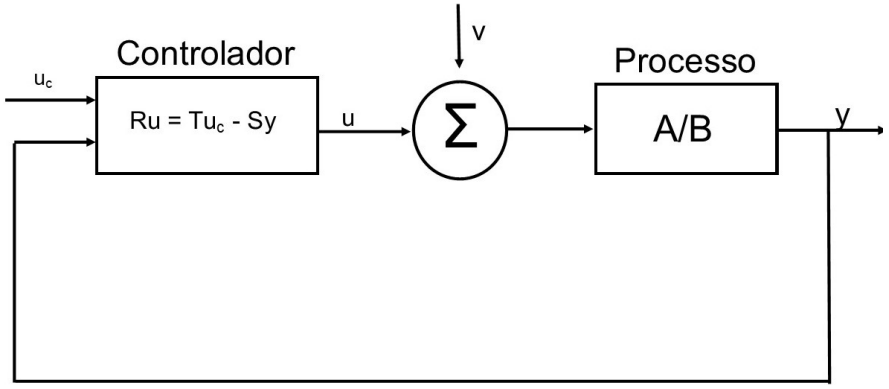

Figura 2.17 – Um controlador geral com dois graus de liberdade.

A equação () mostra a forma do controlador desejado:

$$R(q)u(k) = T(q)y_{ref}(k) - S(q)y(k) \tag{2.72}$$

Onde R,S e T são polinômios. Logo, em malha fechada, temos:

$$\begin{aligned} y(k) &= \frac{B(q)T(q)}{A(q)R(q) + B(q)S(q)}y_{ref}(k) + \frac{B(q)R(q)}{A(q)R(q) + B(q)S(q)}\nu(k) \\ u(k) &= \frac{A(q)T(q)}{A(q)R(q) + B(q)S(q)}y_{ref}(k) + \frac{B(q)S(q)}{A(q)R(q) + B(q)S(q)}\nu(k) \end{aligned} \tag{2.73}$$

A equação caracteristica em malha fechada é então,negigenciando q para facilitar a leitura:

$$AR + BS = A_C \tag{2.74}$$

A idéia principal desse método é especificar o polinômio caracteristo A_C em malha fe-

chada.Os polinômios R e SS podem ser obtidos de (2.74).Essa equação possui solução se os polinômios A e B não possuirem fatores comuns. As soluções podem ser pobremente condicionadas se esses polinômios possuirem fatores que são prôximos.As soluções podem ser encontradas iintroduzindo polinômios com fatores desconhecidos e resolvendo as equações lineares obtidas.

A equação (2.74) determina apenas os polinômios R e S.Para determinar T outras condições devem ser introduzidas.Para que isso ocorra é preciso que a resposta do sinal de comando u_c para as saídas possam ser descritas pela dinâmica

$$A_m y_m(t) = B_m u_c(t) \tag{2.75}$$

Isso se segue da equação (2.73) que a seguinte condição deve se manter:

$$\frac{BT}{AR + BS} = \frac{BT}{A_C} = \frac{B_m}{A_m} \tag{2.76}$$

As consequências da condição de seguidor de modelo em (2.76) implica que há fatores de cancelamento de BT e A_C. Fatore o polinômio B como:

$$B = B^+ B^- \tag{2.77}$$

onde B^+ é um polinômio mônico cujos zeros são estáveis e tão bem condicionados que podem ser cancelados pelo controlador e B^- corresponde aos zeros instáveis ou poucamente condicionados que não podem ser cancelados. Isso mostra que B^- deve ser um fator de B_m,logo:

$$B_m = B^- B_m^{'} \tag{2.78}$$

Já que B^+ é cancelado, deve ser um fator de A_c. Com isso, também temos que A_m também é fator de A_c. O polinômio característico tem a forma:

$$A_C = A_0 A_m B^+ \tag{2.79}$$

Como B^+ é um fator de B e de A_C, tem se que de (2.74) também pode dividir R, logo:

$$R = R^{'} B^+ \tag{2.80}$$

e a equaçao diofantina (2.74) se reduz a:

$$AR^{'} + B^{-}S = A_0 A_m = A_C^{'} \tag{2.81}$$

Introduzindo as equações (2.77),(2.78) e (2.79) em (2.76), obtemos:

$$T = A_0 B_m^{'} \tag{2.82}$$

Para que o controlador seja causal as seguintes condições devem ser satisfeitas:

$$\begin{cases} \text{Grau de } A_m = \text{Grau de } A \\ \text{Grau de } B_m = \text{Grau de } B \\ \text{Grau de } A_0 = \text{Grau de } A - \text{Grau de } B^{+} - 1 \\ \text{Grau de } S < \text{Grau de } A \end{cases} \tag{2.83}$$

Para eliminar o erro em regime permanente, é proposto em Åström e Wittenmark(1995):

$$R^*(q^{-1})\Delta^* u(k) + S^*(q^{-1})\Delta^* y(k) + A_0(1)A_m(1)y(k) = A_0^*(q^{-1})A_m(1)y_{ref}(k) \tag{2.84}$$

Em que a dedução e o significado de cada termo de (2.85) podem ser encontrados em sua respectiva obra.

2.4.4 Controle Preditivo

O controle preditivo baseado em modelo (MPC) surge atualmente como uma das mais populares e eficientes estratégias de controle na indústria de processos. Muitos dos aspectos fundamentais num projeto de controle industrial prático podem ser explorados em um controle preditivo baseado em modelo, como a trajetória de referência futura, previsão de perturbações e a possibilidade de inclusão de restrições, verificando a flexibilidade desta técnica de controle[14][15].

A metodologia de todos os controladores pertencentes à família do MPC é caracterizada pela seguinte estratégia, representada na figura 2.18 [14].

As saídas futuras $\hat{y}(t+k|t)$ com $k = 1, \ldots, N$ para um determinado horizonte de predição N são preditas a cada instante t usando o modelo do processo. As predições das saídas dependem dos valores das entradas e saídas até o instante t e do conjunto dos sinais de controles preditos $u(t+k|t)$ com $k = 1, \ldots, N-1$, que são aqueles que serão calculados e

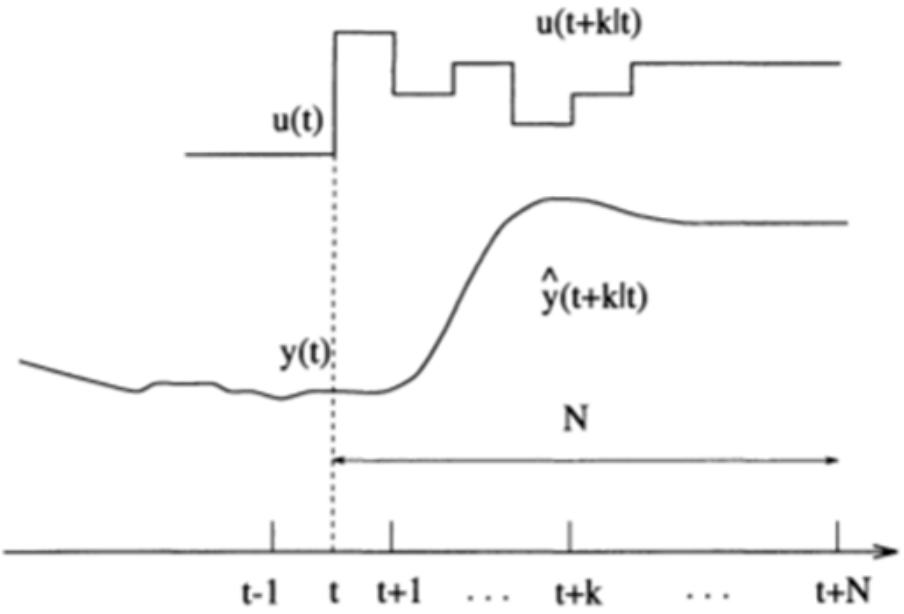

Figura 2.18 – Estratégia do MPC

mandados para o sistema.

1. O conjunto de valores futuros do sinal de controle é calculado pela otimização de uma determinada função de custo de forma a manter o sistema o mais próximo possível da trajetória de referência. A função de custo geralmente possui a forma de uma função quadrática do erro entre as saídas preditas e a trajetória de referência.

2. O sinal de controle $u(t|t)$ é enviado ao processo enquanto o próximo sinal de controle é rejeitado porque a próxima amostra $y(t+1)$ já é conhecida e o passo um é repetido com esse novo valor e todos os valores são atualizados.

A estrutura básica de um MPC é encontrada na figura 2.19, onde os principais elementos são:

- **Trajetória de Referências:** representa o comportamento do sinal de saída desejado no futuro. O conhecimento prévio dessas trajetórias garante uma característica preditiva.

- **Modelo:** representação matemática do processo capaz de representar o comportamento dinâmico do processo de forma suficientemente precisa. O modelo pode ser linear ou não-linear e podendo, ainda, ser atualizado por meio de algoritmos de identificação *online* conferindo ao controlador uma característica adaptativa.

- **Preditor:** fornece uma estimativa da saída futura com base na informação atual da planta.

- **Otimizador:** minimiza a função de custo a cada período de amostragem de forma a obter uma ação de controle adequada aos requisitos do sistema. A função de custo a ser minimizada pode contemplar, além de parcelas relativas ao erro de previsão e a mudanças no sinal de controle, outros termos que forneçam ao controlador propriedades que melhorem seu desempenho em relação às particularides do processo.

- **Função de custo:** Os diferentes algoritmos que implementam o MPC propõem diferentes funções de custo para a obtenção da lei de controle. O objetivo geral é que a saída futura(y), no horizonte considerado, siga a trajetória de referência(w) e ao mesmo tempo o esforço do controlador (Δu) para que isto ocorra seja penalizado. A expressão generalizada pode ser vista na equação (2.85).Onde N_1 e N_2 são os horizontes de custo mínimo e máximo respectivamente e N_u é o horizonte de controle. Os coeficientes $\delta(j)$ e $\lambda(j)$ são os pesos considerados nos valores futuros.

$$J(N_1, N_2, N_u) = E\left\{\sum_{j=N_1}^{N_2} \delta(j)\left[y(t+j|t) - w(t+j)\right]^2 + \sum_{j=1}^{N_u} \lambda(j)\left[\Delta u(t+j-1)\right]^2\right\} \tag{2.85}$$

- **Restrições:** São as restrições impostas ao sistema.Podem ser tanto nos atuadores quanto nas saídas.

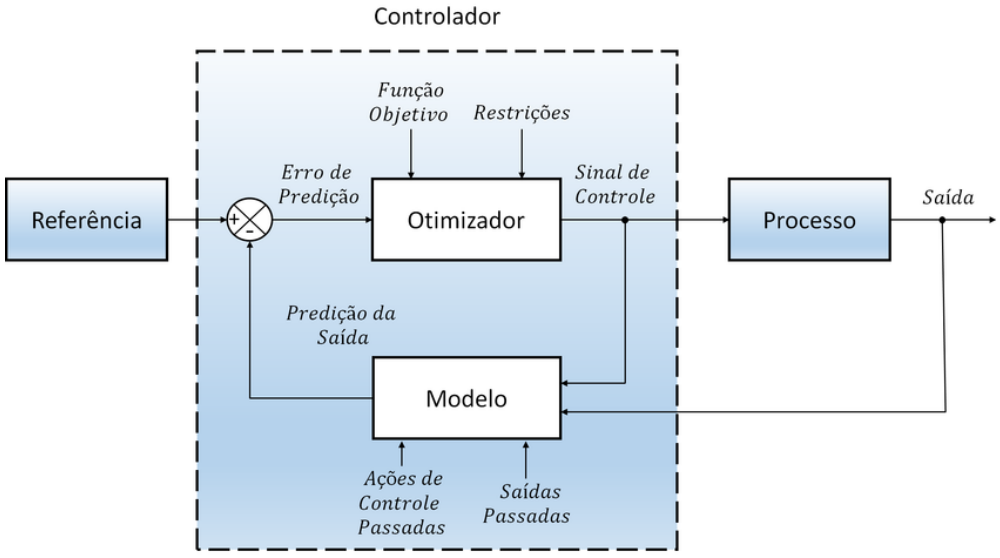

Figura 2.19 – Estrutura básica de um MPC.

2.4.4.1 Controle Preditivo Generalizado(GPC)

A ideia básica do GPC é calcular uma sequência de valores futuros para o sinal de controle de maneira a minimizar a função de custo (2.85) definida em um dado horizonte de predição.

Considerando a planta do sistema na forma:

$$A(z^{-1})y(t) = z^{-d}B(z^{-1})u(t-1) + C(z^{-1})e(t) \tag{2.86}$$

onde $y(t)$ é a saída do sistema no instante t, $u(t)$ é o sinal de controle no instante t, $e(t)$ é o ruido branco e d é o tempo morto do sistema. A, B, C e D são polinômios na seguinte forma:

$$A(z^{-1}) = 1+a_1z^{-1}+a_2z^{-2}+\cdots+a_{an}z^{-an} B(z^{-1}) = b_0+b_1z^{-1}+b_2z^{-2}+\cdots+b_{bn}z^{-bn} C(z^{-1}) = 1+c_1z^{-1}+c$$

(2.87)

Este modelo é conhecido como *Controller Auto-Regressive Moving-Average* (CARMA). De acordo com [16], para muitas aplicações industriais em que as perturbações são não-estacionarias um modelo CARMA integrado (CARIMA) é mais apropriado. O modelo CARIMA é dado por:

$$A(z^{-1})y(t) = z^{-d}B(z^{-1})u(t-1) + C(z^{-1})\frac{e(t)}{1-z^{-1}} \tag{2.88}$$

Por simplicidade o polinômio C foi escolhido como igual a 1[14].

O algoritmo GPC consiste na aplicação de uma sequência de controle que minimiza a função de custo em (2.85).

Para otimizar a função de custo a predição ótima de y(t+j) será obtida para $j > N_1$ e $j < N_2$. Considere a seguinte equação Diofantina:

$$1 = E_j(z^{-1})\tilde{A}(z^{-1}) + z^{-j}F_j(z^{-1}), \text{ onde } \tilde{A}(z^{-1}) = \Delta A(z^{-1}) \tag{2.89}$$

Os polinômios E_j e F_j são unicamente definidos e possuem grau j-1 e na respectivamente. Podem ser obtidos a partir da divisão de 1 por $\tilde{A}(z^{-1})$ até que o resto possa ser fatorado como $z^{-j}F_j(z^{-1})$. O quociente dessa divisão é o polinômio $E_j(z^{-1})$.

Multiplicando (2.88) por $\Delta E_j(z^{-1})z^{-j}$ temos:

$$\tilde{A}(z^{-1})E_j(z^{-1})y(t+j) = E_j(z^{-1})B(z^{-1})\Delta u(t+j-d-1) + E_j(z^{-1})e(t+j) \tag{2.90}$$

Considerando (2.89), a equação (2.90) pode ser reescrita na forma:

$$(1 - z^{-j}F_j(z^{-1}))y(t+j) = E_j(z^{-1})B(z^{-1})\Delta u(t+j-d-1) + E_j(z^{-1})e(t+j)$$

$$y(t+j) = F_j(z^{-1})y(t) + E_j(z^{-1})B(z^{-1})\Delta u(t+j-d-1) + E_j(z^{-1})e(t+j) \quad (2.91)$$

Como o grau de $E_j(z^{-1})$ é $j-1$, o temos do ruído estão todos no futuro. As melhores predições para a saída são dadas por:

$$\hat{y}(t+j|t) = G_j(z^{-1})\Delta u(t+j-d-1) + F_j(z^{-1})y(t), \text{ onde, } G_j(z^{-1}) = E_j(z^{-1})B(z^{-1})$$

Os polinômios E_j e F_j podem ser obtidos de forma recursiva como mostrado em [16].

$$\begin{aligned}
\hat{y}(t+d+1|t) &= G_{d+1}\Delta u(t) + F_{d+1}y(t) \\
\hat{y}(t+d+2|t) &= G_{d+2}\Delta u(t+1) + F_{d+2}y(t) \\
&\vdots \\
\hat{y}(t+d+N|t) &= G_{d+N}\Delta u(t+N-1) + F_{d+N}y(t)
\end{aligned} \quad (2.92)$$

Podemos escrever (2.92) na forma matricial:

$$\hat{Y} = Gu + F(z^{-1})y(t) + G'(z^{-1})\Delta u(t-1) \quad (2.93)$$

onde:

$$\hat{Y} = \begin{bmatrix} \hat{y}(t+d+1|t) \\ \hat{y}(t+d+2|t) \\ \vdots \\ \hat{y}(t+d+N|t) \end{bmatrix} \quad u = \begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+N-1) \end{bmatrix} \quad G = \begin{bmatrix} g_0 & 0 & \dots & 0 \\ g_1 & g_0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ g_{N-1} & g_{N-2} & \dots & g_0 \end{bmatrix}$$

$$G' = \begin{bmatrix} (G_{d+1}(z^{-1} - g_0)z \\ (G_{d+2}(z^{-1} - g_0 - g_1z^{-1})z^2 \\ \vdots \\ (G_{d+N}(z^{-1} - g_0 - g_1z^{-1} - \dots - g_{N-1}z^{-(N-1)})z^N \end{bmatrix}$$

$$F(z^{-1}) = \begin{bmatrix} F_{d+1}(z^{-1}) \\ F_{d+2}(z^{-1}) \\ \vdots \\ F_{d+N}(z^{-1}) \end{bmatrix}$$

Os últimos dois termos de (2.93) somente depende do passado e podem ser incorporados em f, fazendo com que:

$$\hat{Y} = Gu + f$$

A equação (2.85) pode ser escrita como:

$$J = (Gu + f - w)^T(Gu + f - w) + \lambda u^T u \tag{2.94}$$

Onde,

$$w = \begin{bmatrix} w(t+d+1) & w(t+d+2) & \dots & w(t+d+N) \end{bmatrix}$$

A equação (2.94) pode ainda ser escrita como:

$$J = \frac{1}{2}u^T H u + b^T u + f_0 \tag{2.95}$$

Onde,

$$\begin{aligned} H &= 2(G^T G + \lambda I) \\ b^T &= 2(f - w)^T G \\ f_0 &= (f - w)T(f - w) \end{aligned}$$

Para encontrar a solução ótima é preciso minimizar a função J sujeita as restrições impostas. Para isso é utilizado o algoritmo KWIK(*Knows What It Knows*)[17].

