

TRABALHO DE GRADUAÇÃO

**CONTROLE ANTECIPATIVO  
POR ESTIMATIVA DE CARGA TÉRMICA  
EM VÍDEO**

Alexandre Saran Rodrigues  
Mariana Pimentel Martins da Silva

Brasília, Dezembro de 2018



**ENGENHARIA  
MECATRÔNICA**  
UNIVERSIDADE DE BRASÍLIA

UNIVERSIDADE DE BRASÍLIA  
Faculdade de Tecnologia  
Curso de Graduação em Engenharia de Controle e Automação

TRABALHO DE GRADUAÇÃO  
**CONTROLE ANTECIPATIVO  
POR ESTIMATIVA DE CARGA TÉRMICA  
EM VÍDEO**

Alexandre Saran Rodrigues  
Mariana Pimentel Martins da Silva

*Relatório submetido como requisito parcial de obtenção  
de grau de Engenheiro de Controle e Automação*

Banca Examinadora

Prof. Adolfo Bauchspiess, ENE/UnB

*Orientador*

\_\_\_\_\_

Prof. Lélío Ribeiro Soares Júnior, ENE/UnB

*Examinador*

\_\_\_\_\_

Prof. Jones Yudi Mori Alves da Silva,

ENM/UnB

*Examinador*

\_\_\_\_\_

**Brasília, Dezembro de 2018**

## FICHA CATALOGRÁFICA

RODRIGUES, ALEXANDRE S., SILVA, MARIANA P. M..

Controle Antecipativo por estimativa de carga térmica em vídeo visando eficiência energética na climatização de ambiente predial,

[Distrito Federal] 2018.

12p., 69p., 297 mm (FT/UnB, Engenheiro, Controle e Automação, 2018). Trabalho de Graduação – Universidade de Brasília. Faculdade de Tecnologia.

1. Controle Antecipativo

2. Visão Computacional

3. Identificação de Pessoas

4. Eficiência Energética

5. Conforto Térmico

I. Mecatrônica/FT/UnB

II. Título (Série)

## REFERÊNCIA BIBLIOGRÁFICA

RODRIGUES, ALEXANDRE S., SILVA, MARIANA P. M., (2018). Controle Antecipativo por estimativa de carga térmica em vídeo visando eficiência energética na climatização de ambiente predial. Trabalho de Graduação em Engenharia de Controle e Automação, Publicação FT.TG-nº022, Faculdade de Tecnologia, Universidade de Brasília, Brasília, DF, 101p.

## CESSÃO DE DIREITOS

AUTORES: Alexandre Saran Rodrigues e Mariana Pimentel Martins da Silva

TÍTULO DO TRABALHO DE GRADUAÇÃO: Controle Antecipativo por estimativa de carga térmica em vídeo visando eficiência energética na climatização de ambiente predial.

GRAU: Engenheiro de Controle e Automação

ANO: 2018

É concedida à Universidade de Brasília permissão para reproduzir cópias deste Trabalho de Graduação e para emprestar ou vender tais cópias somente para propósitos acadêmicos e científicos. Os autores reservam outros direitos de publicação e nenhuma parte desse Trabalho de Graduação pode ser reproduzida sem autorização por escrito dos autores.

---

Alexandre Saran Rodrigues

Mariana Pimentel Martins da Silva

SCLN 409 Bl. A Apt. 205, Asa Norte

Cond. Solar de Brasília 2 Conj. 6 N. 17, Jardim Botânico

70857-510 Brasília – DF – Brasil

71680-349 Brasília – DF – Brasil

## Dedicatórias

*Dedico este trabalho à minha família por todo amor incondicional e apoio que recebo. Dedico, também, ao meu namorado que estava comigo todos os momentos deixando meu riso mais feliz.*

*Mariana Pimentel Martins da Silva*

*Dedico este trabalho à minha família, meus grandes incentivadores e que tornaram esse sonho possível. Dedico ainda à minha namorada, que esteve comigo e me inspirou a chegar até aqui.*

*Alexandre Saran Rodrigues*

## Agradecimentos

*Agradeço inicialmente ao professor Adolfo por todo o suporte e orientação, com uma enorme paciência e conhecimento, não medindo esforços em nos auxiliar e dar dicas não só sobre o Trabalho de Graduação, mas também sobre a universidade e sobre o mercado de trabalho.*

*À toda minha família que sempre me incentivou a correr atrás dos meus sonhos e que tornou possível eu estar aqui hoje. Em especial, ao meu pai, minha mãe e à minha avó Ilda que são o meu porto seguro e que sempre estão comigo apesar da distância. Ainda sobre a família, sou bastante grato à minha família informal, os meus dois grandes amigos que tenho para toda a vida: Paulo Vitor Morato Melo e Victor Gomide Ferraz, grandes mentores e incentivadores que sempre se fazem presentes.*

*Agradeço de maneira especial à Mecajun, empresa responsável por grande parte do conhecimento extracurricular que tenho hoje, responsável ainda por recuperar e aumentar a minha paixão pelo curso e onde consolidei laços especiais.*

*À todos os amigos que o curso de Mecatrônica me proporcionou, amigos que marcaram a minha história e com quem cresci pessoalmente. De maneira especial: Mariana Pimentel, Rodrigo Flores, Iracema Siqueira, Carlos Rocha, Rafaela Senhoroto, Daniel Amaral, Kenneth Lui, Eric Gil, Francisco Matheus, Thalbert Miranda, João Lucas e Gabriell Barbedo.*

*Finalmente, agradeço à Mariana Pimentel, minha parceira para todas as horas, uma pessoa excepcional, que com toda a certeza me inspirou a dar o meu melhor em minha vida e que foi fundamental para a realização deste trabalho.*

*Alexandre Saran Rodrigues*

*Agradeço ao professor Adolfo por toda a ajuda oferecida durante o projeto e preocupação com nós, seus alunos e orientados.*

*À minha família que sempre me oferece o que há de melhor, especialmente meu pai, minha mãe e meus irmãos. Ao meu irmão Tiago que é meu monitor em todas as matérias da universidade e na vida, contribuindo para o meu crescimento e aprendizado. À empresa Júnior Mecajun por me oferecer minha primeira experiência profissional e me capacitar com projetos, palestras e treinamentos, além dos aprendizados como liderança, gerenciamento de projetos e gestão de tempo que tive ao participar da empresa e da sua mesa diretora.*

*À todos que me ajudaram nesses anos de graduação. Aos meus colegas nas matérias com quem passei fins de semana ou até noites fazendo trabalhos e estudando para provas. Em especial, a Rafa Senhoroto, o Carlos Rocha, o Rodrigo Flores, o Eric Gil, o Daniel Amaral, o Francisco Matheus, o Thalbert Miranda e a Iracema Siqueira. E à minha amiga de coração desde que consigo me lembrar, a Júlia Caroline.*