2.5 MODULAÇÃO DE LARGURA DE PULSO(PWM)

Modulação por largura de pulso, ou mais conhecida como pulse-width modulation (PWM), é uma técnica de modulação em que a largura do pulso é adequada a um sinal de informa-

ção. Ela pode ser usada em sistemas de comunicação, mas seu principal uso é controlar o fornecimento de potência a um dispositivo elétrico. Para este uso, sua principal vantagem é a baixa perda de potência durante a transmissão.

O valor da tensão ou corrente a alimentar a carga é controlado por um pulso que varia de 0 (não condução) a 1 (condução) ao longo do tempo. A razão entre o tempo em que o pulso está conduzindo (valor 1) e o tempo total do pulso é denominada ciclo de trabalho (duty cicle, em inglês). Esta razão determina a tensão, ou corrente, média e define a potência entregue à carga.

Neste trabalho, foi usado um PWM junto ao sinal de controle, para a modulação do acionamento do relé. O período do PWM foi ajustado empiricamente, de forma a obter o melhor desempenho possível dos controladores usados.

3 MATERIAIS

3.1 SENSOR DE TEMPERATURA DS18B20

O DS18B20(figura 3.1) é um sensor de temperatura digital fabricado inicialmente pela empresa *Dallas Semiconductor* e utiliza a interface 1-Wire para comunicar-se com o microcontrolador. A interface 1-Wire utiliza apenas um pino para comunicação bidirecional suportando mais de um sensor no mesmo pino de comunicação o que é uma grande vantagem quando se trabalha com diversos sensores em um projeto. O sensor pode ser alimentado através da linha de dados o que o torna ideal em ambientes onde fontes de alimentação são limitadas[18].

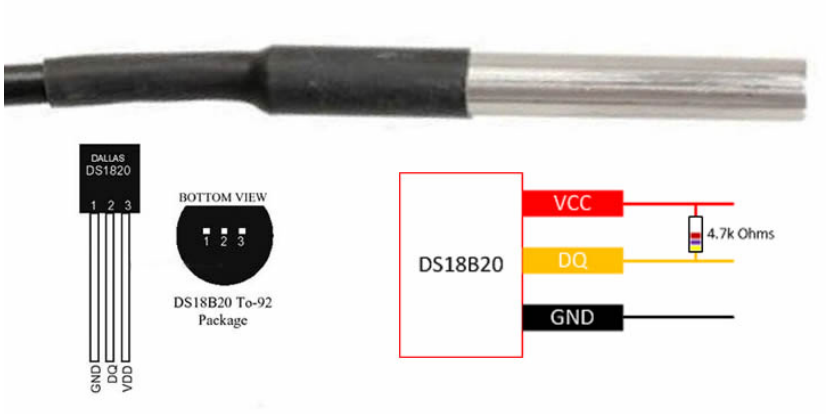

Figura 3.1 – Sensor de Temperatura DS18B20

Os dados de medida são transferidos como valores de até 12-bits (*unsigned interge*). Esses valores são linearizados e compensados em relação aos efeitos de temperatura e tensão de alimentação. A figura 3.2 ilustra o esquema de ligação típica.

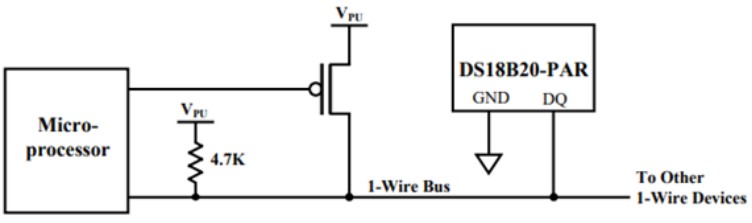

Figura 3.2 – Típico circuito de aplicação, incluindo o resistor de pull-up.

3.1.1 Especificações Técnicas

A tabela 3.1 mostra as especificações de temperatura do sensor:

Tabela 3.1 – Especificações de Temperatura para o DS18B20.

Parâmetros	**Condições**	**Valor**	**Unidades**
Acurácia	Típicas	±0,5	ºC
Resolução	12 bits	0,0625	ºC
Faixa de operação	-	-55 a 100	ºC
Tempo de Resposta	12 bits	750	ms
Afastamento a longo prazo	Típica	0,03	ºC/ano

A resolução, acurácia e facilidade de conexão tornam o sensor apropriado para a captação de dados no projeto.

3.2 SENSOR DE TEMPERATURA E UMIDADE DHT22

O DHT22 possui um sensor de umidade capacitiva e um termistor que, juntos, geram um sinal digital no pino de saída de dados.

As especificações do sensor podem ser observadas na tabela 3.2.

Tabela 3.2 – Especificações do sensor DHT22.

Tensão de alimentação [V]		3,3 - 6
Sinal de Saída		Digital
Faixa de Operação	**Umidade [RH]**	0 – 100%
	Temperatura [°C]	-40 até 80
Acurácia	**Umidade [RH]**	± 2%
	Temperatura [°C]	±0,5
Período Médio		2 s

Pode-se ver que a sua tensão de alimentação, possibilita que ele seja utilizado e alimentado pelo NodeMCU. Outro fator determinante para sua escolha é a sua boa distância de transmissão de dados que atinge cerca de 20 metros. Uma restrição é o tempo de amostra de dados, essa especificação limita o período de amostragem de dados a tempos superiores à 2 segundos[19], o que determina o tempo de amostragem mínimo.

O sensor já vem calibrado de fábrica e não necessita de calibração posterior. Ele teve sua temperatura compensada e calibrada em uma câmara de calibração e os coeficientes salvos em uma memória OTP (one-time programmable). Sendo assim, toda vez que o sensor é utilizado, ele carrega os coeficientes da memória e ajusta os dados. Além disso utiliza uma comunicação simples de um único fio.

As ligações do sensor podem ser vistas na figura 3.3. Recomenda-se que o pino 3 não seja conectado:

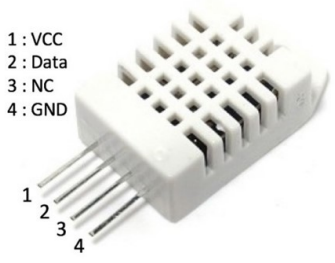

Figura 3.3 – Ligações do sensor DHT22.

3.3 NODEMCU

O NodeMCU(figura 3.4) é uma plataforma *open source* da família ESP8266 criado para ser utilizado no desenvolvimento de projetos de Internet of Things (IoT).

Esta plataforma é composta basicamente por um chip controlador (ESP8266 ESP-12E), uma porta micro USB para alimentação e programação, conversor USB serial integrado e já possui WiFi nativo.

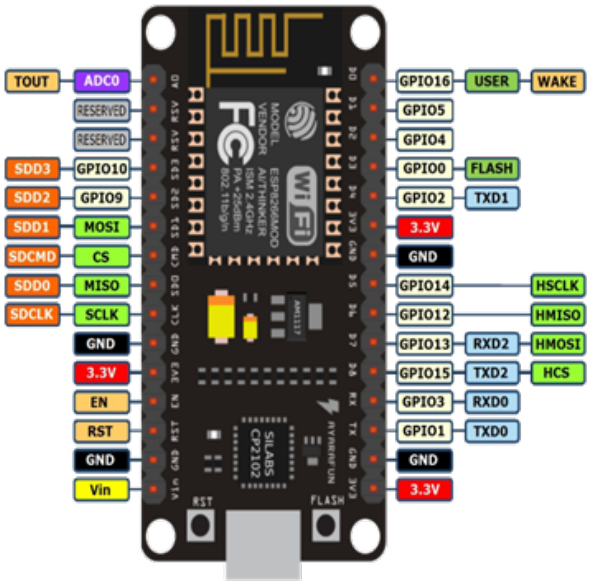

Figura 3.4 – NodeMCU e sua pinagem.

Uma das grandes vantagens em utilizar plataformas baseadas no ESP8266, é a possibilidade de se programar utilizando a IDE do Arduino. Assim como em outras placas da família ESP8266, o NodeMCU também é compatível com o ambiente de desenvolvimento do Arduino. Além disso, a placa pode ser programada utilizando a linguagem LUA (linguagem desenvolvida por brasileiros).

Ao ser utilizado a IDE do Arduino para programar o NodeMCU, será possível fazer o uso de diversas bibliotecas que já fazem grande parte da programação do Arduino. As principais características do NodeMCU são:

- Processador ESP8266-12E;
- Arquitetura RISC de 32 bits;
- 80/160 MHz;
- 4Mb de memória flash;
- 64Kb para instruções;

- 96 kb para dados;
- WiFi nativo padrão 802.11 b/g/n;
- Alimentação de 5 VDC pela porta micro USB;
- Nível lógico 3.3 V;
- 1 pino analógico com resolução de 10 bits;
- Pinos digitais com interrupções, PWM, I_2C e 1-wire(exceto o pino D0).

3.4 RELÉ DE ESTADO SÓLIDO - T2405Z-M

Para fazer acionamento de dispositivos, é comum usar uma chave estática. Elas promovem o chaveamento da potência fornecida para a carga, mas não a modifica em nenhum outro aspecto. O uso de relés é algo corriqueiro neste sentido.

Comumente, existem dois tipos de relés: os eletromecânicos e os de estado sólido. Estes são uma alternativa moderna para o uso de relés eletromecânicos, uma vez que são mais robustos quanto a ruídos e problemas mecânicos. Vale lembrar que o uso de relés de estado sólido é um pouco restrito, pois, não é toda aplicação que é possível usá-los.

O SSR T2405Z-M (figura 3.5) da Teletrom é um relé de estado sólido, cuja tensão de comando pode variar de 3 V a 32 V DC. A tensão de carga é de 240 V AC, com corrente de saída de até 5 A.

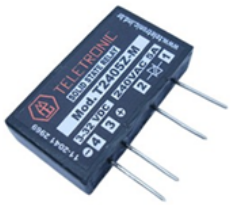

Figura 3.5 – Relé T2405Z-M (teletronic.ind.br).

Em um SSR, tem-se um dispositivo semicondutor que ao ser excitado, emite radiação infravermelha. Essa radiação atua sobre o receptor do dispositivo, o qual controla um circuito externo, através de um dispositivo de potência como, por exemplo, um transistor de potência, um SCR, um TRIAC, etc. Para mais detalhes,veja a figura 3.6.

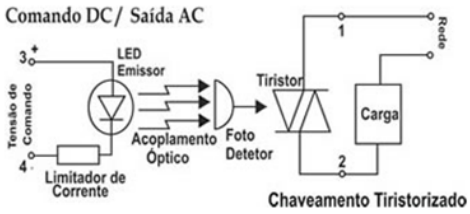

Figura 3.6 – Circuito interno ao relé T2405Z-M (teletronic.ind.br).

3.5 RASPBERRY PI 4B

O Raspberry Pi é uma série de embarcados desenvolvidos pela Fundação Raspberry Pi e baseados em System on a Chip (SoC). Seu principal objetivo é difundir e promover o estudo de Ciências da Computação e Informática em escolas. Hoje em dia, o Raspberry Pi uma tecnologia extremamente utilizada devido ao seu tamanho reduzido e ao seu baixo custo. É um computador embarcado muito difundido para a pesquisa e o desenvolvimento de projetos na área de automação residencial e predial.

Para o trabalho em questão, foi utilizado o Raspberry Pi 4 Model B mostrado na figura 3.7.

Figura 3.7 – Raspberry Pi 4 model B.

Suas especificações gerais são[20]:

- Processador: Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit 1.5GHz;
- Alimentação: Socket USB tipo C 5V 2,5A;
- 4 GB DDR4 de memória RAM;
- Duas portas USB 3.0 e Duas portas USB 2.0
- 15 pinos MIPI interface serial da câmera (CSI 2);

- Duas saídas micro HDMI;
- Porta micro SD para carregar o Sistema Operacional (SO) e salvar os dados;
- Dimensões: 85 x 56 x 17 mm;
- Raspberry Pi standard 40 pin GPIO header.

Para esse projeto, o sistema operacional utilizado no Raspberry Pi foi o Raspbian versão 5.4.51. O Raspbian é um sistema operacional gratuito baseado no Debian e otimizado para a arquitetura de hardware do Raspberry Pi.

3.6 MATLAB

O software MATLAB(3.8), que vem de Matrix Laboratory, consiste em um programa amplamente utilizado para operações numéricas, modelagens e simulações de sistemas desenvolvido pela Mathworks. Uma das principais ferramentas de modelagem presente no software é o Simulink. Este é amplamente utilizado para a simulação e modelagem de sistemas dinâmicos.

Devido às diversas ferramentas, de comunicação e controle, já implementadas no Simulink esse foi utilizado para a simulação e implementação da malha de controle.

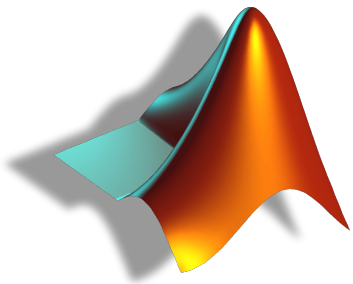

Figura 3.8 – Logo Matlab

3.7 THINGSPEAK

ThingSpeak é uma plataforma open source para logging e análise de aplicações de IoT. ThingSpeak se comunica usando o protocolo HTTP na internet ou rede local. A plataforma possui suporte integrado com o MATLAB, permitindo a visualização e análise dos dados carregados pelos dispositivos conectados ao sistema diretamente no software da Mathworks.

Pode se utilizar a plataforma gratuitamente com a limitação do número de dados enviados e com um intervalo de envio mínimo de 15 segundos. No seu modo pago é permitido

um intervalo mínimo de 1 segundo e o número de mensagens enviadas aumenta significadamente.

Um esquema de conexão genérica pode ser visto na figura 3.9.

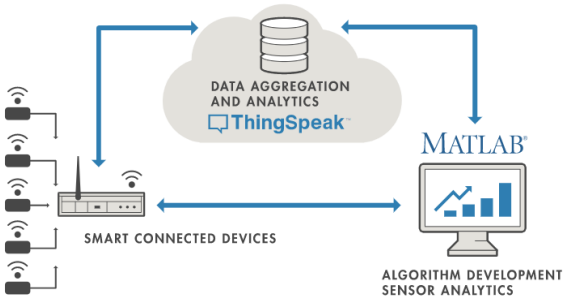

Figura 3.9 – Esquema de ligação genérica.

4 PROCEDIMENTOS

4.1 VISÃO GERAL DO SISTEMA

O sistema projetado pode ser dividido em três módulos:

- Aquisição de dados e atuação no sistema;
- Controle;
- Comunicação;

A aquisição de dados é realizada por meio do sensor de temperatura DS18B20 que coleta os dados de temperatura na sala de reuniões localizada no LARA como mostrado na seção 2.1, para o ambiente externo ao laboratório e para a sala vizinha foram utilizados sensores do tipo DHT22.O NodeMCU coleta esses dados e os envia para o servidor na nuvem(ThingSpeak) através da Internet.O Raspberry Pi coleta o número de pessoas no ambiente e o transmite para a plataforma na nuvem.

A posição dos sensores de temperatura externos e da câmera de vídeo é mostrada na figura 4.1:

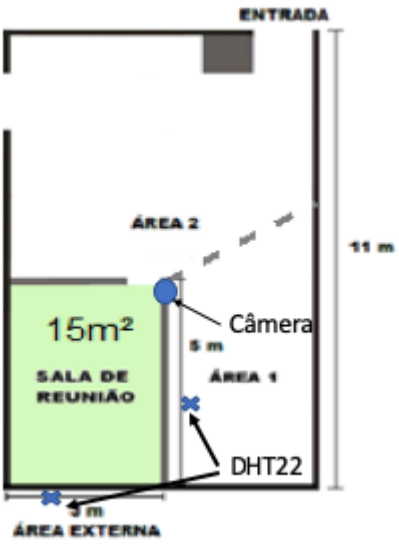

Figura 4.1 – Posição dos sensores para as medidas das perturbações

Os dados coletados foram os seguintes:

- **Temperatura da sala de reuniões**:variável controlada;

- **Temperatura externa ao laboratório**:componente da perturbação ao sistema;
- **Temperatura da sala externa à sala de reuniões**:componente da perturbação ao sistema;
- **Número de pessoas presentes na sala**:detecção da carga térmica humana;

A comunicação entre os módulos e a central de controle (Computador Pessoal) é feita utilizando-se do protocolo MQTT através da plataforma ThingSpeak.os dados são coletados e calculados e transmitidos para a ThingSpeak onde o computador executando o MATLAB os analisa.