*Por fim, agradeço ao Alexandre Saran que está sempre ao meu lado para me ajudar nas mais diversas dificuldades e para comemorar todas as conquistas, além de ser uma dupla com quem sempre gosto de trabalhar independente do projeto. Obrigada por toda ajuda tanto durante o meu percurso na universidade quanto no trabalho final.*

*Mariana Pimentel Martins da Silva*

---

## RESUMO

É proposta deste projeto implementar controladores que propiciem conforto térmico com eficiência energética em equipamentos de climatização. O ponto de partida deste trabalho é o tratamento explícito de perturbações em um ambiente predial. O ambiente estudado, uma sala de reuniões, tem como parcelas majoritárias de influência o número de ocupantes e as temperaturas em ambientes vizinhos. Na segunda parte do trabalho são projetados controladores antecipativos, capazes de estimar a perturbação. Finalmente, a última etapa é a validação: teste dos controladores e a comparação com técnicas clássicas, bem como com trabalhos anteriores.

O controlador antecipativo trás benefícios, visto que numa malha de controle tradicional a realimentação informa o controlador sobre a perturbação. O caminho via processo mais realimentação insere um atraso considerável na mitigação da perturbação. A antecipação permite rejeitar a perturbação mais rapidamente. Logo, o controle antecipativo é mais rápido que um observador de perturbações tradicional.

A aquisição dos dados necessários foi feita com a utilização de algoritmos de Visão Computacional e da biblioteca *OpenCV* para a contagem do número de pessoas, bem como de sensores de temperatura localizados na sala onde se realizarão os ensaios, na sala vizinha e no ambiente externo. As imagens serão capturadas por uma câmera instalada na sala de reunião e o processamento será feito em um *Raspberry Pi*. A informação dos sensores será lida por uma placa *Arduino* e enviada ao computador embarcado. Em posse das informações de interesse, é possível ajustar o sinal de controle de modo a antecipar a perturbação do número de pessoas e cancelar as demais perturbações. Finalmente, o sinal de controle é convertido em um sinal PWM e enviado ao *Arduino*, que por sua vez aciona um relé para atuar sobre o compressor.

Foram realizados testes com os controladores liga-desliga, PI e antecipativo em intervalos de 8h e 24h. Comprovou-se que os controladores tradicionais apresentavam grande dificuldade quando o ambiente estava ocupado por mais de quatro pessoas, resultando em erros superiores. Ao implementar o controle antecipativo, percebeu-se melhorias superiores a 10% no erro, garantindo um maior conforto térmico. Além disso, os ensaios mostraram ainda uma economia energética superior a 2,5% quando comparado ao controlador PI e a 24% quando comparado ao tradicional liga-desliga. Isso considerando que o controlador antecipativo foi submetido a condições mais severas em seus ensaios.

Palavras Chave: Controle Antecipativo, Eficiência Energética, Conforto Térmico, Visão Computacional.

---

## ABSTRACT

This project proposes the implementation of controllers which propitiate thermal comfort and energy efficiency in air conditioning equipment. The main goal of this work is the explicit treatment of disturbances in building environments. The studied environment is highly affected by its occupants and the adjacent room's temperatures. We project feedforward controllers which are able to estimate the disturbance. Finally, we compare our proposed controller with both classic techniques and previous work, testing them in a real environment.

In traditional closed loop controllers, feedback is responsible for notifying the controller about disturbances. So a significant delay is inserted in the system. The forwardness allows treating the disturbance quickly. Therefore, the feedforward controller is faster than a traditional disturbance observer.

Furthermore, we apply computer Vision techniques and OpenCV library to count the number of people in a meeting room. Temperature sensors were installed in the studied environment, in the neighbor room and in the external environment. A camera takes pictures and a Raspberry Pi processes the algorithms. An Arduino board reads the sensors data and changes the relay state. The controller signal is adjusted to anticipate and cancel the disturbances.

Finally, we implemented three different controllers: on-off; proportional-integrative (PI); and feedforward. The traditional controllers presented difficulties when the room was occupied by more than four persons. Overall, the feedforward controller achieved better results, showing an improvement of more than 10% in the goal temperature error. Besides more than 2.5% in energy saving when compared to PI and 24% when compared to on-off.

Keywords: Feedforward Control, Energy Efficiency, Thermal Comfort, Computer Vision.

# SUMÁRIO

<b>1</b>	<b>Introdução</b>	<b>1</b>
1.1	MOTIVAÇÃO	1
1.2	OBJETIVOS	2
1.2.1	OBJETIVOS GERAIS	2
1.2.2	OBJETIVOS ESPECÍFICOS	2
1.3	APRESENTAÇÃO DO MANUSCRITO	3
<b>2</b>	<b>Fundamentação Teórica</b>	<b>4</b>
2.1	CONFORTO TÉRMICO	4
2.1.1	TERMORREGULAÇÃO HUMANA	4
2.1.2	ÍNDICE DE CONFORTO TÉRMICO: PMV	5
2.2	CARGA TÉRMICA	7
2.2.1	CARGA DEVIDO ÀS PESSOAS	7
2.3	VISÃO COMPUTACIONAL	8
2.3.1	HISTOGRAMA DE GRADIENTES ORIENTADOS	8
2.3.2	ALGORITMO VIOLA-JONES PARA DETECÇÃO DE FACES	9
2.3.3	SUBTRAÇÃO DE FUNDO	10
2.4	IDENTIFICAÇÃO DO SISTEMA	11
2.5	CONTROLE	11
2.5.1	CONTROLE LIGA-DESLIGA	11
2.5.2	CONTROLE PI	13
2.5.3	CONTROLE ANTECIPATIVO	13
2.5.4	MODULAÇÃO DE LARGURA DE PULSO - PWM	14
2.6	TRABALHOS ANTERIORES	16
<b>3</b>	<b>Materiais e Ferramentas</b>	<b>18</b>
3.1	<i>Raspberry Pi</i>	18
3.2	CÂMERA	19
3.3	<i>Arduino</i>	20
3.4	MÓDULOS RELÉ DE ESTADO SÓLIDO	21
3.5	SENSOR DE TEMPERATURA E UMIDADE SHT71	22
3.6	SENSOR DE TEMPERATURA E UMIDADE <i>DHT22</i>	23
3.7	MÓDULO SENSOR MAGNÉTICO ( <i>Reed Switch</i> )	24

3.8	<i>OpenCV</i> .....	25
3.9	<i>Matlab</i> E <i>Simulink</i> .....	25
3.10	<i>SQLite</i> .....	25
3.11	SISTEMA DE CLIMATIZAÇÃO .....	26
3.12	VISÃO GERAL.....	27
<b>4</b>	<b>Métodos</b> .....	<b>29</b>
4.1	ESTUDO DO AMBIENTE.....	29
4.2	IDENTIFICAÇÃO DE PESSOAS .....	30
4.3	INSTALAÇÃO ELÉTRICA .....	32
4.4	INSTALAÇÃO DOS SENSORES E ATUADORES .....	33
4.5	IDENTIFICAÇÃO DO MODELO .....	33
4.5.1	IDENTIFICAÇÃO DO SISTEMA - <i>Matlab</i> .....	34
4.6	ABORDAGENS DE CONTROLE .....	35
4.6.1	CONTROLADOR LIGA-DESLIGA .....	36
4.6.2	CONTROLADOR PI .....	37
4.6.3	CONTROLADOR ANTECIPATIVO.....	38
4.7	ENSAIOS NO SISTEMA REAL .....	39
<b>5</b>	<b>Resultados</b> .....	<b>42</b>
5.1	IDENTIFICAÇÃO DE PESSOAS .....	42
5.2	IDENTIFICAÇÃO DA PLANTA .....	46
5.2.1	IDENTIFICAÇÃO DO SISTEMA - <i>Matlab</i> .....	46
5.2.2	SIMULAÇÃO DO SISTEMA EM MALHA ABERTA .....	48
5.3	CONTROLADOR LIGA-DESLIGA .....	49
5.3.1	SIMULAÇÃO DO CONTROLADOR LIGA-DESLIGA.....	49
5.3.2	CONTROLADOR LIGA-DESLIGA NO SISTEMA REAL .....	50
5.3.3	TESTES DE 8H .....	52
5.4	CONTROLADOR PI .....	54
5.4.1	SIMULAÇÃO DO CONTROLADOR PI .....	54
5.4.2	CONTROLADOR PI NO SISTEMA REAL.....	54
5.5	CONTROLADOR ANTECIPATIVO.....	60
5.5.1	SIMULAÇÃO DO CONTROLADOR ANTECIPATIVO .....	60
5.5.2	CONTROLADOR ANTECIPATIVO NO SISTEMA REAL .....	61
5.6	COMPARAÇÃO.....	65
<b>6</b>	<b>Conclusão</b> .....	<b>68</b>
6.1	SUGESTÕES PARA TRABALHOS FUTUROS .....	68
	<b>REFERÊNCIAS BIBLIOGRÁFICAS</b> .....	<b>71</b>
	<b>Apêndices</b> .....	<b>75</b>

<b>I</b>	<b>Programas utilizados.....</b>	<b>76</b>
I.1	PROGRAMA PRINCIPAL .....	76
I.2	CONTAGEM DO NÚMERO DE PESSOAS .....	78
I.2.1	BIBLIOTECA PARA CONTAGEM DO NÚMERO DE PESSOAS.....	78
I.2.2	CLASSE AUXILIAR PARA REPRESENTAR A PESSOA.....	85
I.3	BANCO DE DADOS .....	88
I.4	CONTROLE .....	92
I.5	CÂMERA .....	98
I.6	CONFIGURAÇÕES .....	99
I.7	CÓDIGO DO ARDUINO .....	99
I.8	SIMULAÇÃO <i>Matlab</i> E <i>Simulink</i> .....	103
	<b>Anexos.....</b>	<b>107</b>
<b>I</b>	<b>Diagramas Elétricos.....</b>	<b>108</b>
<b>II</b>	<b>Manual de Instalação e Configuração.....</b>	<b>110</b>
II.1	SISTEMA OPERACIONAL <i>Raspbian</i> .....	110
II.2	CÂMERA .....	110
II.3	BANCO DE DADOS SQLITE.....	112
II.4	OPENCV .....	112

# LISTA DE FIGURAS

2.1	Esquema da Fisiologia Humana e suas Trocas Térmicas [1].	5
2.2	Relação entre o PPD e o PMV [1].	6
2.3	Cadeia de Identificação de Pessoas [2].	9
2.4	Exemplo de Resultado para Detecção de Pessoas a partir do Descritor HOG.	9
2.5	Exemplo de Resultado para Aplicação do Algoritmo Viola-Jones [3].	10
2.6	Exemplo de aplicação do subtrator de fundos em uma imagem [4].	10
2.7	Exemplo de histerese [5].	12
2.8	Exemplo de ondas referentes ao controlador Liga-Desliga [5].	12
2.9	Exemplo de ondas referentes ao controlador PI [6].	13
2.10	Diagrama de blocos de um sistema com Controle Antecipativo.	14
2.11	Comparação entre respostas com e sem perturbações para os controladores por realimentação e controle antecipativo.	15
2.12	Exemplo de onda PWM.	15
3.1	<i>Raspberry Pi 3 Model B</i> [7].	19
3.2	Módulo Câmera <i>Raspberry Pi v2</i> [8].	20
3.3	<i>Arduino Mega</i> [9].	20
3.4	Circuito interno de um <i>SSR</i> [10].	22
3.5	Módulo relé <i>SSR</i> de 4 canais.	22
3.6	Sensor de temperatura e umidade <i>SHT71</i> [11].	23
3.7	Circuito para a interface serial entre o sensor e o <i>Arduino</i> [10].	23
3.8	Diagrama para conexão do sensor <i>DHT22</i> com o <i>Arduino</i> [12].	24
3.9	Sensor Magnético [13].	25
3.10	Sistema de climatização híbrido.	26
3.11	Ciclo de refrigeração [14].	27
3.12	Diagrama de Conexão dos Materiais e Ferramentas.	28
4.1	Planta do ambiente LARA [15].	29
4.2	Ambiente da sala de reunião do LARA.	30
4.3	Quadro elétrico do sistema.	32
4.4	Modelagem para impressão 3D.	33
4.5	Encapsulamentos impressos.	34
4.6	<i>Toolbox</i> de identificação do sistema ( <i>Matlab</i> ).	34
4.7	Modelo <i>simulink</i> para simulação do sistema em malha aberta.	35