O MATLAB coleta esses dados e os utiliza para calcular o sinal de controle para atuar no sistema e envia esse sinal (já modulado em PWM) para a nuvem e então o módulo NodeMCU, responsável pelo acionamento do ar-condicionado, coleta-o.

O sinal de controle é transmitido para o NodeMCU que por meio do relê de estado sólido que aciona o circuito de acionamento do compressor.

O esquemático da implementação pode ser visto na figura 4.2 :

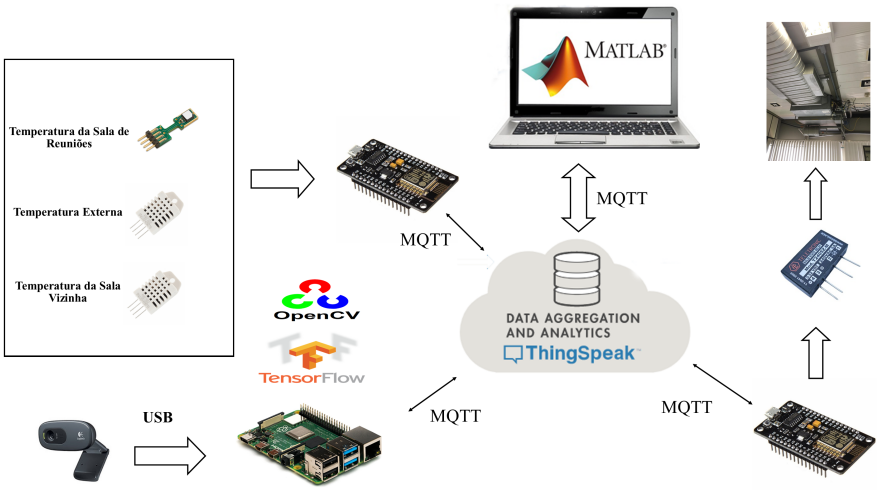

Figura 4.2 – Visão geral do sistema.

4.2 IDENTIFICAÇÃO DO SISTEMA

Para a sintonia dos controladores é preciso realizar a identificação do sistema.A identificação do sistema foi feita utilizando duas estratégias.

- **Identificação por batelada:** utilizando a *toolbox* de identificação de sistemas do Matlab já que essa possuí uma grande variedade de algoritmos.entre eles o algoritmo de

identificação por subespaço mostrado na subseção 2.2.3. A interface da *toolbox* pode ser vista na figura 4.3. Este modelo foi utilizado para a sintonia dos controladores PI e MPC.

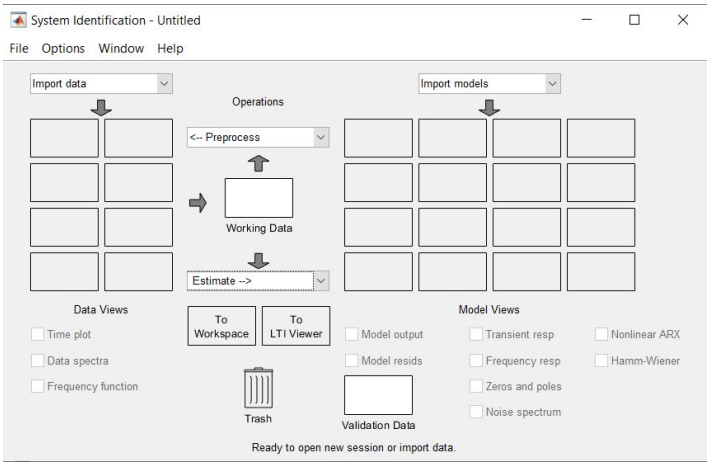

Figura 4.3 – Interface da toolbox de identificação de sistemas

- **Identificação recursiva:** feita utilizando o algoritmo de MQ com fator de esquecimento descrito na subseção 2.2.4 e implementado no código no Anexo III para o controlador adaptativo que necessita de identificação online.

4.3 DETERMINAÇÃO DA CARGA TÉRMICA

4.3.1 Detecção da Carga Térmica Humana

A detecção da carga térmica devido à presença humana foi feita através da contagem do número de pessoas presentes no ambiente de estudo.

A detecção de pessoas é feita a partir de uma câmera conectada ao modulo Raspberry PI 4 utilizando uma CNN com o modelo treinado com o algoritmo SSD(Single Shot Detection) juntamente com as bibliotecas do OpenCV e TensorFlow.

O *frame* capturado pela câmera é analisado pelo modelo treinado. Onde é feita a detecção de todos os objetos presentes no ambiente. O algoritmo implementado separa todos os objetos detectados como "pessoas" com uma certeza maior do que 50% é os adiciona na contagem.

O código implementado pode ser visto no Anexo IV.

Considerando que a contribuição de cada pessoa pode ser vista como o calor sensível radiado por cada pessoa e este, na média e para o ambiente estudado, de acordo com [21] em sua tabela 12 é dado como 71 kcal/h (82,53 W).A pertubação devido à carga térmica

proveniente da presença das pessoas foi definida no esquemático na figura 4.4.

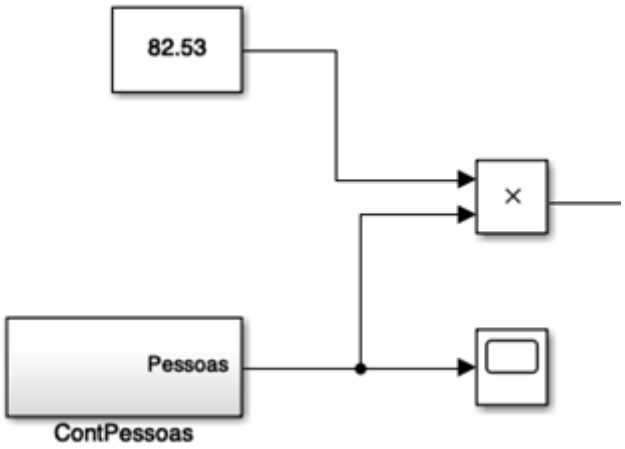

Figura 4.4 – Perturbação devido a presença das pessoas.

4.3.2 Detecção da Carga térmica devido a diferença de temperatura dos ambientes vizinhos

Para medir a perturbação devido aos ambientes vizinhos utilizou-se de sensores de temperatura DHT22 na área externa ao laboratório e na sala vizinha (figura 4.1). A transferência de temperaturas entre os ambientes vizinhos é descrita por (2.66).

A função de transferência de temperatura do ambiente externo para a sala $G_e(s)$ é dada por [3]:

$$G_e(s) = \frac{1}{220s + 1} \tag{4.1}$$

Já a função de transferência de temperatura entre a sala vizinha e a sala de reuniões $G_v(s)$ foi identificada[3] como:

$$G_v(s) = \frac{1}{321,3s + 1} \tag{4.2}$$

A contribuição térmica dos ambientes pode ser vista na figura 4.5.

Em [3] realizou-se a identificação dos parâmetros K_e e K_v encontrando como valores 0,035 e 0,095 respectivamente.

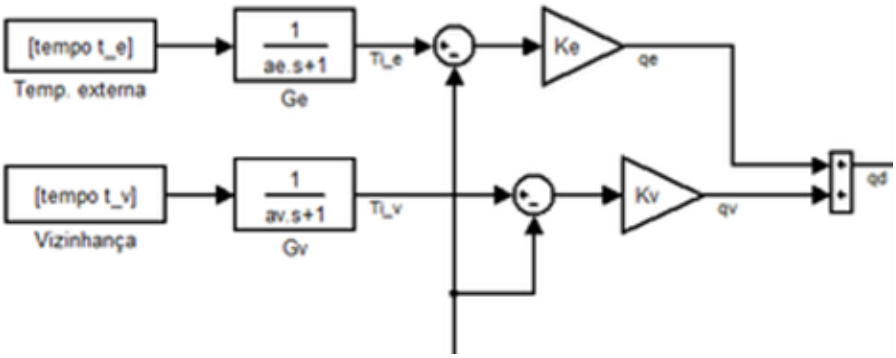

Figura 4.5 – Contribuições térmicas dos ambientes vizinhos.

4.4 SINTONIA DOS CONTROLADORES

4.4.1 Controlador Liga-desliga

O controlador liga-desliga é simples e rápido e leva em conta o erro entre a referência e a temperatura medida.A lei de controle é determinada por:

$$\begin{cases} 0 \; se \; y_{ref} - y < -0,2 \\ 1 \; se \; y_{ref} - y > 0,2 \end{cases} \tag{4.3}$$

Em que, 1 equivale a ligar o atuador, e 0,2 desligá-lo.O valor de 0 equivale à histerese do respectivos atuador.

4.4.2 Controlador PI

Foi utilizada a *toolbox*(figura 4.6) de sintonia de controladores PI do Matlab para realizar a sintonia deste controlador.

A sintonia foi feita utilizando o método CHR descrito na subseção 2.4.2.1. Os parâmetros para a sintonia do controlador são encontrados na tabela 2.1.

O fator integrativo diminui o erro em regime permanente e não é tão sensível a ruídos – ao contrário da parcela derivativa, cuja função é tornar o controle mais rápido. Para um processo térmico, o tempo para chegar ao valor de referência não é tão crítico, pois a sua dinâmica é lenta. A amplitude do sobressinal, porém, é um ponto relevante do projeto, uma vez que, além da preocupação com saturação, prejudicaria o conforto térmico. Com isso, e considerando o que foi dito no parágrafo anterior, escolhemos fazer um controlador PI, com *overshoot* de 0%.A estrutura do controlador é mostrada em (4.4).

$$C_{PI}(z) = -0,01439\frac{1+72z}{z} \tag{4.4}$$

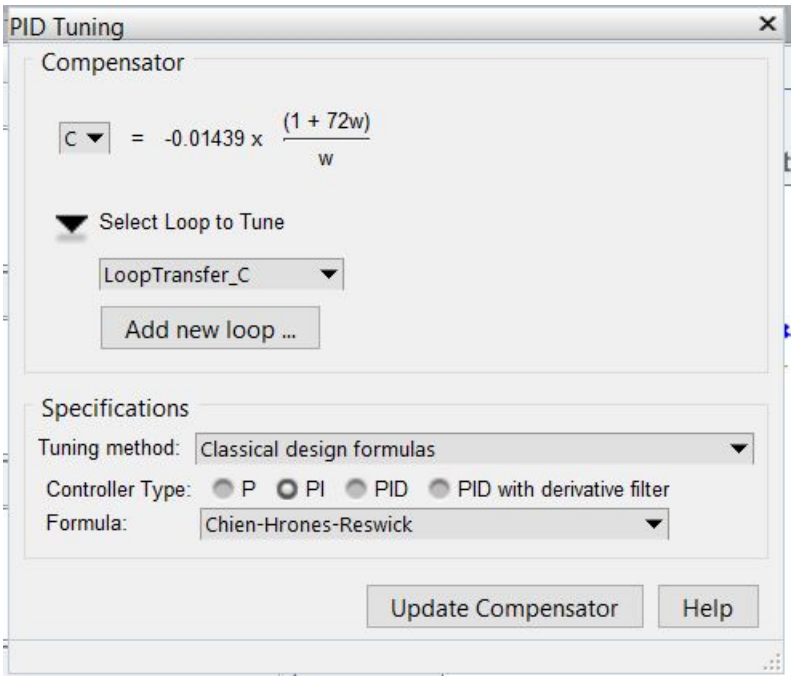

Figura 4.6 – Toolbox de sintonia de controladores PID.

4.4.3 Controlador Adaptativo

As especificações definidas para os controladores clássicos foram as mesmas para o controlador adaptativo, isto é, sobressinal de 0%, com pouca variação em torno da referência.

Embora existam condições para definir os polinômios R, S e T, não há um algoritmo específico para encontrá-los. Os autores em [13] mostram certas escolhas que facilitam o design do controlador e, por isso, são mais usuais. As definições de alguns coeficientes e do grau dos polinômios envolvidos foram feitas empiricamente, considerando o domínio do tempo discreto. Os parâmetros dos polinômios em questão foram obtidos por meio de simulações, sempre buscando a melhor resposta do sistema.

De acordo com [22], uma função de segunda ordem com canal integral é uma boa representação para um modelo de referência. Desta forma, temos que:

$$R = b_0 z \tag{4.5}$$

$$T = b_{m0} \tag{4.6}$$

$$S = a_0 - a_{m0} \tag{4.7}$$

Em que a_0 e b_0 são parâmetros da planta, e a_m0 e b_m0 são parâmetros do modelo de referência.

Por sua vez, o modelo de referência é dado por:

$$G_m(z) = \frac{b_{m0}z + b_{m1}}{z^2 + a_{m1}z + a_{m2}} = \frac{0,00925z + 0,08738}{z^2 - 0,8998z + 0,07984} \tag{4.8}$$

A lei de controle é expressa pela equação (4.9), a seguir:

$$\begin{aligned} u(k) = K[&u(k+1) + \frac{b_1}{b_0}(a_{m0} - a_2)(y(k-2) - y(k-1)) \\ &- (a_{m2} - a)(y(k-1) - y(k)) \\ &+ (a_{m0} + a_{m1} + a_{m2})(y_{ref}(k) - y(k))] \end{aligned} \tag{4.9}$$

A inclusão do ganho K na equação (4.8) é uma adaptação da lei de controle, pois foi visto que, com esta inclusão, melhorava-se o transiente. O valor de escolhido empiricamente foi de $K = 33$

4.4.4 MPC

A sintonia do controlador MPC foi feita utilizando a toolbox MPC do Matlab®. A toolbox permite a inserção dos parâmetros como horizonte de predição, horizonte de controle, restrições e pesos.

Recomenda-se [17] que o horizonte de predição tenha entre 20 e 30 amostras e como o sistema possui uma taxa de amostragem de 2 segundos, o horizonte de predição foi definido como 60 segundos. O horizonte de controle é recomendado entre 10% e 20% do horizonte de predição logo foi definido como sendo 6 segundos.

O atuador é restrito entre 0 e 220 V portanto as restrições do sinal de controle são 0 para o valor mínimo e 220 para o valor máximo.

Em termos de performance foi escolhido que o sistema deverá oscilar pouco logo será mais robusto e como a planta apresenta uma dinâmica lenta a estimação dos estados pode ser feita de maneira lenta.

A simulação do controlador e do sinal de controle podem ser vistos nas figuras 4.7 e 4.8.

O sistema completo no Simulink pode ser visto na figura 4.9.

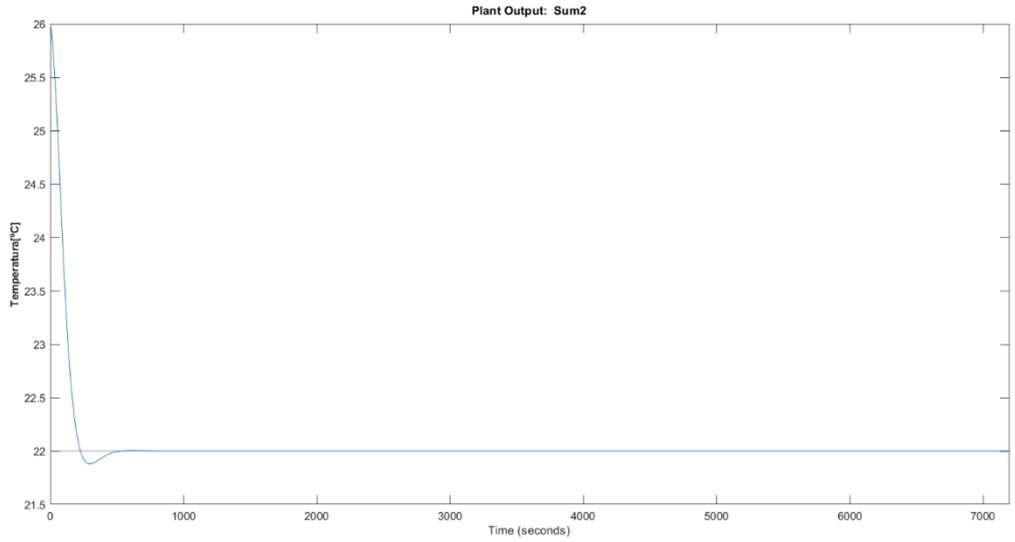

Figura 4.7 – Simulação MPC.

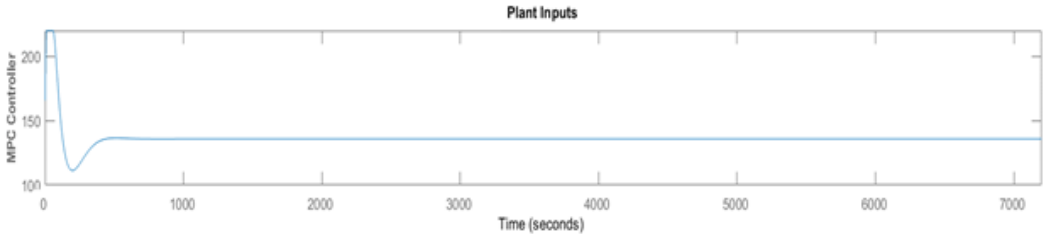

Figura 4.8 – Sinal de Controle Simulado.

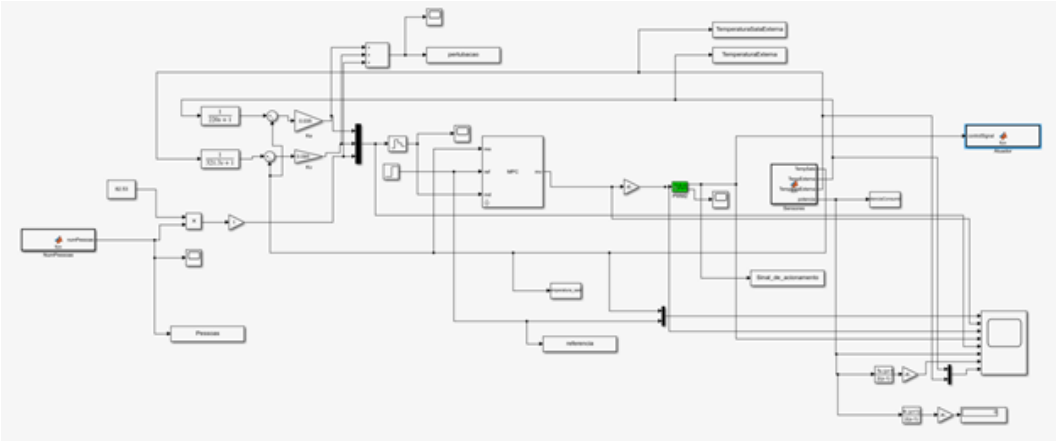

Figura 4.9 – Sistema completo no Simulink.

5 RESULTADOS

5.1 IDENTIFICAÇÃO DO SISTEMA

5.1.1 Identificação por Batelada

A técnica de estimação que obteve o melhor resultado quanto ao ajuste aos dados de validação foi a N4SID(subseção 2.2.3) com 90,27% de adequação como pode ser visto na figura 5.1 e na tabela 5.1.

Tabela 5.1 – Adequação aos dados de validação para os difentes algoritmos

Algoritmo	**Fit[%]**
N4SID	90,27
Arx	84,51
Função de Transferência	56,37
Impulso	49,04

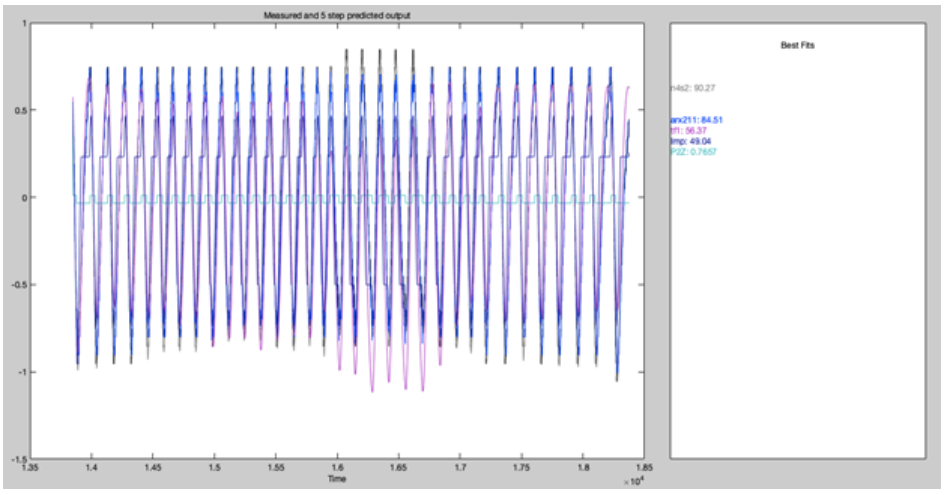

Figura 5.1 – Validação dos modelos estimados.

O modelo resultante possui no espaço de estados discreto a seguinte representação:

$$
\begin{aligned}
x(k+1) &= \begin{bmatrix} 0 & -0,9462 \\ 1 & 1,945 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u(k) \\
y(k) &= \begin{bmatrix} -0,0026 & -0,006 \end{bmatrix} x(k)
\end{aligned}
\tag{5.1}
$$

5.1.2 Identificação Recursiva

Utilizando de MQ com fator de esquecimento implementado como mostra o Anexo III nas seguintes condições:

A matriz de regressores é:

$$\psi(k) = \begin{bmatrix} y(k-1) & y(k-2) & u(k-5) & u(k-4) \end{bmatrix} \tag{5.2}$$

O vetor de parâmetros definido como

$$\theta(k) = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 & \theta_4 \end{bmatrix}^T \tag{5.3}$$

e inicializado com todos os parâmetros $\theta = 0,1$ e assumindo o fator de esquecimento como $\lambda = 0,9502$.

Foram obtidos os seguites resultados:

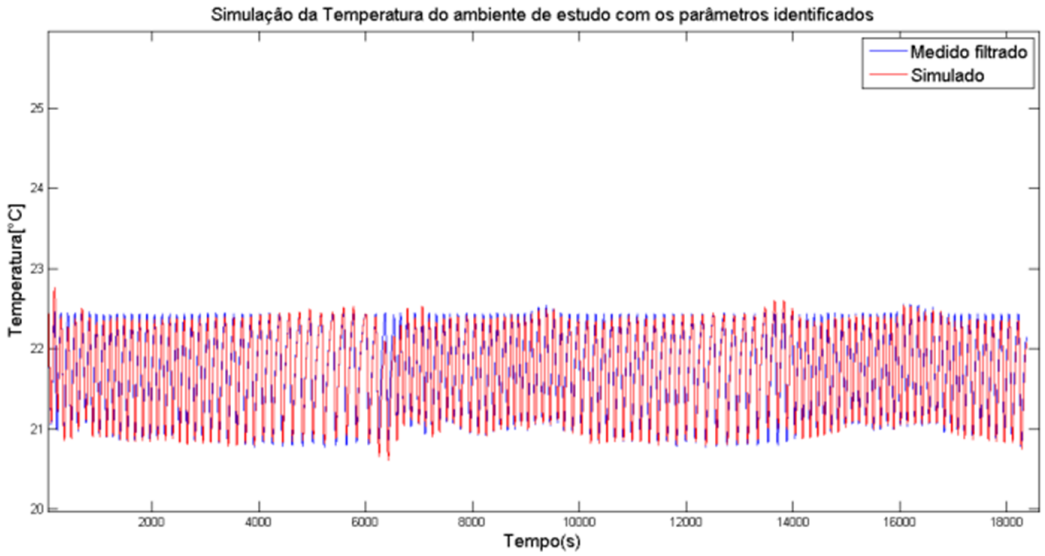

Figura 5.2 – Estimação da temperatura

Podemos ver pela figura 5.2, que a identificação foi satisfatória. O erro quadrático médio foi de 0.0156. Os parâmetros da identificação, de acordo com a figura 5.3, variam bastante quando não filtrados. Reitera-se, portanto, a necessidade de submeter a identificação a um filtro passa-baixa Butterworth de segunda ordem, com frequência de corte de 1Hz.

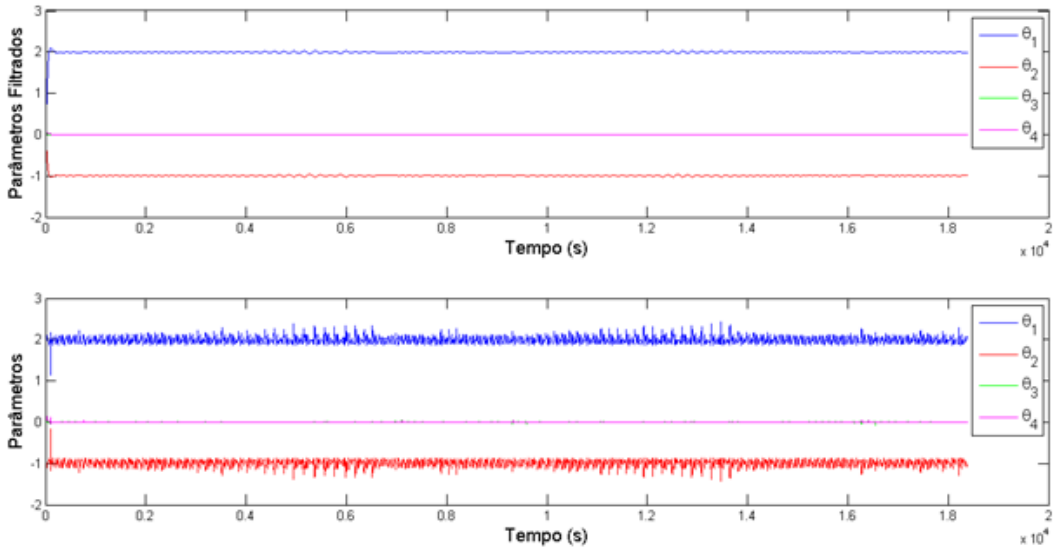

Figura 5.3 – Estimação dos parâmetros

5.2 DETECÇÃO DE PESSOAS PARA A DETERMINAÇÃO DA CARGA TÉRMICA

om a instalação do Raspberry Pi no ambiente foi possível, utilizando o algoritmo apresentado no Anexo IV, a detecção de pessoas e assim estimar a carga térmica presente.

O resultado para o ambiente vazio pode ser observado na figura 5.4. O espalhamento de diferentes objetos, com cadeiras e o tripé, não provocou a ocorrência de falsos positivos.

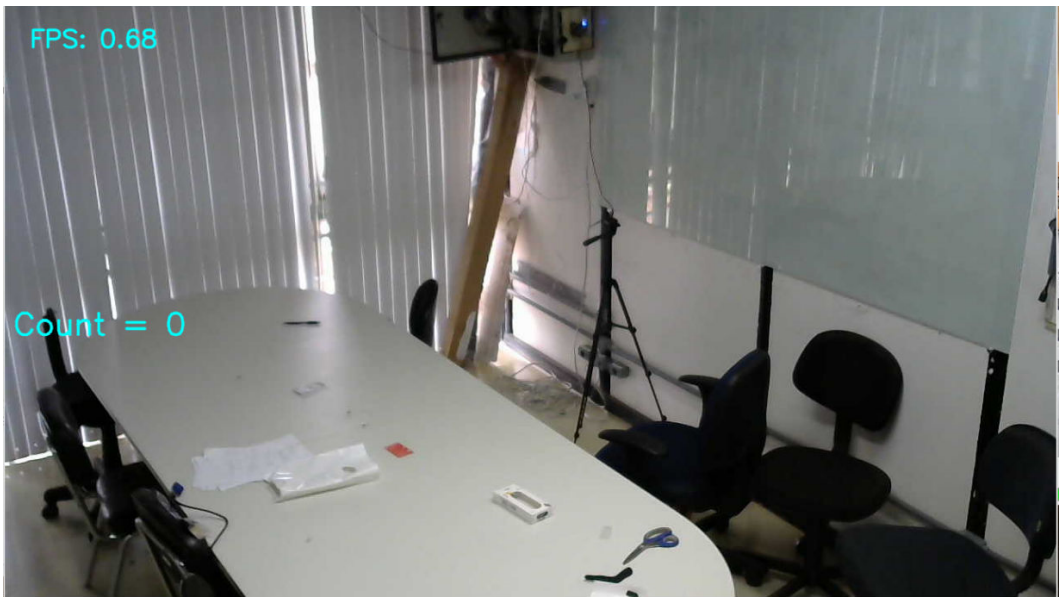

Figura 5.4 – Ambiente vazio.

Para o ambiente com apenas uma pessoa podemos observar o resultado na figura 5.5. É observado o *Score* (porcentagem de certeza da detecção) assim como a contagem. O sistema

montado apresentou pouca dificuldade com apenas uma pessoa o ambiente a não ser nos cantos do *frame* onde parte da pessoa fica de fora.

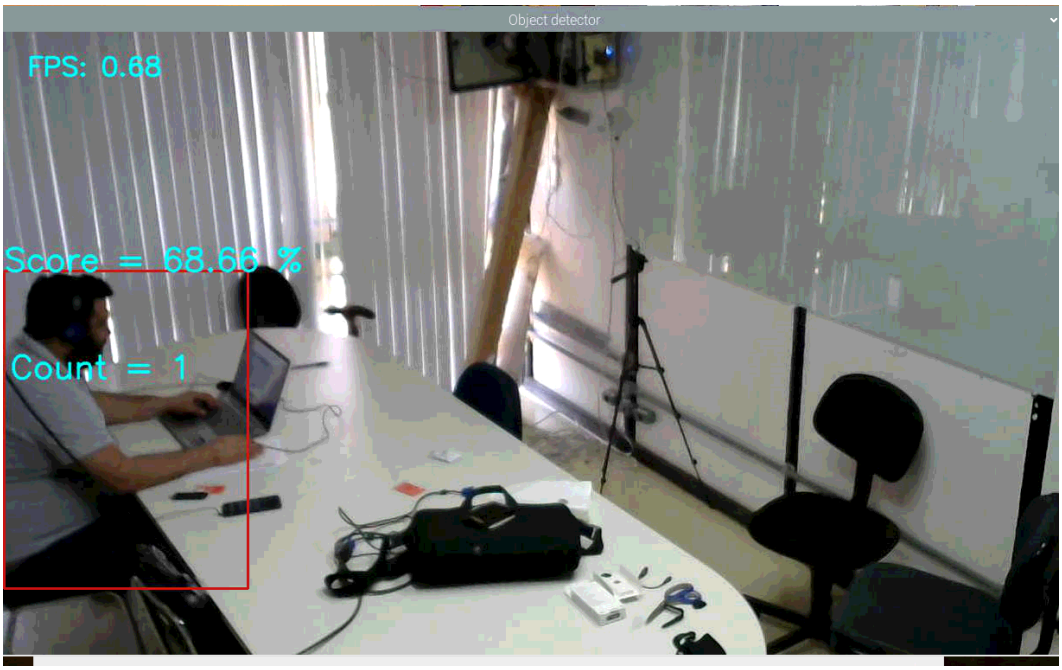

Figura 5.5 – Ambiente ocupado por uma pessoa.

Com a até duas pessoas verificou-se um resultado adequado tanto quanto à classificação como a contagem das pessoas. Mesmo com os indivíduos ocupando lugares próximos foi possível fazer a contagem como observado na figura 5.6.

Devido à pandemia de **COVID-19** (2020), não foi possível realizar testes com uma maior ocupação do ambiente de estudos.

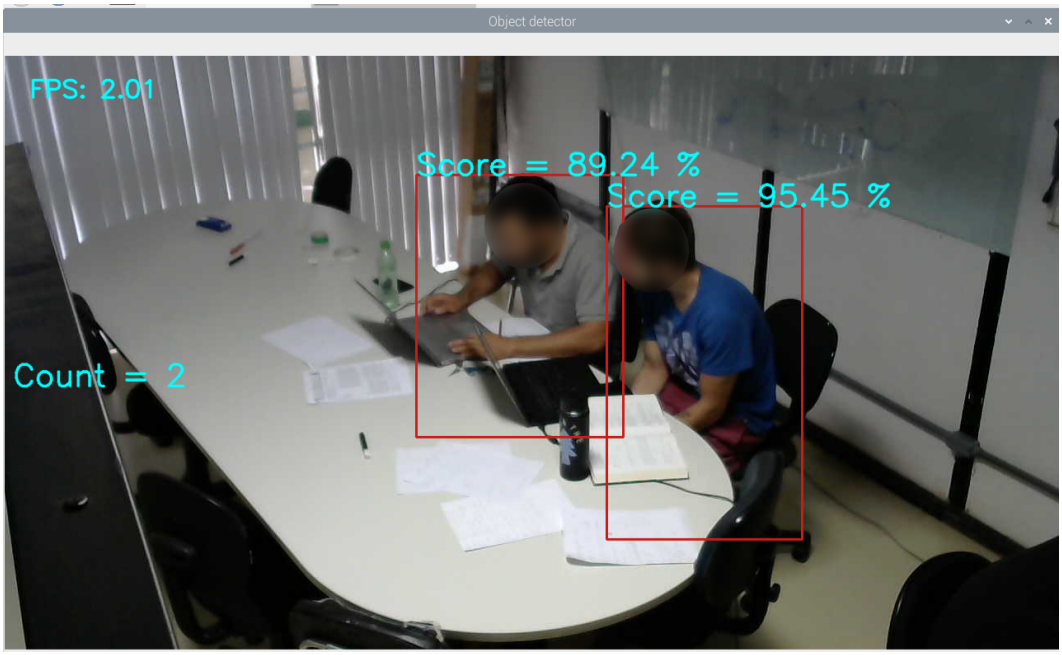

Figura 5.6 – Ambiente ocupado por duas pessoas.