4.8	Modelo <i>simulink</i> para simulação do sistema com o controlador liga-desliga.....	36
4.9	Especificação para a sintonia do controlador PI realizada no <i>Matlab</i> . .....	37
4.10	Modelo <i>simulink</i> para simulação do sistema com o controlador PI. ....	38
4.11	Modelo <i>simulink</i> para simulação do sistema com o controlador antecipativo. ....	39
5.1	Contagem correta no vídeo BD1 (ambiente aberto). .....	43
5.2	Contagem incorreta no vídeo BD1 (ambiente aberto).....	43
5.3	Contagem correta no vídeo BD2 (ambiente fechado).....	43
5.4	Contagem incorreta no vídeo BD2 (ambiente fechado). .....	44
5.5	Contagem em tempo-real na sala de reuniões do LARA. ....	46
5.6	Dados coletados para identificação da planta.....	47
5.7	Resultado da simulação do sistema em malha aberta e temperaturas reais medidas..	49
5.8	Resultado com o controlador liga-desliga. ....	50
5.9	Controlador liga-desliga (24h). ....	51
5.10	Controlador liga-desliga (8h). ....	53
5.11	Resultado da simulação do sistema com o controlador PI.....	54
5.12	Controlador PI (24h) com baixo número de pessoas.....	56
5.13	Controlador PI (24h) com alto número de pessoas.....	58
5.14	Controlador PI (8h).....	59
5.15	Resultado da simulação do sistema com o controlador antecipativo.....	61
5.16	Controlador antecipativo (24h). Quando a sala está em uso, a referência é 23°C. Caso contrário, a referência é 25°C. ....	62
5.17	Controlador antecipativo (8h). Quando a sala está em uso, a referência é 23°C. Caso contrário, a referência é 30°C. ....	64
5.18	Controlador antecipativo (8h). Sem mudança de referência conforme o número de ocupantes da sala. ....	66
I.1	Interligação elétrica entre a evaporadora e a condensadora [16]. ....	108
I.2	Diagrama elétrico da condensadora [16]. ....	108
I.3	Diagrama elétrico da evaporadora [16]. ....	109
II.1	Conexão do Módulo Câmera com o Raspberry Pi [17].....	111
II.2	Tela de Configurações do <i>Raspberry Pi</i> com opção <i>Enable Camera</i> Seleccionada [17].	111
II.3	Tela de Configurações do <i>Raspberry Pi</i> com opção <i>Expand Filesystem</i> Seleccionada [17] .....	113
II.4	Tela de Configurações do <i>Raspberry Pi</i> com opção <i>Expand Filesystem</i> Seleccionada [17] .....	116

# LISTA DE TABELAS

2.1	Padrão de Temperatura ASHRAE [18].....	6
3.1	Especificações de Temperatura do Sensor <i>SHT71</i> . ....	23
3.2	Especificações do Sensor <i>DHT22</i> . [19] .....	24
5.1	Comparação entre os Tipos de Controladores. ....	65
5.2	Melhoria Percentual do Controle Antecipativo.....	67

# LISTA DE SÍMBOLOS

## Símbolos Latinos

$G$	Função de Transferência	[°C/°C]
$IBT$	Índice de Balanço Térmico	[°C]
$T$	Temperatura	[°C]
$t$	tempo	[s]
$a$	atraso	[s]

## Símbolos Gregos

$\tau$	Constante de Tempo	[s]
$\theta$	Atraso	[s]

## Grupos Adimensionais

$i, k$	Contador
$e$	Número de Euler
$K$	Ganho
$N$	Número Total
$PMV$	Voto Médio Previsível
$PPD$	Porcentagem Previsível de Insatisfeitos

## Subscritos

<i>antecipativo</i>	Controlador Antecipativo
<i>e</i>	Ambiente Externo
<i>energetica</i>	Gasto Energético
<i>max</i>	Máximo
<i>medida</i>	Medida
<i>min</i>	Mínimo
<i>np</i>	Número de Pessoas
<i>p</i>	Pessoa
<i>ref</i>	Referência
<i>RMSE</i>	Raiz do Erro Médio Quadrático
<i>tradicional</i>	Controlador Tradicional (Liga-Desliga ou PI)
<i>tp</i>	Temperatura das Pessoas
<i>v</i>	Sala Vizinha

## Siglas

A/D	Análogo/Digital
CPU	<i>Central Processing Unit</i> - Unidade Central de Processamento
CSI	<i>Camera Serial Interface</i> - Interface Serial da Câmera
EEPROM	<i>Electrically Erasable Programmable Read-Only Memory</i> - Memória Apagável Eletricamente Programável Somente para Leitura
EMR	<i>Electro-Mechanical Relays</i> - Relé Eletromecânico
FF	<i>Feedforward</i> - Controle de Alimentação
HDMI	<i>High-Definition Multimedia Interface</i> - Interface Multimídia de Alta Resolução
HOG	<i>Histogram of Oriented Gradients</i> - Histograma de Gradientes Orientados
IBT	Índice de Balanço Térmico
ICSP	<i>In Circuit Serial Programming</i> - Programação Serial em Circuito
ID	<i>Identity</i> - Identidade
IDE	<i>Integrated Development Environment</i> - Ambiente de Desenvolvimento Integrado
IHM	Interface Homem-Máquina
IoT	<i>Internet-of-Things</i> - Internet das Coisas
IP	<i>Internet Protocol</i> - Protocolo da Internet
LARA	Laboratório de Automação e Robótica
LED	<i>Light Emitting Diode</i> - Diodo Emissor de Luz
MATLAB	<i>Matrix Laboratory</i> - Laboratório de Matrizes
MIPI	<i>Mobile Industry Processor Interface</i> - Interface do Processador da Indústria Móvel
OpenCV	<i>Open Source Computer Vision Library</i> - Biblioteca Aberta de Visão Computacional
OpenHAB	<i>Open Home Automation Bus</i> - Barramento Aberto de Automação Residencial
OTP	<i>One-Time Programmable</i> - Uma Vez Programável
RAM	<i>Random Access Memory</i> - Memória de Acesso Aleatório
RNA	Rede Neural Artificial
PI	Proporcional-Integrativo
PMV	<i>Predicted Mean Vote</i> - Voto Médio Previsível

PPD	<i>Predicted Percentage of Dissatisfied</i> - Porcentagem Previsível de Insatisfeitos
PWM	<i>Pulse-Width Modulation</i> - Modulação por Largura de Pulso
RMSE	<i>Root Mean Square Error</i> - Raiz do Erro Médio Quadrático
SGBD	Sistema de Gerenciamento de Banco de Dados
SISO	<i>Single Input Single Output</i> - Uma Entrada e Uma Saída
SO	Sistema Operacional
SoC	<i>System on a Chip</i> - Sistema em um Chip
SQL	<i>Structured Query Language</i> - Linguagem de Consulta Estruturada
SRAM	<i>Static Random Access Memory</i> - Memória Estática de Acesso Aleatório
SSR	<i>State Solid Relay</i> - Relé de Estado Sólido
SVM	<i>Support Vector Machine</i> - Máquina de Vetores de Suporte
TRIAC	<i>Triode for Alternating Current</i> - Trido para Corrente Alternada
UART	<i>Universal Asynchronous Receiver-Transmitter</i> - Receptor-Transmissor Assíncrono Universal
USB	<i>Universal Serial Bus</i> - Barramento Serial Universal
NOOBS	<i>New Out Of the Box Software</i> - Programa Novo Fora da Caixa

# Capítulo 1

## Introdução

### 1.1 Motivação

Um ambiente devidamente climatizado é desejado ou, até mesmo, necessário em diversas situações. Segundo [20], “Os edifícios tornaram-se o centro das atividades de negócios e de prestação de serviços, constituindo-se na base da vida urbana pós-moderna”, dessa forma, tornar os edifícios mais agradáveis aos seus usuários e contribuir para um uso mais sustentável deve ser uma grande preocupação do século XXI.