5.3 PROBLEMAS EM RELAÇÃO À CONEXÃO DE REDE COM A INTERNET

Devido à instabilidade da conexão de internet do laboratório vários experimentos foram perdidos como mostrado na figura 5.7:

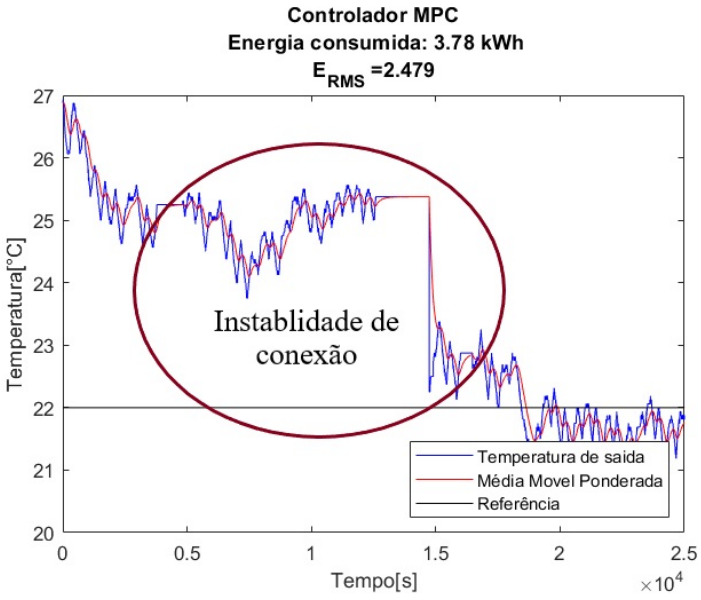

Figura 5.7 – experimentos com instabilidade de conexão.

Problemas de conexão com a Internet foram encontrados devido à instabilidade da conexão local(entre o roteador e os módulos microcontrolador) e também entre a o roteador e o servidor de Internet.

5.4 CONTROLADORES SEM PERTUBAÇÃO

Definimos perturbações como a presença de pessoas no ambiente e o fluxo de calor dos ambientes externos.

O primeiro resultado que tivemos foi a saída e o sinal de controle do controlador Liga-Desliga, representados pelas figuras 5.8 e 5.9, respectivamente.

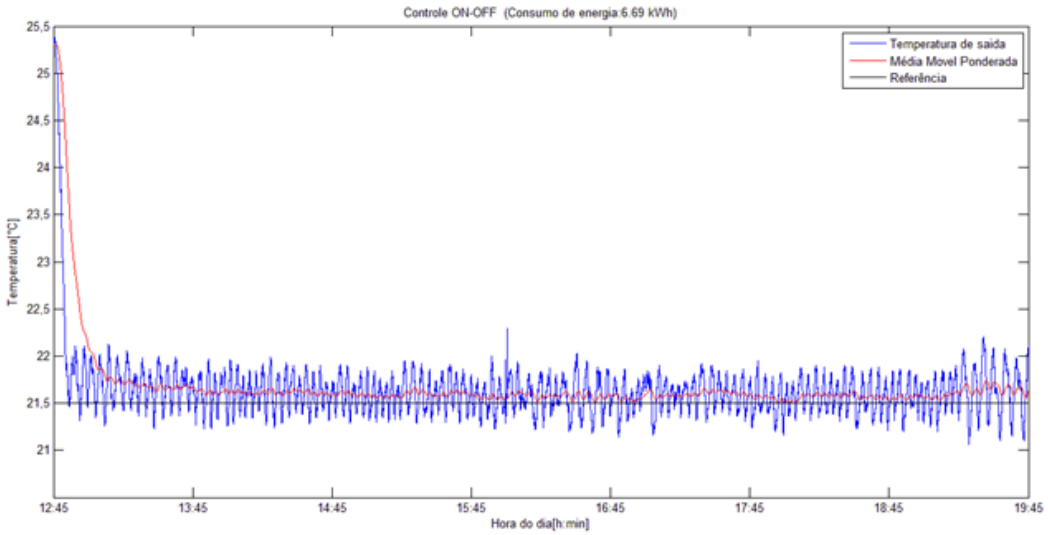

Figura 5.8 – Medição de temperatura usando controle Liga-Desliga com histerese

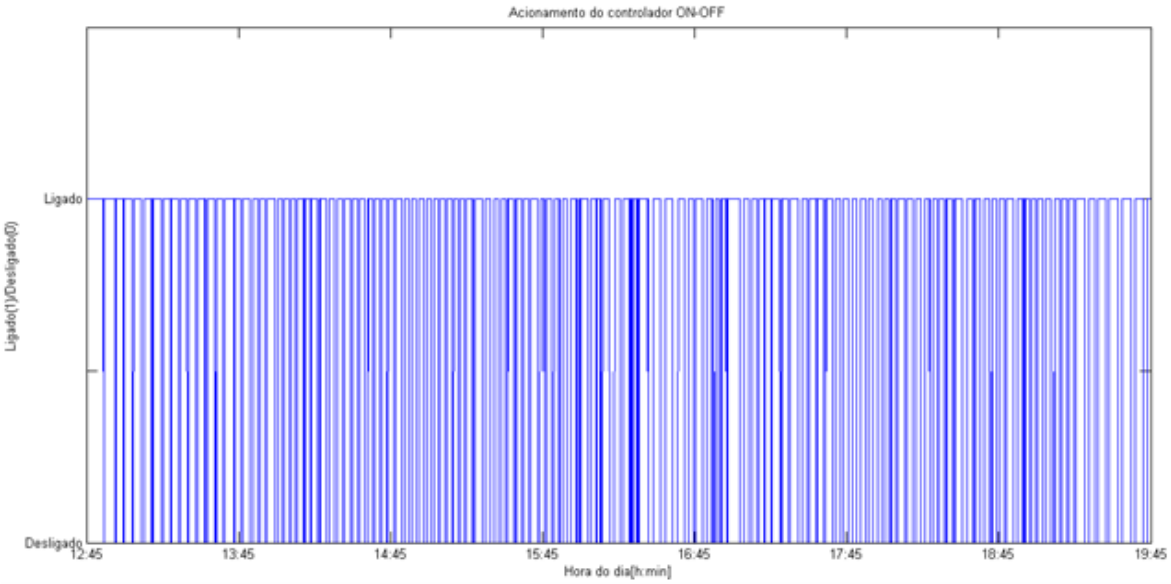

Figura 5.9 – Sinal de controle do controlador ON-OFF.

A figura 5.10 mostra o controle da temperatura quando o atuador está associado ao controlador PI na forma da equação (4.4). A figura 5.11 ilustra o sinal de controle deste controlador.

A seguir, tem-se o desempenho do controlador adaptativo onde sua estrutura se encontra

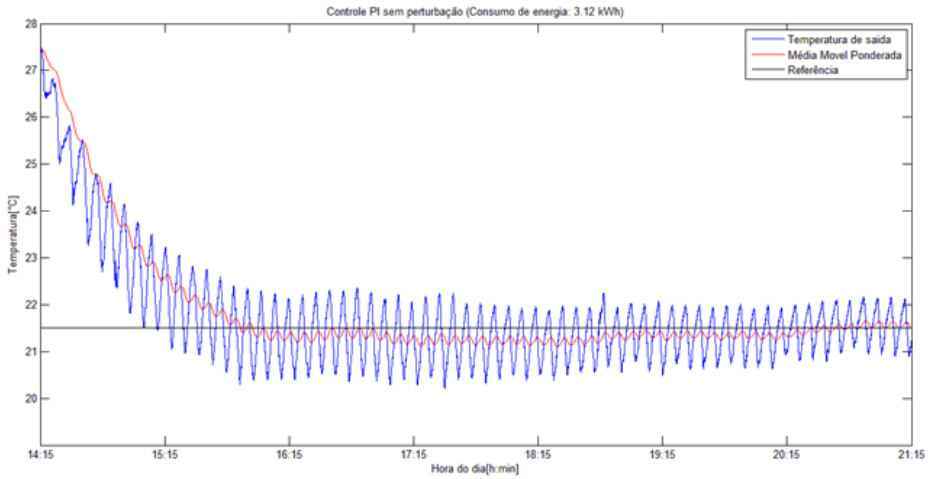

Figura 5.10 – Medição de temperatura usando controle PI.

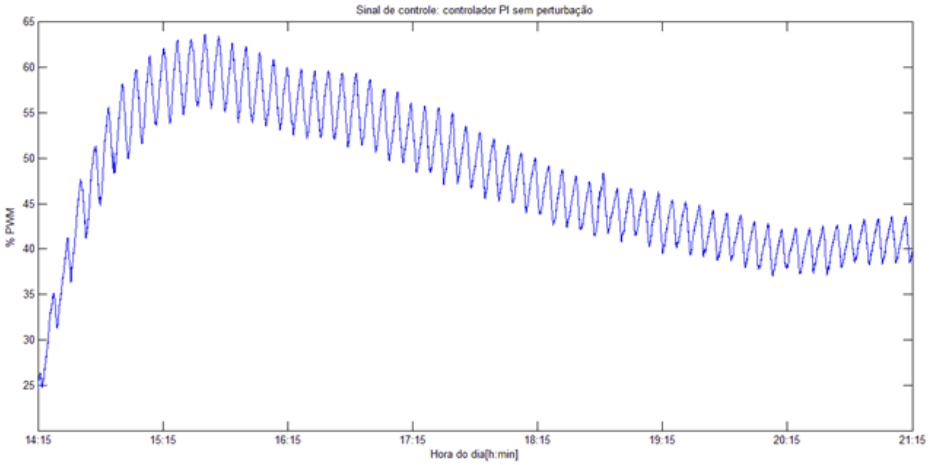

Figura 5.11 – Sinal de controle do controlador PI.

na equação 4.9. A figura 5.12 mostra a temperatura do ambiente controlada por um controlador RST. A figura 5.13 mostra o seu sinal de controle.

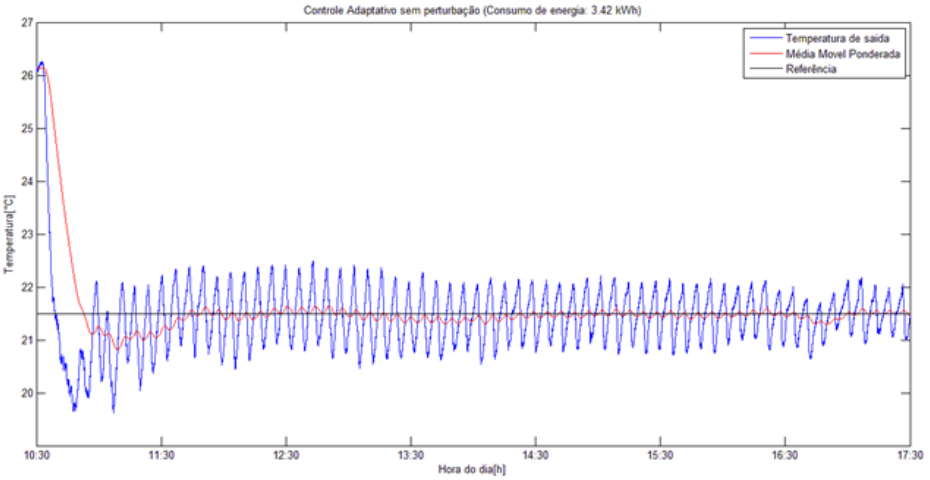

Figura 5.12 – Medição de temperatura usando controle RST

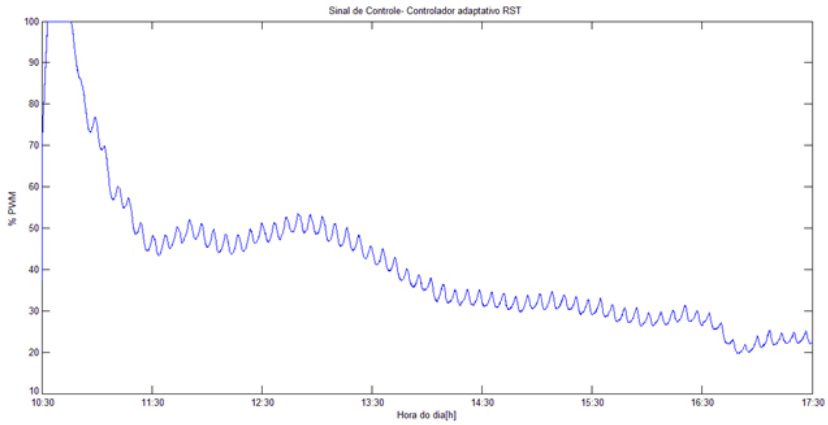

Figura 5.13 – Sinal de controle do controlador RST.

Para o controlador MPC, sintonizado na subseção 4.4.4, foi obtido o seguinte resultado:

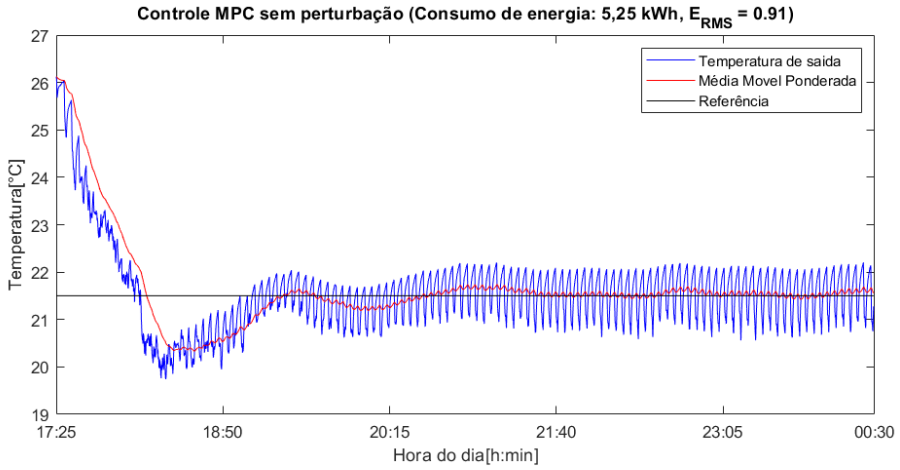

Figura 5.14 – Controle da temperatura usando o controlador MPC.

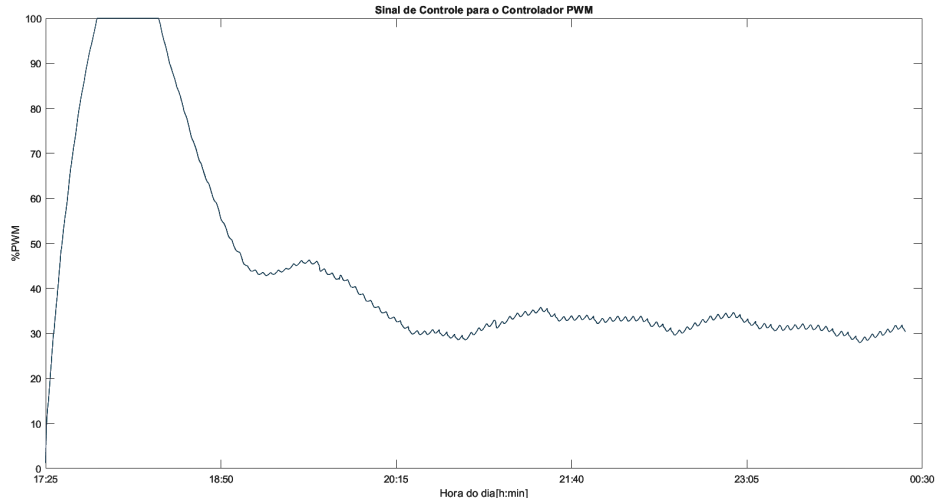

Figura 5.15 – Sinal de controle para o controlador MPC.

Com todos estes resultados, faremos a comparação de desempenho, levando em conta o tempo para atingir a referência e a exigência de acionamento.

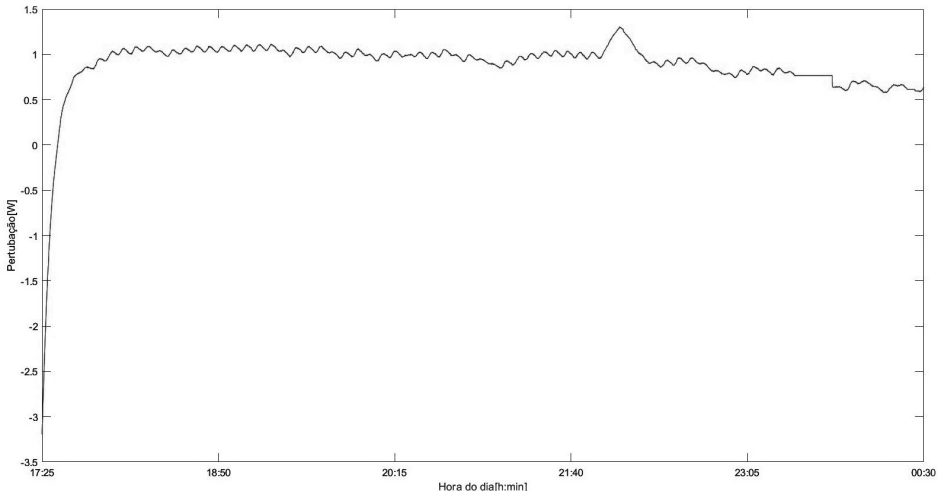

Figura 5.16 – Pertubação medida para o controlador MPC.

Vemos que ON-OFF é o mais rápido, de acordo com a Figura 5.8. Comparando-se com todos os controladores, no entanto, a diferença de tempo para alcançar a referência não é exatamente uma vantagem. Este controlador é o que mais gasta energia, como podemos ver pela Figura 5.9. O consumo de energia foi medido apartir do contador de energia presente no quadro de energia do ar-condicionado.

O controle original do ar-condicionado da sala era do tipo ON-OFF diferencial, pois baseavase apenas em um valor referencial. Desta forma, é possível dimensionar a melhora que os outros controladores adicionariam ao sistema. A equação (5.8) define essa melhoria em termos matemáticos, em que E_{ON-OFF} é a energia, em kWh, gasta pelo controlador original, E_{Novo} é a energia gasta pelo controlador novo(PI, RST, MPC) e M é o percentual de melhoria.

$$M = \frac{E_{ON-OFF} - E_{NOVO}}{E_{NOVO}} \times 100 \tag{5.4}$$

Devemos definir também o erro médio quadrático(E_{RMS}) da temperatura de saída em relação à referencia como sendo:

$$E_{RMS} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_{ref} - y_i)} \tag{5.5}$$

Onde,

N é o número de amostras;

y_{ref} é a referência, e;

y_i é a medida de número i na amostra.

Os Dados resultantes desses experimentos podem ser vistos na tabela 5.2.

Tabela 5.2 – Dados dos experimentos realizados sem carga térmica.

Controlador ON-OFF		**Controlador PI**		**Controlador Adaptativo**		**Controlador MPC**	
E_{RMS} *[°C]*	Consumo [kWh]	E_{RMS} *[°C]*	Consumo [kWh]	E_{RMS} *[°C]*	Consumo [kWh]	E_{RMS} *[°C]*	Consumo [kWh]
0,1399	*6,69*	*1,5524*	*3,12*	*0,5291*	*3,42*	0,91	5,25

Com esses dados e utilizando a equação (5.4) foi possível encontrar o percentual de melhoria dos controladores PI, RST e MPC em relação ao controlador padrão do sistema(LIGA-DESLIGA) como visto na tabela 5.3.

Tabela 5.3 – Relação entre tipos de controladores e melhoria em termos de energia.

Controlador	**Percentual de melhoria (em relação ao ON-OFF)**
PI	53,36%
RST	48,88%
MPC	27,43%

Vê-se, pela tabela 5.3, que o controlador PI, em relação ao MPC e ao RST, é o que menos gasta energia, principalmente no começo do acionamento, nos primeiros 15 minutos transcorridos, como é possível verificar pelas figuras 5.8, 5.10, 5.12 e 5.14. Ainda analisando estas figuras, após 30 minutos aproximadamente, o desempenho dos controladores MPC e RST, em termos de sinal de controle, passa a ser parecido. A diferença em termos de percentual de melhoria entre os dois é de 21,45%, o que mostra um desempenho superior do RST tanto em relação ao consumo de energia quanto ao tempo de resfriamento do ambiente.

5.5 CONTROLE COM CARGA TÉRMICA VARIÁVEL

Nesta etapa do projeto, expomos o ar-condicionado da sala a perturbações cotidianas, como pessoas em reunião, ou apenas estudando na sala, e a porta sendo aberta várias vezes. Esta é a configuração mais importante na definição de um bom controlador, que é quando o ambiente está em uso.

Para fazermos as devidas comparações, o experimento foi refeito para o controlador LIGA-DESLIGA, o controle original do ar-condicionado. As figuras 5.17 e 5.18 mostram, respectivamente, a sua saída e o seu sinal de controle.

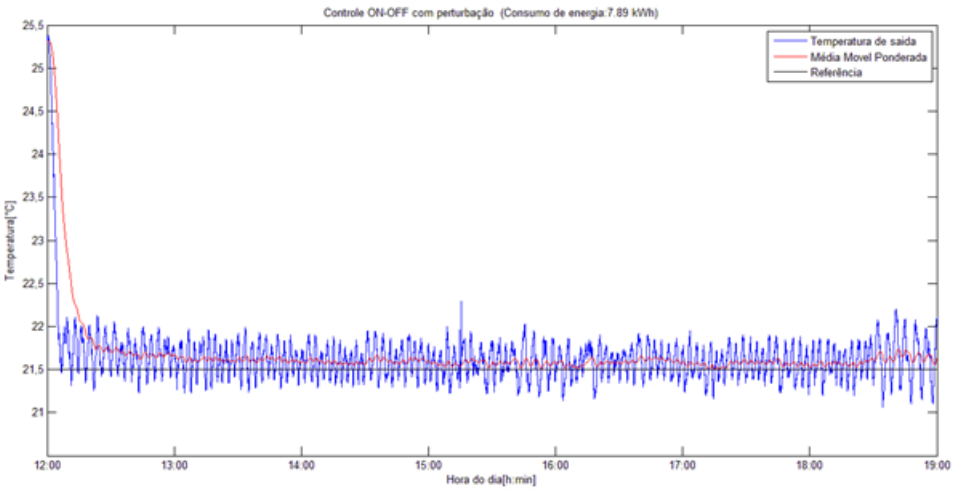

Figura 5.17 – Medição da temperatura usando controlador ON-OFF, com perturbações.

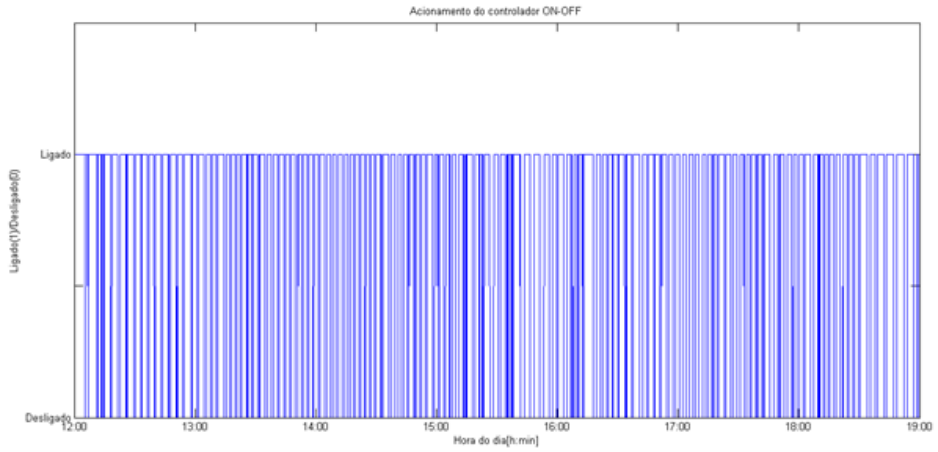

Figura 5.18 – Sinal de controle do controlador ON-OFF, com perturbações.

Para o controle PI, chegamos às figuras 5.19 e 5.20 , que ilustram a saída em termos de temperatura e o sinal de controle, respectivamente.

Para o controlador RST, podemos ver seu desempenho pelas figuras 5.21 e 5.22.

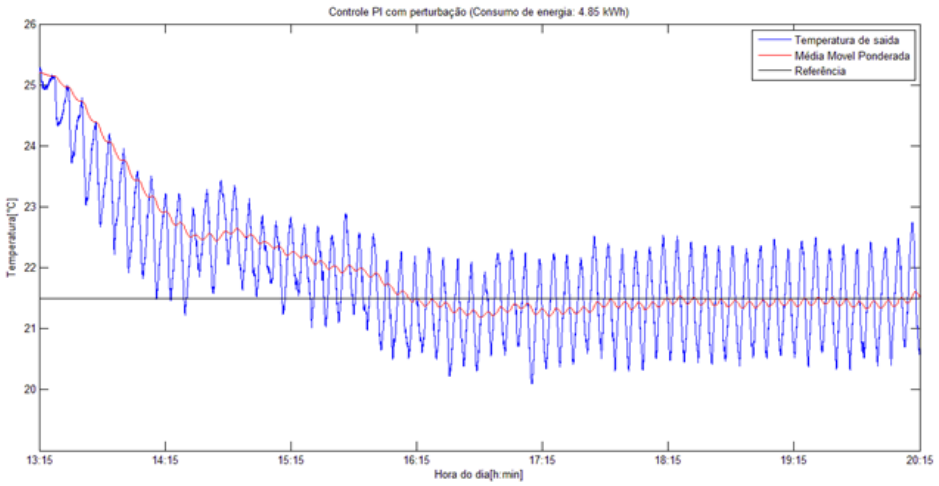

Figura 5.19 – Medição da temperatura usando controlador PI, com perturbações.

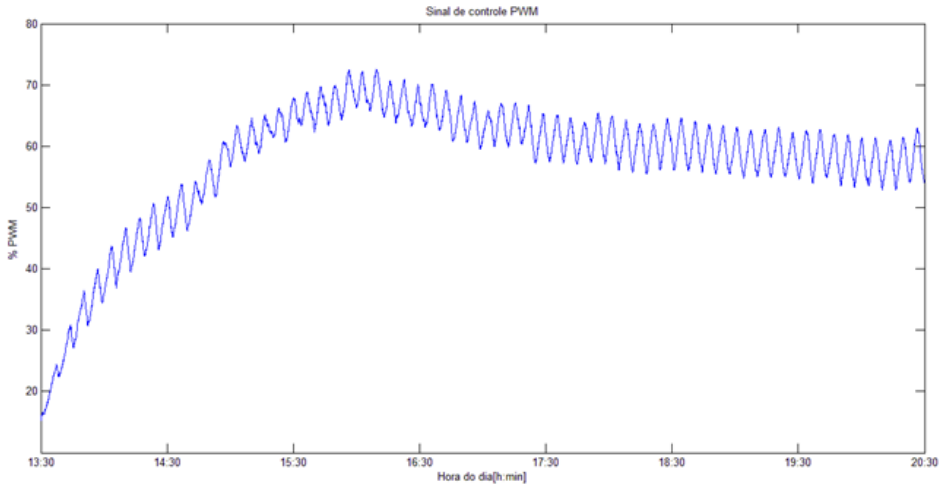

Figura 5.20 – Sinal de controle do controlador PI, com perturbações.

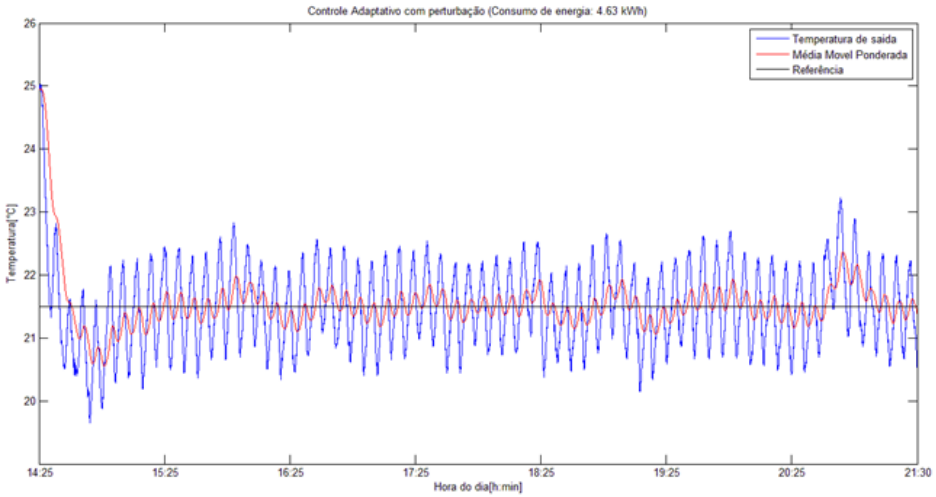

Figura 5.21 – Medição da temperatura usando controlador RST, com perturbações.

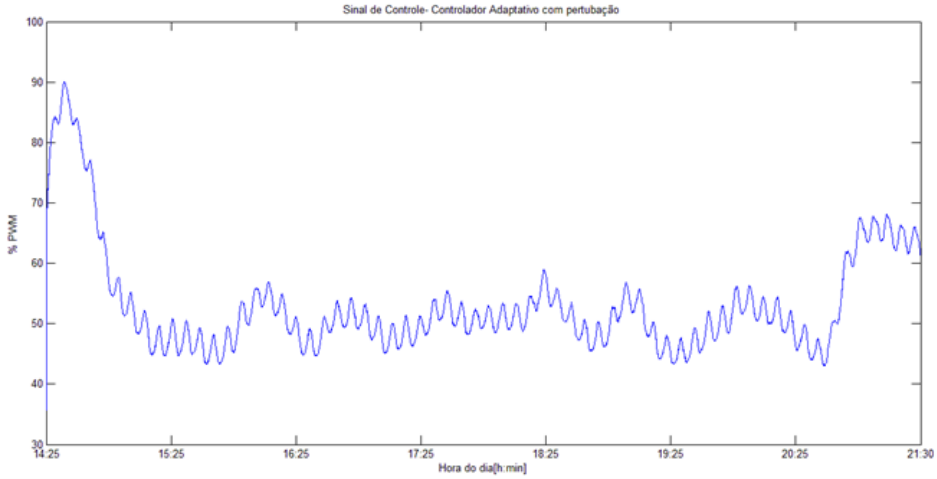

Figura 5.22 – Sinal de controle do controlador RST, com perturbações.

Devido à pandemia de COVID-19 (2020) não foi possível realizar experimentos com a presença de pessoas no ambiente.

Fazendo o mesmo cálculo percentual de melhoria – feito para o caso sem variação da carga térmica –, temos a tabela 5.4:

Tabela 5.4 – Relação entre tipos de controladores e melhoria em termos de energia, em comparação com o controlador ON-OFF, com perturbações.

Controlador	**Percentual de melhoria (em comparação ao ON-OFF)**
PI	38,53%
RST	41,32%
MPC	-

Em se tratando do caso com perturbações, vemos que o RST é mais econômico e, portanto, a característica adaptativa torna o controlador melhor, como imaginávamos. Anali-

sando as Figuras 5.20 e 5.22, vemos que, quanto ao acionamento, no começo, o controlador PI é menos exigente,mas, ao longo do tempo, este gasta mais energia do que o RST, com uma diferença de 10% a 15% entre os dois. A economia de energia efetiva é de 2,79%, como é possível checar a partir da Tabela 5.4 acima. Seria o caso, também, de utilizar um controlador inicial PI, e fazer a transição para o RST perto do ponto de operação.

Ainda nas figuras 5.19 e 5.21, em termos de rapidez, vê-se que o PI demora mais que o RST para atingir a referência, com uma diferença de um pouco mais de 1 hora.

Ao se comparar o desempenho quanto a atingir a referência do controlador RST com e sem variação térmica, olhando as figuras 5.12 e 5.21, vemos que este não muda muito, com uma variação em torno de 5 a 6 minutos.

A tabela 5.5 resume o desempenho de cada controlador, nas duas configurações de carga térmica.

Tabela 5.5 – Tabela resumo de desempenho dos controladores (configuração, erro RMS e consumo energético).

Carga Térmica	**Controlador ON-OFF**		**Controlador PI**		**Controlador Adaptativo**		**Controlador MPC**	
	E_{RMS} *[°C]*	Consumo [kWh]	E_{RMS} *[°C]*	Consumo [kWh]	E_{RMS} *[°C]*	Consumo [kWh]	E_{RMS} *[°C]*	Consumo [kWh]
Constante	*0,1399*	*6,69*	*1,5524*	*3,12*	*0,5291*	*3,42*	0,91	5,25
Variante	*0,1236*	*7,89*	*1,1850*	*4,85*	*0,4342*	*4,63*	-	-

6 CONCLUSÃO

Como era proposta deste trabalho, devíamos elaborar um controlador preditivo com módulos interligados através da internet para um sistema de ar-condicionado híbrido e comparar o seu desempenho com as técnicas desenvolvidas em trabalhos passados.

Para se obter uma boa estratégia de controle é preciso obter uma estimativa das perturbações presentes no ambiente. Para tanto foi desenvolvido uma rede de sensores de temperatura assim como um sistema de detecção de pessoas no ambiente utilizando um feed de vídeo com o auxílio de redes neurais.

O modulo de detecção de pessoas funcionou da maneira esperada para o hadware(Raspberry PI 4) onde foi implementada que possui recursos limitados, assim, tivemos que utilizar um modelo menos poderoso.

Em trabalhos anteriores fizemos uma identificação recursiva de uma sala de reuniões, feita com base no algoritmo de Mínimos Quadrados Recursivo. O erro quadrático médio obtido para a sala foi de 0,015. A identificação recursiva é rápida, pois há o paralelismo na ação de identificar e de coletar dados, e é mais assertiva, principalmente se adaptarmos o algoritmo de identificação com um fator de esquecimento, cuja função é priorizar as medidas mais recentes.

Posteriormente, a estimação dos parâmetros determinada pela identificação foi usada no "design" de um controlador adaptativo. À priori, foram desenvolvidos controladores ON-OFF, PI e adaptativo RST estes últimos associados a um PWM.

Sobre os controles clássicos, chegamos à conclusão de que o PI é mais vantajoso para esse tipo de trabalho, uma vez que é o mais econômico em termos de acionamento.

A finalidade de se fazer um controle preditivo é a de tornar-se um controlador simples mais flexível à demanda de carga. Com isso em mente, elaboramos um controlador MPC.

Com os resultado foi visto que se gasta menos energia com uma abordagem adaptativa. Devemos esclarecer que, embora o controle adaptativo seja mais lento, isto não é exatamente um problema em se tratando de processos térmicos, em que as variações são lentas. Podemos acrescentar ainda um aspecto benéfico dessa lentidão, que é não exigir demais do atuador ao mudar de estado (ligado e desligado). É preciso certo tempo para ligar e desligar o compressor e, caso este tempo não seja respeitado, a vida útil do aparelho é reduzida sensivelmente.