Além disso, a climatização do ambiente pode influenciar diretamente na produtividade de seus ocupantes. No calor, por exemplo, o corpo costuma gastar mais energia para manter a temperatura corporal. Como consequência o corpo pode reagir causando sintomas indesejáveis e desagradáveis como: transpiração, dores de cabeça, cansaço, queda na pressão, entre outros [21].

Objetivando o conforto, faz-se o uso de aquecedores e condicionadores de ar. Em países tropicais como o Brasil, onde as temperaturas muitas vezes são elevadas, o uso de aparelhos condicionadores de ar é frequente. Além disso, o acesso à esses aparelhos de ar-condicionado se torna cada vez mais fácil visto a diminuição dos preços e aumento das opções disponíveis no mercado.

A racionalização do consumo de energia elétrica tornou-se extremamente importante. Neste contexto, vários estudos têm sido feitos no intuito de tornar mais eficiente o uso da energia. Estima-se que as edificações representam 46,7% do consumo total de energia elétrica no Brasil, tendo um potencial enorme no campo de eficiência energética [22] e [23]. Outro estudo aponta que esse consumo irá aumentar em 50% até 2050 [24].

Verificou-se que, em geral, sistemas térmicos são os que representam a maior parcela do consumo energético em edifícios, empresas e residências. Utilizando como exemplo o ambiente doméstico, o consumo de energia elétrica tem como grande vilão os chuveiros elétricos, as geladeiras e os aparelhos de ar-condicionado, dispositivos estes que apresentam um grande potencial de redução de consumo através de soluções inteligentes. Ao observar os ambientes comerciais, conclui-se que o principal gasto energético é em aparelhos de ar-condicionado para refrigeração do ambiente.

Especificamente em edifícios, a refrigeração é responsável por cerca de 47% do consumo de energia elétrica [25] também possibilitando um excelente potencial de redução.

Assim, foram desenvolvidos diversos ambientes inteligentes com o objetivo de racionalizar o uso da energia elétrica, proporcionando uma maior economia financeira e conforto térmico.

No contexto atual de preocupação socioambiental surgiu o termo *Smart Buildings* que pode ser definido como edifícios que ofereçam um ambiente produtivo com uma relação custo-benefício otimizada através de seus sistemas, estrutura, gerenciamento e manutenção ao longo de sua vida útil [26].

Uma grande dificuldade no desenvolvimento desses ambientes inteligentes está ligada à entrada e a saída das pessoas. Ao analisar-se os controles de temperatura para climatização de um ambiente, percebe-se que a presença de um grande número de ocupantes prejudica o controle e aumenta o erro.

Um sistema de visão computacional é uma possível solução para detectar automaticamente à entrada e a saída de pessoas da sala. Associando essa informação com o controle da temperatura, pode-se contrabalancear o efeito dessa perturbação térmica.

## 1.2 Objetivos

Os objetivos deste trabalho foram divididos em duas categorias: objetivos gerais e objetivos específicos. Ambos estão descritos a seguir.

### 1.2.1 Objetivos Gerais

A proposta central deste projeto é racionalizar o uso da energia elétrica em sistemas térmicos prediais, mantendo o conforto térmico. Este objetivo deverá ser alcançado adaptando-se o controlador do ar-condicionado em função da carga térmica de um certo ambiente. O número de ocupantes será estimado pelo uso de uma câmera de vídeo fixada no teto do laboratório. A partir da contagem do número de pessoas, obtida por técnicas de visão computacional, é possível estimar o valor da perturbação causada pelos ocupantes da sala. Após obter essa informação, pode-se atuar no sistema de forma a contrabalanceá-la.

Uma vez que sistemas de controle tradicionais se baseiam somente no erro, dinâmicas lentas demoram a ser percebidas e corrigidas pelo controlador. Portanto, esses controladores demonstram um queda no conforto térmico quando submetidos a grandes perturbações. Sendo assim, o controlador proposto deve apresentar um melhor desempenho por antecipar o efeito da perturbação e diminuir suas consequências no erro do sistema.

### 1.2.2 Objetivos Específicos

Os objetivos específicos do projeto são:

- instrumentação do processo;
- instalação dos sensores e atuadores no ambiente;
- desenvolvimento de um código para identificação automática de pessoas em vídeo;
- implementação de uma malha de controle incluindo o controle antecipativo das perturbações externas causadas pela mudança no número de pessoas presentes na sala e
- comparação entre os resultados obtidos e os resultados utilizando controladores mais tradicionais como o PI e o liga-desliga.

O cenário escolhido para o projeto é a sala de reuniões do LARA. Portanto, a instalação dos sensores e atuadores será feita nesse ambiente, assim como todos os testes.

### 1.3 Apresentação do Manuscrito

O trabalho tem seis capítulos, além dos apêndices e anexos. Para dar um visão geral do texto, apresenta-se aqui o formato do trabalho. O capítulo atual, capítulo 1, contém a introdução, a motivação e os objetivos do trabalho.

O segundo capítulo descreve a fundamentação teórica e a literatura correlata utilizada como base para o projeto, explicando os mais diversos conceitos englobando termodinâmica, visão computacional e controle. Além disso, descreve os trabalhos anteriores realizados no mesmo ambiente, a sala de reuniões do LARA.

O terceiro capítulo cita em detalhes os materiais e as ferramentas utilizadas no decorrer do projeto, assim como as suas especificações e os seus papéis no projeto. O capítulo é finalizado com um visão geral do sistema desenvolvido.

O quarto capítulo descreve os procedimentos e métodos seguidos no decorrer do projeto. Neste capítulo é feito um estudo sobre o ambiente da sala de reuniões. Também explica-se o algoritmo de identificação de pessoas. Além disso, é descrita a metodologia utilizada para identificação do modelo e para o projeto de controladores. Por fim, são explicadas as métricas de comparação usadas como base.

O capítulo 6, por sua vez, conta com a conclusão do trabalho. Além das sugestões para trabalhos futuros.

Em seguida, será apresentada a fundamentação teórica utilizada. O capítulo 2 abordará os principais conceitos necessários para um entendimento do projeto. Os conteúdos abrangem conforto térmico, cálculo de carga térmica, técnicas de visão computacional para detecção de pessoas e projeto de controladores.

## Capítulo 2

# Fundamentação Teórica

### 2.1 Conforto Térmico

O conforto térmico não é uma variável simples de ser definida. Isso porque ela não se limita a fatores físicos, mensuráveis ou computáveis, o conforto térmico depende também de fatores pessoais que variam para cada indivíduo. Ou seja, o conforto térmico é uma variável subjetiva e isso deve ser considerado ao tentar conceituá-lo.

Diversos autores já definiram o conforto térmico. Segundo [27], o conforto térmico é o estado no qual não há impulso de condução para corrigir o comportamento do ambiente. Enquanto que [18] define como uma condição de satisfação da mente com relação ao ambiente térmico. Ainda acrescenta que essa satisfação é influenciada por vários fatores desde diferenças de humor até diferenças culturais e sociais de cada indivíduo.

Em um mesmo ambiente, diferentes pessoas vão apresentar diferentes sensações térmicas. O comportamento humano não é como um sensor que deve apresentar os mesmos resultados quando submetidos as mesmas condições. Portanto, isso significa que pessoas situadas em um espaço semelhante com as mesmas condições de temperatura podem emitir opiniões completamente diferentes sobre a sensação térmica [28].

Alguns autores apresentam um ponto de vista diferente e focam mais no aspecto fisiológico dos seres humanos. Os seres humanos são animais homeotérmicos, ou seja, tendem a manter a temperatura interna do corpo constante independente do ambiente ao qual estão submetidos. Então, pode-se assumir que um ambiente confortável termicamente permite que as pessoas mantenham a temperatura interna constante sem muito esforço e que fiquem em equilíbrio térmico com o meio [29].

#### 2.1.1 Termorregulação Humana

Como mencionado anteriormente, o ser humano é um animal homeotérmico, portanto possui um mecanismo termorregulador para controlar as variações na temperatura do corpo. Sendo

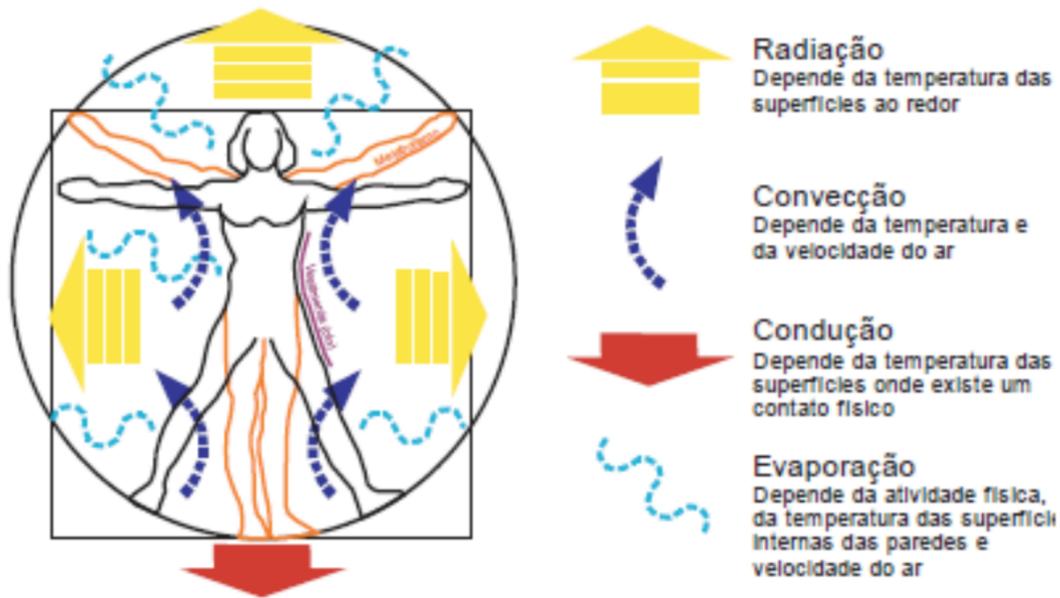


Figura 2.1: Esquema da Fisiologia Humana e suas Trocas Térmicas [1].

assim, o corpo humano pode ser aproximado por uma máquina térmica.

Como uma máquina térmica, o corpo humano precisa de um calor mínimo para conseguir produzir mais calor. Esse calor mínimo necessário é obtido por meio da metabolização dos alimentos ingeridos.

Parte do calor produzido é utilizado de forma a garantir bom funcionamento fisiológico do corpo humano. Enquanto que a outra parte é dissipada para que o corpo humano não superaqueça.

A troca de calor entre o corpo humano e o meio ambiente pode acontecer de quatro formas distintas: a radiação, a convecção, a condução e a evaporação. A figura 2.1 representa de maneira esquemática essas quatro formas de troca de calor associado a fisiologia do ser humano.

### 2.1.2 Índice de Conforto Térmico: PMV

Com base em um estudo feito com 1296 estudantes, [30] propôs dois índices para avaliar o conforto térmico de um dado ambiente [29]. Os índices propostos foram o PMV (voto médio previsível) e o PPD (porcentagem previsível de insatisfeitos).

O índice PMV foi definido como padrão e é usado, por exemplo, nas normas ISO7730 [31] e ASHRAE Standard 55 [18]. Além de ser um índice altamente aceito no meio acadêmico.

O índice PMV varia entre -3 e 3. Um resultado igual a -3 significa um ambiente extremamente frio, enquanto que um resultado igual a 3 significa um ambiente extremamente quente. Já o zero equivale ao equilíbrio ideal de conforto térmico [10]. Para calcular o índice PMV, pode-se utilizar a equação (2.1):

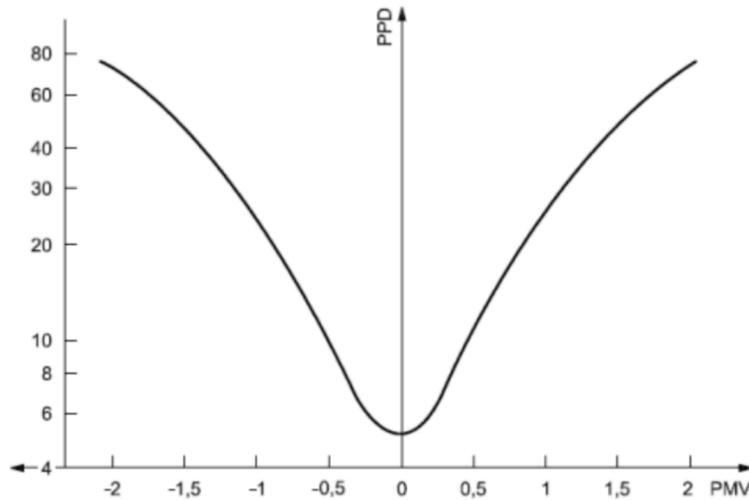


Figura 2.2: Relação entre o PPD e o PMV [1].