O desempenho do controlador MPC foi diminuído devido ao desempenho da rede de internet encontrada no laboratório. A rede se mostrou instável o que fez que o controlador perdesse entradas de retro alimentação fazendo com que o cálculo do sinal de controle

estivesse errado para o real estado presente no ambiente.

Ao fazer o experimento na sala de reuniões do laboratório, consideramos duas configurações: uma situação em que a variação térmica é mínima, isto é, há pouca variação térmica, e outra situação, com bastante variação, a qual denominamos, ao longo do trabalho, como perturbações.

Para um ambiente sem variação térmica, vimos uma peculiaridade: o controlador PI se saiu melhor quanto à economia de energia. Além disso, vimos que este controlador foi um pouco mais lento que o MPC e o RST. O fato é que um ambiente com pouca variação térmica possui parâmetros mais constantes, e, assim, a característica adaptativa do controlador RST não foi observada.

Quando expostos à variação térmica, tivemos o resultado que esperávamos: o controlador RST mostrou grande economia de energia em relação ao controlador ON-OFF, e relativa economia em relação ao controlador PI. Inclusive, o controle RST foi mais rápido que este último, com uma diferença de tempo um pouco mais de 1 hora. Devemos esclarecer que, quando falamos do aspecto benéfico de uma demora do controlador para atingir a referência, em virtude da grande diferença temporal para este caso específico, essa lentidão deixa de ser benéfica.

Devido à pandemia de COVID-19(2020) não foi possível realizar o experimento com o controlador MPC em condições de perturbação devido à presença de pessoas.

Com isso, concluímos que o controlador adaptativo possui um desempenho superior ao controlador preditivo, para ambientes térmicos sujeitos a variações térmicas, é a melhor opção, tanto em desempenho quanto em rapidez, dentro do contexto deste projeto.

Este trabalho espera auxiliar outros pesquisadores em ambientes prediais inteligentes do Laboratório de Automação e Robótica. Salientamos que é possível otimizar alguns resultados. Seguem abaixo algumas sugestões:

- Usar mais variáveis na estimação do modelo, como humidade, radiação solar;
- Utilizar módulos de rede que apresentem uma maior estabilidade de conexão;
- Testar outros algoritmos de identificação, como o Filtro de Kalman e suas variantes de estimação não-linear;
- Desenvolver um controlador híbrido, em que duas ou mais técnicas de controle sejam empregadas, de acordo com a necessidade;

REFERÊNCIAS BIBLIOGRÁFICAS

1 LAMBERTS, R. *Conforto e Stress Térmico*. [S.l.], 2011.

2 BAUCHPIESS, A. et al. First-principles structured identification for predictive hvac control. *CLCA 2006*, 2006.

3 OLIVEIRA, F. A. R. de. *Instrumentação e Identificação de um Ambiente Predial Visando Controle Preditivo do Conforto Térmico*. 2010.

4 LJUNG, L. *System Identification – Theory for the user*. Upper Saddler River,New Jersey: Prentice Hall, 1999.

5 RICCO, R. A. *Identificação de Sistemas Utilizando Métodos de Subespaços*. 2012.

6 AGUIRRE, L. A. *Introdução à Identificação de Sistemas – Técnicas Lineares e Não-Lineares Aplicadas a Sistemas Reais*. [S.l.]: Ed. UFMG, 2007.

7 BOLDRIN, J. L. et al. *Algebra Linear*. [S.l.]: Harbra ltda, 1980.

8 ZHANG, A. et al. *Dive into Deep Learning*. [S.l.], 2019.

9 RODRIGUES, A. S.; SILVA, M. P. M. da. *Controle Antecipativo por estimativa de carga térmica em vídeo visando eficiência energética na climatização de ambiente predial*. 2018. Trabalho de Graduação em Engenharia de Controle e Automação, Publicação FT.TGn∘022, Faculdade de Tecnologia, Universidade de Brasília, Brasília, DF, 101p.

10 ARAUJO, H. A. S. de; MELO, M. C. C. *Estudo do Controle Adaptativo na Eficiência Energética de Climatização Utilizando Ambiente Predial*. 2013. Trabalho de Graduação em Engenharia de Controle e Automação, Publicação FT.TG-nº 05/2013, Faculdade de Tecnologia, Universidade de Brasília, Brasília, DF, 75p.

11 AçãO Liga-Desliga (On-off). <http://www.ece.ufrgs.br/~jmgomes/pid/Apostila/apostila/node20.html>. Accessed: 2020-12-12.

12 DINGYU, X.; CHEN, Y.; ATHERTON, D. P. *Linear feedback control: analysis and design with MATLAB*. [S.l.]: Society for Industrial and Applied Mathematics, 2007.

13 ASTROM, K. J.; BJORN, W. *Adaptative control*. [S.l.]: Addison Wesley, 1995.

14 CAMACHO, E. F.; ALBA, C. B. *Model Predictive Control*. 2. ed. [S.l.]: Springer London, 2007.

15 OGUNNAIKE, B. A.; RAY, W. H. *Instructors manual for process dynamics, modeling, and control*. [S.l.]: Oxford University Press, 1997.

16 CLARKE, D. W.; MOHTADI, C.; TUFFS, P. S. *Generalized predictive control: Part 1; the basic algorithm*. [S.l.]: University of Oxford, 1984.

17 MATLAB Model Predictive Control Toolbox. 2017. The MathWorks, Natick, MA, USA.

18 DALLAS SEMICONDUCTORS. *DS18B20-PAR 1-Wire Parasite-Power Digital Thermometer*. [S.l.], 2019. Rev. 6.

19 AOSONG ELECTRONICS CO.,LTD. *Digital-output relative humidity temperature sensor/module DHT22*. [S.l.], 2018.

20 RASPBERRY Pi 4 Model B specifications. <https://www.raspberrypi.org/products/raspberry-pi-4-model-b/specifications/?resellerType=home>. Accessed: 2020-12-12.

21 Associação Brasileira de Normas Técnicas. *NBR 6401*: Instalações centrais de ar-condicionado para conforto - parâmetros básicos de projeto. 1980.

22 SOUTO, R. F.; BORGES, G. A.; BAUCHSPIESS, A. Estudo sobre o controle térmico para ambientes prediais. In: ____. *XII CONGRESSO BRASILEIRO DE AUTOMÁTICA*. [S.l.: s.n.], 2006. p. 3224–3229.

Anexos

A ANEXO I - CÓDIGO DE AQUISIÇÃO DE TEMPERATURAS

```
#include <ESP8266WiFi.h>
#include <SoftwareSerial.h>
#include <math.h>
#include <DHT.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <ThingSpeak.h>

#define ONE_WIRE_BUS 5

SoftwareSerial r(D5,D6);

//Parametros da rede
const char* ssid = ""; //your WiFi Name
const char* password = "";  //Your Wifi Password
WiFiClient client;
const char* apiKey = "YourAPIKey";
unsigned long myChannelNumber = //Your Channel ID;

//DS18b20

OneWire barramento(ONE_WIRE_BUS);
DallasTemperature sensor(&barramento);

//DHT22

const int DHTPinSalaExterna = D3;
const int DHTPinExterna = D4;
#define DHTTYPE DHT22
DHT dhtSalaExterna(DHTPinSalaExterna, DHTTYPE);
DHT dhtExterno(DHTPinExterna, DHTTYPE);


//Variaveis de temperatura
float tempSala;
float auxTempSala=0;
float tempExterna;   // Temperatura da Externa em C
float auxTempExterna=0;
float tempSalaExterna;   // Temperatura da Sala Externa em C
float auxTempSalaExterna=0;
```

```
41
42 float data;
43 float reciveCurrent();
44
45 void setup(){
46
47     Serial.begin(115200);
48     delay(10);
49     r.begin(115200);
50     pinMode(LED_BUILTIN,OUTPUT);
51
52     Serial.println();
53     Serial.println();
54
55     Serial.print("Connecting to ");
56     Serial.println(ssid);
57     WiFi.begin(ssid, password);
58     while (WiFi.status() != WL_CONNECTED) {
59         delay(500);
60         Serial.print(".");
61     }
62     WiFi.setAutoReconnect(true);
63     WiFi.persistent(true);
64   ThingSpeak.begin(client);
65   Serial.println("");
66   Serial.println("WiFi connected");
67   digitalWrite(LED_BUILTIN,LOW);
68
69 //  udp.begin(localUdpPort);
70   dhtSalaExterna.begin();
71   dhtExterno.begin();
72   sensor.begin();
73   pinMode(sensorPin, INPUT);
74 }
75
76 void loop(){
77
78   digitalWrite(LED_BUILTIN,HIGH);
79   if(WiFi.status() != WL_CONNECTED)
80   {
81     ESP.reset();
82     }
83   sensor.requestTemperatures();
84   tempSala = sensor.getTempCByIndex(0);
85
86   if(tempSala != DEVICE_DISCONNECTED_C){
87     Serial.println(tempSala);
```

```
88    }else{
89      Serial.println("Error");
90      }
91
92
93  //Sensor Externo
94
95  tempExterna = dhtExterno.readTemperature();   // calcula a temperatura
     em celsius
96  Serial.println(tempExterna);
97
98  //Sensor DHT22 da sala externa
99
100 tempSalaExterna = dhtSalaExterna.readTemperature();
101 //Serial.println(tempSalaExterna);
102
103 //Leitura de energia
104
105
106 currentValue = reciveCurrent();
107
108 Serial.println(currentValue);
109 float PotConsumida = 0;
110 PotConsumida = (currentValue * 220)/1000;
111
112 //String t1,t2,t3,p, data;
113
114 if ((!isnan(tempExterna)||tempExterna >= 0) && (!isnan(tempSalaExterna)
      || tempSalaExterna >= 0) && (!isnan(tempSala) || tempSala >= 0))
115   {
116
117     auxTempExterna = tempExterna;
118     auxTempSalaExterna = tempSalaExterna;
119     auxTempSala = tempSala;
120
121
122   }
123
124   ThingSpeak.setField(1,tempSala);
125   ThingSpeak.setField(2,tempExterna);
126   ThingSpeak.setField(3,tempSalaExterna);
127   ThingSpeak.setField(4,PotConsumida);
128   ThingSpeak.writeFields(myChannelNumber,apiKey);
129
130
131
132 delay(2000);
```

```
133   digitalWrite(LED_BUILTIN,LOW);
134   }
135
136 float reciveCurrent(){
137
138
139   if(r.available()>0){
140
141     data = r.parseFloat();
142     r.flush();
143     }
144
145
146   return data;
147
148   }
```

B ANEXO II - CÓDIGO DO MÓDULO DE ACIONAMENTO

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ThingSpeak.h>

const char* server = "api.thingspeak.com";
const char* ssid = ""; //your WiFi Name
const char* password = "";  //Your Wifi Password

WiFiClient client;
const char* apiKey = "YourAPIKey";
unsigned long myChannelNumber = //Your Channel ID
unsigned int field=1;
int controlSignal=1;

void setup() {
  Serial.begin(115200);
  delay(10);
  pinMode(LED_BUILTIN,OUTPUT);
  pinMode(D4,OUTPUT);
  digitalWrite(LED_BUILTIN,LOW);

  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
      delay(500);
      Serial.print(".");
  }

  WiFi.setAutoReconnect(true);
  WiFi.persistent(true);

  ThingSpeak.begin(client);
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.print("IP address:");
  Serial.print(WiFi.localIP());
  digitalWrite(LED_BUILTIN,HIGH);
```

```
41
42 }
43
44 void loop() {
45   // put your main code here, to run repeatedly:
46
47   if(WiFi.status() != WL_CONNECTED)
48   {
49     ESP.reset();
50     }
51
52
53   controlSignal = ThingSpeak.readIntField(myChannelNumber,field,apiKey);
54   Serial.println(controlSignal);
55
56   if(controlSignal == 0)
57   {
58     digitalWrite(D4,LOW);
59     digitalWrite(LED_BUILTIN,HIGH);
60   }else
61   {
62     digitalWrite(D4,HIGH);
63     digitalWrite(LED_BUILTIN,LOW);
64   }
65
66   delay(2000);
67
68 }
```

C ANEXO III - CÓDIGO DO ALGORITMO DE IDENTIFICAÇÃO (MATLAB)

```
%algoritmo de identificacao recursiva
clc
% in= Sinal;
in=acio;
out=T;
% out=lopass_butterworth(T,0.2,2,1);
% out=T_o;% out=t_o;
% fator de decima  o a ser usado
fd=1;
% periodo de amostragem em segundos
Ts=fd*2;
% referir os dados ao ponto de opera  o em que foi feito o teste
[a b]=size(in);
[c d]=size(out);
u=in;
y=out;
% condi  es iniciais
P=eye(4)*10^6;
ini=3;% teta99(:,ini-1)=[1.918;-0.9191;-0.0219;0.0154];
teta99(:,ini-1)=[0.01;0.01;0.01;0.01];
xi(1,ini-1)=0;% vetor de res duos
mse99=0;

yhat99(ini-1)=y(ini-1);% aqui a sa da ainda n o foi estimada
yhat99(ini-2)=y(ini-2);% ent o s o usados valores medidos
% fator de esquecimento 0.835 e fd=1%fator de esquecimento

lambda=0.9502;
erro99=mse99/(length(y));
%calcula o erro m dio quadr tico
% Algoritmo recursivo
For k=ini:length(y);
psi_k=[y(k-1);y(k-2);u(k-1);u(k-2)];
K_k=(P*psi_k)/((psi_k'*P*psi_k)+1);
teta99(:,k)=teta99(:,k-1)+K_k*(y(k)-psi_k'*teta99(:,k-1));
P=(1/lambda)*(P-
(psi_k*psi_k'*P)/(psi_k'*P*psi_k+lambda));yhat99(k)=[yhat99(k-
1);yhat99(k-2);u(k);u(k-1)]'*teta99(:,k);utilizando dados
passados
xi(k)=y(k)-yhat99(k);% res duo
```

```
41 mse99=(xi(k))^2+mse99;% esse somat rio    utilizado para calcular
42 o MSE
43 end;
44 teta=zeros(4,length(out));
45 forh=1:4teta(h,:)=lopass_butterworth(teta99(h,:),0.05,Ts,2);%fil
46 tra os parametros utilizando um filtro passa-baixas
47 end
48 % valor predito(estimado)
49 figure(1)
50 subplot(2,1,1);
51
52 plot(teta(1,:),'b');
53 hold on;
54 plot(teta(2,:),'r');
55 plot(teta(3,:),'g');
56 plot(teta(4,:),'y');
57 xlabel('Tempo (s)');
58 ylabel('Par metros Filtrados');
59 legend('\theta_{1}','\theta_{2}','\theta_{3}','\theta_{4}');
60 subplot(2,1,2);
61 plot(teta99(1,:),'b');
62 hold on
63 plot(teta99(2,:),'r');
64 plot(teta99(3,:),'g');
65 plot(teta99(4,:),'y');
66 xlabel('Tempo (s)');
67 ylabel('Par metros');
68 legend('\theta_{1}','\theta_{2}','\theta_{3}','\theta_{4}');
69 hold off
70 figure(2)
71 plot(y','b');
72 hold on
73 plot(yhat99,'r')
74 hold off
75 xlabel('Tempo(s)')
76 ylabel('Temperatura')
77 legend('Medido filtrado','Simulado')
```

D ANEXO IV - CÓDIGO PARA CONTAGEM DE PESSOAS

```
################ People count Using Tensorflow Classifier ################################
#Author: Heyder Araujo
#Description:
#This program uses a Tensorflow classifier to peerform people detection
#and count the number of people in the video feed.
#The model used is the ssdlite_mobilenet_v2_coco_2018_05_09 downloaded from the Tensorflow model zoo.
#This code is based on the exemple code from the Tensorflow repositori on Github:
# https://github.com/tensorflow/models/blob/master/research/object_detection/object_detection_tutorial.ipynb


# import packages

import os
import cv2
import numpy as np
import tensorflow.compat.v1 as tf
import argparse
import sys

import thingspeak




tf.disable_v2_behavior()
scoreMin = 0.4


#thingspeak
channelID = #Channel ID
writeKey = "#API Key"

Channel = thingspeak.Channel(id=channelID,api_key=writeKey)


#Camera constants
# Normal resolution:
```

```
38 IM_WIDTH = 1280
39 IM_HEIGHT = 720
40 #Smaller Resolution for a litle better FPS rate:
41 #IM_WIDTH = 640
42 #IM_HEIGHT = 480
43
44 sys.path.append('..')
45
46 # Import utilites
47 from utils import label_map_util
48
49
50 # Name of the directory containing the object detection module we're
       using
51 MODEL_NAME = 'ssdlite_mobilenet_v2_coco_2018_05_09'
52
53 # Grab path to current working directory
54 CWD_PATH = os.getcwd()
55
56 # Path to frozen detection graph .pb file, which contains the model that
       is used
57 # for object detection.
58 PATH_TO_CKPT = os.path.join(CWD_PATH,MODEL_NAME,'frozen_inference_graph.
       pb')
59
60 # Path to label map file
61 PATH_TO_LABELS = os.path.join(CWD_PATH,'data','mscoco_label_map.pbtxt')
62
63 # Number of classes the object detector can identify
64 NUM_CLASSES = 90
65
66 ## Load the label map.
67 # Label maps map indices to category names, so that when the convolution
68 # network predicts `1`, we know that this corresponds to `person`.
69 # Here we use internal utility functions, but anything that returns a
70 # dictionary mapping integers to appropriate string labels would be fine
71 label_map = label_map_util.load_labelmap(PATH_TO_LABELS)
72 categories = label_map_util.convert_label_map_to_categories(label_map,
       max_num_classes=NUM_CLASSES, use_display_name=True)
73 category_index = label_map_util.create_category_index(categories)
74
75 # Load the Tensorflow model into memory.
76 detection_graph = tf.Graph()
77 with detection_graph.as_default():
78     od_graph_def = tf.GraphDef()
79     with tf.gfile.GFile(PATH_TO_CKPT, 'rb') as fid:
80         serialized_graph = fid.read()
```