Tabela 2.1: Padrão de Temperatura ASHRAE [18].

Estação	Temperatura Operacional	Faixa Aceitável
Verão	22 °C	20 - 23 °C
Inverno	24.4 °C	22 - 26 °C

$$PMV = 0.303 \cdot (e^{-0.036 \cdot M} + 0.028) \cdot L, \quad (2.1)$$

onde  $M$  é a taxa de produção metabólica, em  $W/m^2$ , e  $L$  é a carga térmica atuante sobre o corpo do indivíduo, em  $W$ . Outro método para obter o índice PMV é consultando as tabelas disponíveis no anexo E da norma ISO7730 [31].

O PPD, que representa o número estimado de pessoas insatisfeitas termicamente com o ambiente, é calculado a partir do PMV. O cálculo do PPD pode ser feito utilizando a equação (2.2):

$$PPD = 100 - 95 \cdot e^{-0.03353 \cdot PMV^4 - 0.2179 \cdot PMV^2}. \quad (2.2)$$

Já a figura 2.2 mostra a relação entre os índices PPD e PMV. Ao analisá-lo vemos que para valores nas extremidades do índice PMV, o número estimado de pessoas insatisfeitas aumenta. Enquanto que ao se aproximar do zero, o resultado dessa estimativa diminui.

É impossível projetar ambientes que agradem a todas as pessoas sempre. Então é plausível especificar como um ambiente aceitável termicamente um ambiente que agrade uma grande parcela das pessoas. A norma ASHRAE Standard 55 [18] define temperaturas recomendáveis para o conforto térmico, porém desconsidera informações relevantes como a localização geográfica.

Neste trabalho, é utilizado como temperatura operacional um valor intermediário entre as faixas aceitáveis definida pela norma [18] para o verão e o inverno, ou seja, 23 °C foi a temperatura base escolhida para o conforto térmico. Outro fator determinante para escolha dessa temperatura

é a grande quantidade de trabalhos desenvolvidos anteriormente no mesmo ambiente considerando como base 23 °C. Escolhendo a mesma temperatura, torna-se mais fácil comparar os resultados do trabalho proposto com os resultados de projetos passados.

Além disso, utilizando as tabelas mostradas nos anexos A, B e E da norma ISO7730 [31], é possível concluir também que a temperatura de referência deve ser 23 °C. Isso considerando como referência pessoas realizando atividades sedentárias como em um escritório, escola ou laboratório. Esse tipo de atividade equivale a uma taxa metabólica de aproximadamente  $70 \text{ W/m}^2$ . Pode-se assumir, também, um isolamento das roupas igual a 0,75 *clo* que corresponde a uma vestimenta casual incluindo calça, blusa, meia e tênis. Ao analisar um ambiente com tais características, vê-se que a temperatura definida de 23 °C possui um PMV próximo ao zero.

Segundo [15], a temperatura é a variável essencial para o conforto térmico. Além disso, a presença de um ser humano assim como suas trocas complexas de calor podem ser tratados e modelados como uma perturbação para a temperatura ambiente.

## 2.2 Carga Térmica

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 [32]. Essa carga térmica pode ter diferentes fontes, como por exemplo:

- condução;
- insolação;
- dutos;
- pessoas e
- ventilação.

### 2.2.1 Carga Devido às 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. Um pessoa possui uma temperatura média de 37°C. 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.

Segundo [32], uma pessoa sentada ou em movimento lento gasta aproximadamente  $100 \text{ kcal/h}$ ,  $69,1 \text{ kcal/h}$  de calor sensível e  $31 \text{ kcal/h}$  de calor latente, em um ambiente com temperatura  $23^\circ\text{C}$ . 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.

## 2.3 Visão Computacional

Sabendo que as trocas de calor do ser humano em um dado ambiente podem ser modeladas como uma perturbação, sugere-se fazer um controle que estima o efeito dessa perturbação de forma antecipativa.

Para implementar essa ideia, poder-se-ia utilizar um banco de dados com o peso de cada ocupante e detectar a atividade realizada na sala de reunião. No entanto, por razões de simplicidade serão assumidos valores médios para todos os ocupantes. Portanto, torna-se necessária a utilização de técnicas para a contagem do número de pessoas em um dado ambiente. Essa contagem pode ser estimada por meio de algoritmos de visão computacional.

A contagem do número de pessoas em vídeos é um problema de visão computacional que ainda possui diversos desafios como: variações de iluminação no ambiente, oclusão, diferentes pontos de vista das câmeras e pessoas muito próximas. Esses fatores frequentemente limitam os resultados atingidos, sendo necessária a utilização de diversas técnicas para contorná-los.

Para fazer a contagem, é necessário antes implementar alguma técnica de identificação de pessoas. Embora a identificação de pessoas tenha sido extensivamente explorada recentemente, ainda existe a necessidade de se estudar os métodos propostos de forma a obter um bom desempenho para situações específicas.

### 2.3.1 Histograma de Gradientes Orientados

A utilização de técnicas envolvendo o Histograma de Gradientes Orientados (HOG) para a detecção de pessoas já é uma técnica bem consolidada no meio acadêmico. O trabalho desenvolvido por Dalal e Triggs [2] concluiu que a aparência e a forma dos objetos podem ser caracterizados pela intensidade local e a direção de seus gradientes, mesmo sem uma informação precisa da sua posição.

A figura 2.3 descreve o metodologia utilizada para a identificação de pessoas em uma dada imagem. Inicialmente é feito um pré-processamento e normalização da imagem, para então computar os seus gradientes. A normalização de contraste é feita, então, para diminuir a variância quanto a iluminação e sombras, melhorando assim o resultado obtido. Em seguida, coleta-se o HOG para a dada janela de detecção. O vetor de HOG obtido alimenta um Máquina de Vetores de Suporte (SVM).