```
81          od_graph_def.ParseFromString(serialized_graph)
82          tf.import_graph_def(od_graph_def, name='')
83
84      sess = tf.Session(graph=detection_graph)
85
86
87  # Define input and output tensors (i.e. data) for the object detection
        classifier
88
89  # Input tensor is the image
90  image_tensor = detection_graph.get_tensor_by_name('image_tensor:0')
91
92  # Output tensors are the detection boxes, scores, and classes
93  # Each box represents a part of the image where a particular object was
        detected
94  detection_boxes = detection_graph.get_tensor_by_name('detection_boxes:0')
95
96  # Each score represents level of confidence for each of the objects.
97  # The score is shown on the result image, together with the class label.
98  detection_scores = detection_graph.get_tensor_by_name('detection_scores:0
        ')
99  detection_classes = detection_graph.get_tensor_by_name('detection_classes
        :0')
100
101 # Number of objects detected
102 num_detections = detection_graph.get_tensor_by_name('num_detections:0')
103
104 # Initialize frame rate calculation
105 frame_rate_calc = 1
106 freq = cv2.getTickFrequency()
107 font = cv2.FONT_HERSHEY_SIMPLEX
108
109 #This function perform the detection and return the boxes, the scores,
        the classes and thenumber of detections
110
111
112 def detect(image):
113     image_np_expanded = np.expand_dims(image, axis=0)
114     (boxes, scores, classes, num) = sess.run(
115         [detection_boxes, detection_scores, detection_classes,
        num_detections],
116         feed_dict={image_tensor: image_np_expanded}) # Using the model
        for detection
117
118     im_height, im_width,_ = image.shape
119     boxes_list = [None for i in range(boxes.shape[1])]
120     for i in range(boxes.shape[1]):
```

```
121         boxes_list[i] = (int(boxes[0,i,0] * im_height),
122                     int(boxes[0,i,1]*im_width),
123                     int(boxes[0,i,2] * im_height),
124                     int(boxes[0,i,3]*im_width))
125 
126     return boxes_list, scores[0].tolist(), [int(x) for x in classes[0].
    tolist()], int(num[0])
127 
128 
129 
130 
131 
132 
133 # Initialize USB webcam feed
134 camera = cv2.VideoCapture(0)
135 camera.set(cv2.CAP_PROP_FOURCC,cv2.VideoWriter_fourcc('M','J','P','G'))
136 ret = camera.set(3,IM_WIDTH)
137 ret = camera.set(4,IM_HEIGHT)
138 
139 
140 while(True):
141 
142     t1 = cv2.getTickCount()
143     count = 0
144 
145     # Acquire frame
146     ret, frame = camera.read()
147     frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
148     #uses the detect fuction to expand the dimensions  of the frame and
    perform the object detection
149     boxes, scores, classes, num = detect(frame_rgb)
150 
151     #if the detected object is a person the count variable is increased
    and boxes are put around the person
152     for i in range(len(boxes)):
153         if np.any(classes[i] == 1) and np.any(scores[i]>scoreMin):
154 
155             box = boxes[i]
156             cv2.rectangle(frame, (int(box[1]),int(box[0])),(int(box[3]),
    int(box[2])),(25,30,200),2)
157             cv2.putText(frame,'Score = {0:.2f} %'.format(scores[i]*100),(
    box[1],box[0]),cv2.FONT_HERSHEY_SIMPLEX, 1.25,(255,255,0),2,cv2.
    LINE_AA)
158             count+=1
159 
160     # Show the results
161 
```

```
162     print("Valor predito: {}".format(count))
163 
164     cv2.putText(frame,'Count = '+str(count),(10,400),cv2.
    FONT_HERSHEY_SIMPLEX, 1.25,(255,255,0),2,cv2.LINE_AA)
165     cv2.putText(frame,"FPS: {0:.2f}".format(frame_rate_calc),(30,50),font
    ,1,(255,255,0),2,cv2.LINE_AA)
166 
167     #send data to thingspeak
168     try:
169         response = Channel.update({1:count})
170         print(response)
171     except:
172         print("Connection Failed")
173 
174     cv2.imshow('Object detector', frame)
175 
176     t2 = cv2.getTickCount()
177     time1 = (t2-t1)/freq
178     frame_rate_calc = 1/time1
179 
180     # Press 'q' to quit
181     if cv2.waitKey(1) == ord('q'):
182         break
183 
184 
185 camera.release()
186 
187 cv2.destroyAllWindows()
```

ANEXO V - DIAGRAMAS DO CONTROLE ADAPTATIVO

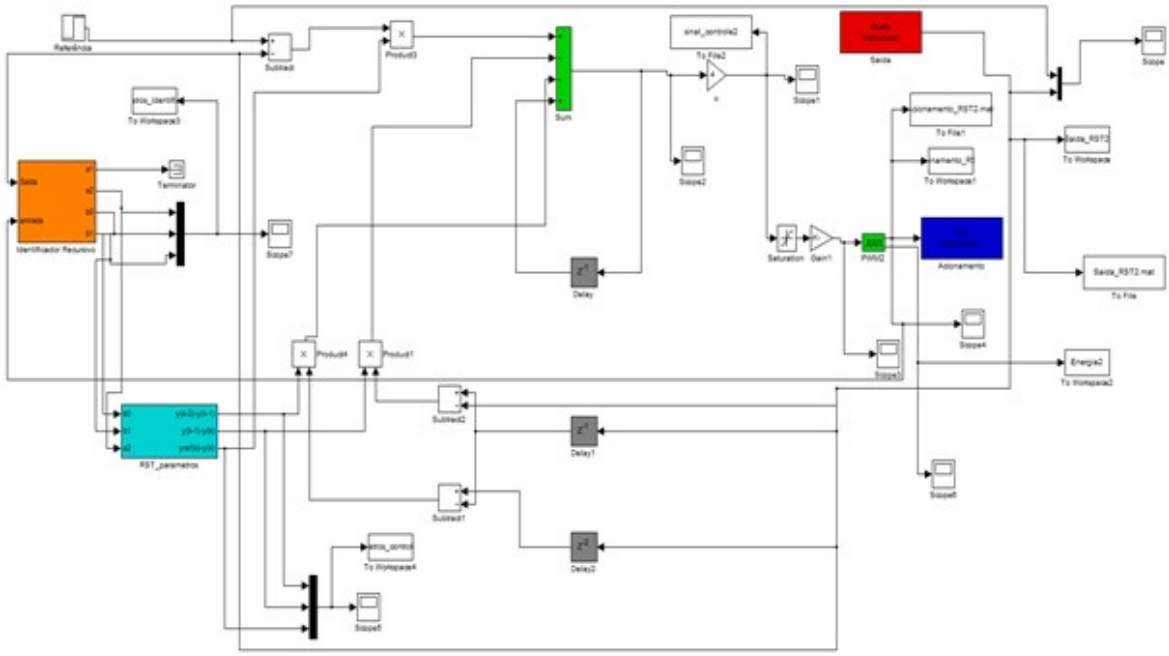

Figura E.1 – Diagramas do Controle Adaptativo

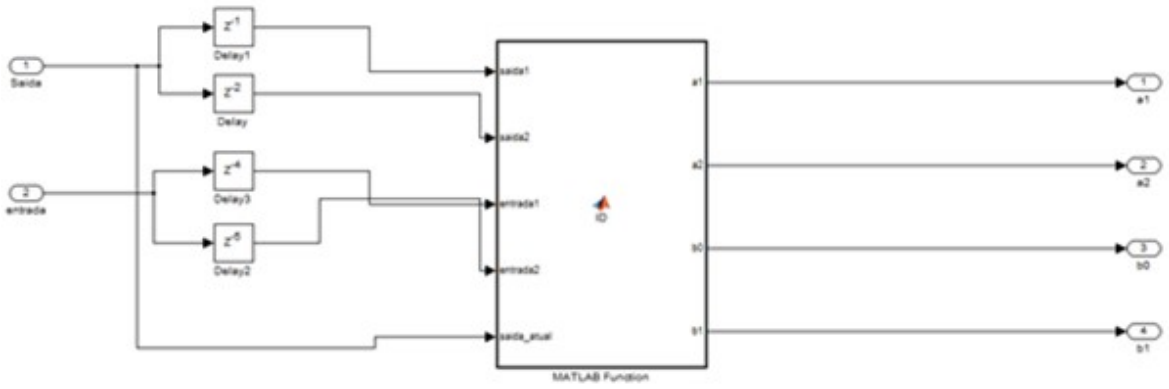

Figura E.2 – Diagrama simulink do processo de identificação.

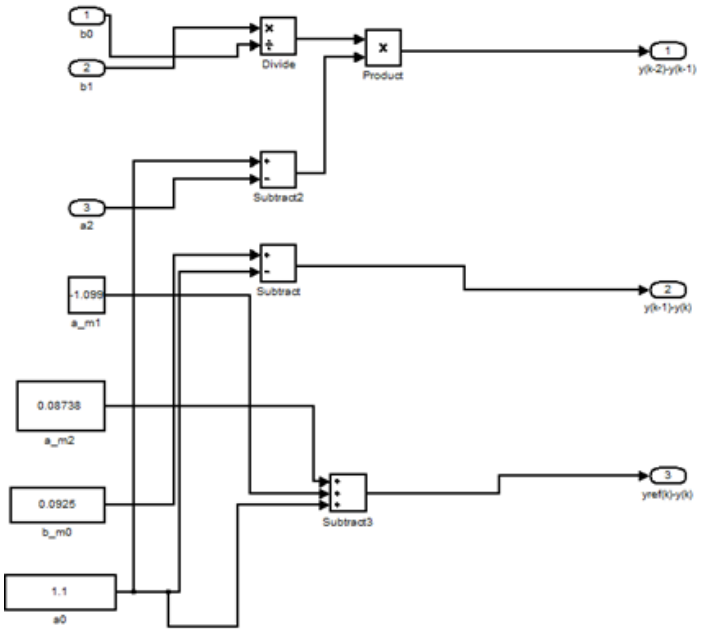

Figura E.3 – Diagrama do simulink para o calculo do sinal de controle.

F ANEXO VI - DIAGRAMA DO CONTROLE ON-OFF

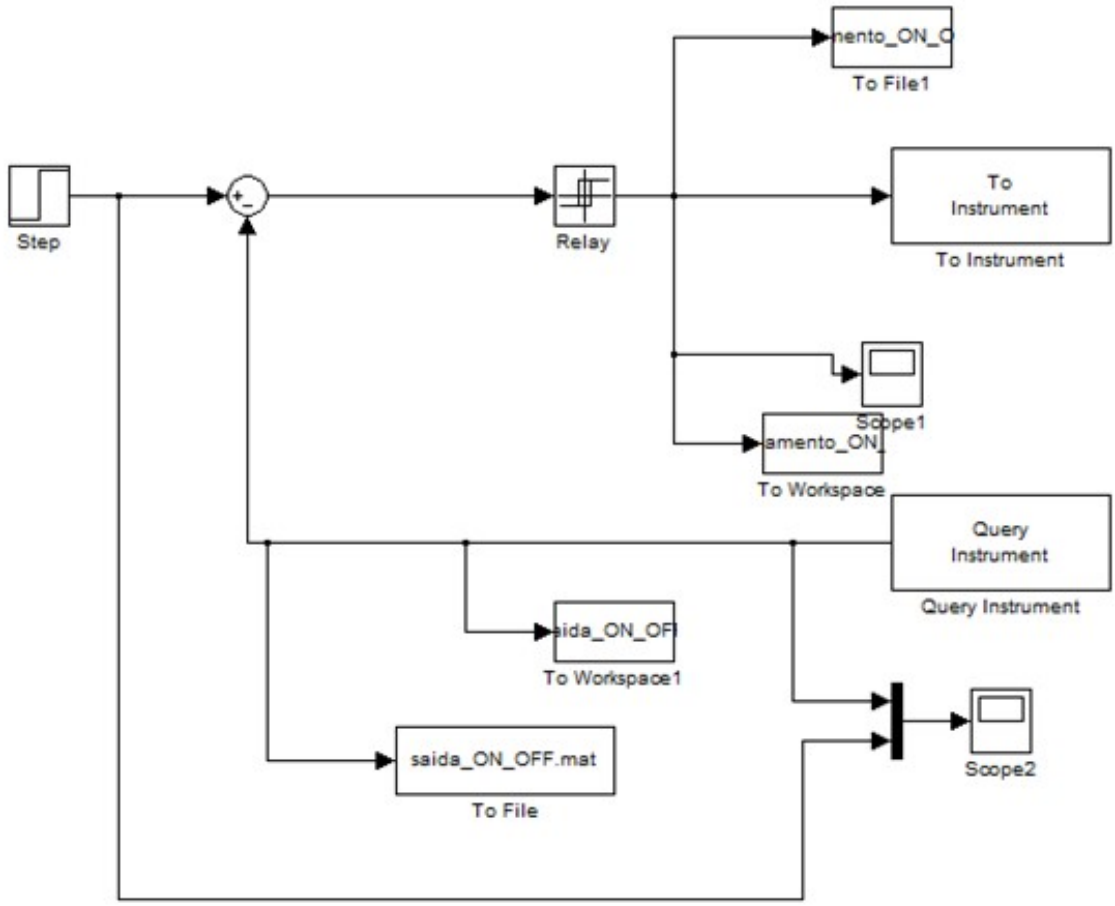

Figura F.1 – Diagrama simulink do controle ON-OFF

G ANEXO VII - DIAGRAMA DO CONTROLE PROPORCIONAL INTEGRATIVO

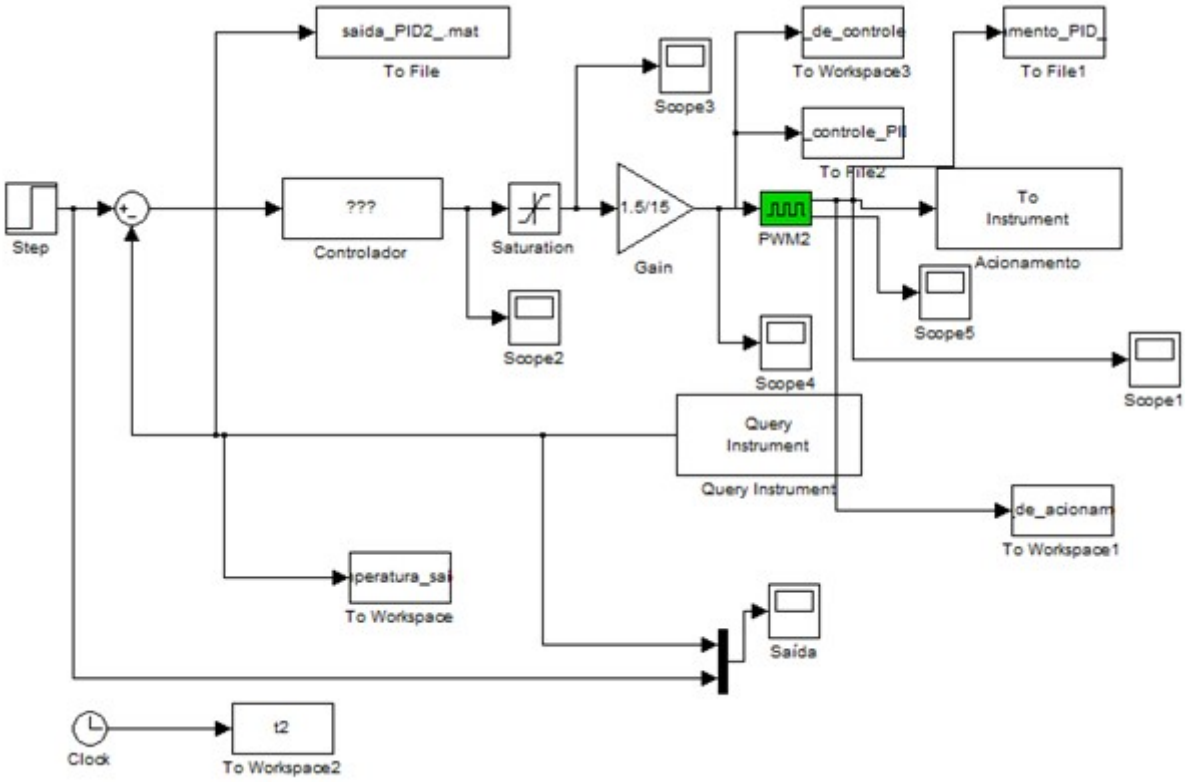

Figura G.1 – Diagrama simulink do controle PI.