O SVM é utilizado para analisar dados e reconhecer padrões a partir de um treinamento supervisionado. É uma técnica muito utilizada para classificação e para análise de regressão. No



Figura 2.3: Cadeia de Identificação de Pessoas [2].

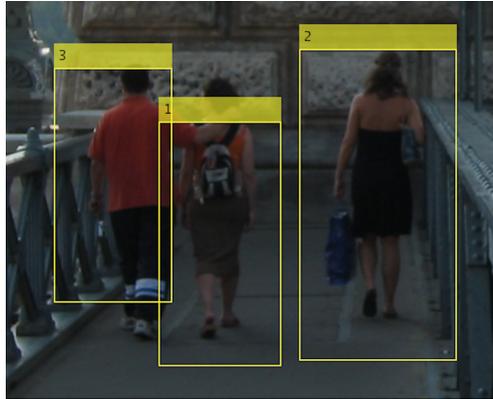


Figura 2.4: Exemplo de Resultado para Detecção de Pessoas a partir do Descritor HOG.

trabalho [2], o SVM recebe o vetor de HOG e o utiliza para classificar o objeto como pessoa ou não.

Um exemplo de resultado obtido através da aplicações do detector HOG para identificação de pessoas em imagens pode ser visto na figura 2.4. Nesse exemplo, pode notar-se que as três pessoas presentes na imagem foram corretamente identificadas.

Por fim, também é importante ressaltar algumas desvantagens desse tipo de método como a alta influência sofrida pelas diferentes poses que as pessoas podem assumir, assim como pelos diferentes posicionamentos da câmera e a alta chance de perda de informação em caso de oclusão.

### 2.3.2 Algoritmo Viola-Jones para Detecção de Faces

O algoritmo Viola-Jones é outro algoritmo de visão computacional fortemente utilizado para a detecção de pessoas, a partir da detecção de faces.

O artigo [3] descreve um método de aprendizagem de máquina para a detecção de objetos rápida e eficaz. Aplica-se um algoritmo de aprendizagem que seleciona um pequeno número de características visuais críticas a partir do uso de classificadores eficazes.

Utiliza-se, em seguida, um método de combinar classificadores complexos em "cascata". Isso permite que as regiões de fundo sejam descartadas e, assim, o recurso computacional é focado em regiões mais promissoras. Essa técnica, então, é aplicada em um sistema de detecção de faces que atinge altas taxas de acerto e que podem ser aplicados em um sistema tempo-real.

Um exemplo de resultado que pode ser obtido através da aplicação desse algoritmo é mostrado na figura 2.5. Nela pode-se ver que o índice de acerto foi alto, porém houveram erros. Um erro claro foi a detecção falso positivo, ou seja, da bola.

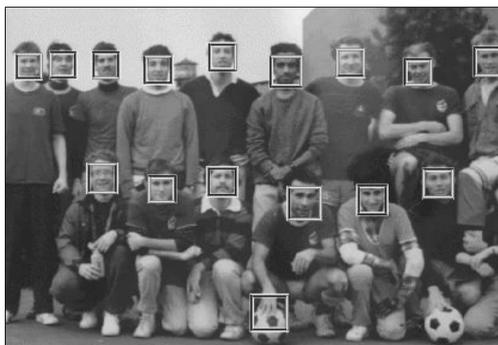
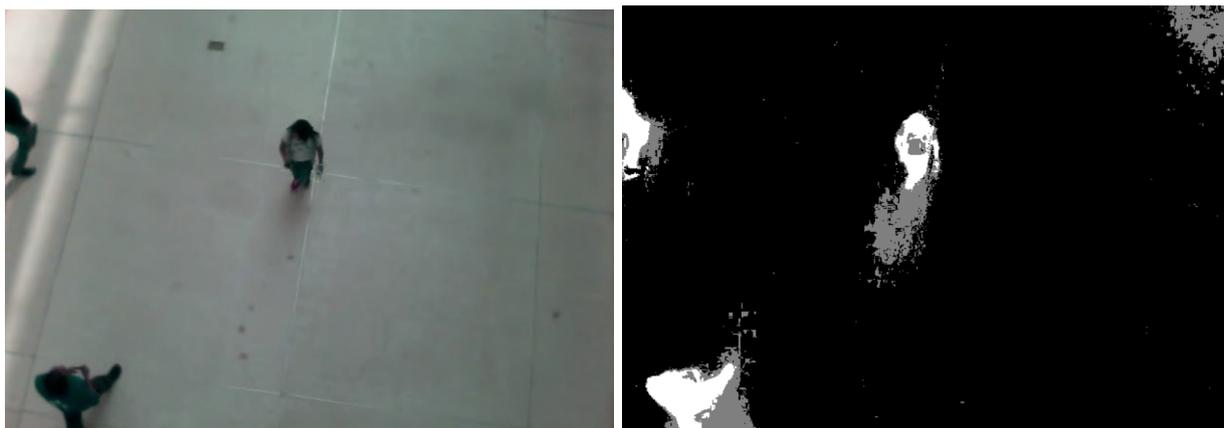


Figura 2.5: Exemplo de Resultado para Aplicação do Algoritmo Viola-Jones [3].



(a) Imagem Original.

(b) Resultado.

Figura 2.6: Exemplo de aplicação do subtrator de fundos em uma imagem [4].

Pode-se também observar algumas desvantagens da aplicação deste algoritmo para a detecção e contagem do número de pessoas. A maior desvantagem é ser focado na detecção de faces. Em uma aplicação real, a imagem pode não ser capaz de englobar a face de todas as pessoas presentes.

### 2.3.3 Subtração de Fundo

Um subtrator de fundo permite distinguir o plano de fundo do primeiro plano de uma imagem. Isso considerando que o plano de fundo é a parte estática, ou seja, tudo que permanece constante em uma série de imagens. Já o primeiro plano é considerado tudo que se move.

Pode-se ver o efeito de um subtrator de fundos comparando as figuras 2.6(a) e 2.6(b). A figura 2.6(a) é a imagem original, enquanto que a figura 2.6(b) é após a aplicação da técnica. Pode-se ver que o plano de fundo é representado pela cor preta, o primeiro plano é representado pela cor branca e a parte cinza, por sua vez, é a sombra deixada pelos objetos em movimento.

A partir dessa informação de movimentação é possível definir e contar as pessoas em um determinado ambiente.

## 2.4 Identificação do Sistema

Para que novas técnicas de controle sejam implementadas para o sistema em questão, é necessário antes saber o comportamento real do sistema e tão importante quanto, a função que representa esse funcionamento, ou ao menos uma aproximação adequada. Com isso em mente, é necessário fazer a identificação do sistema. A ideia geral para a identificação de sistemas dinâmicos de 1º ou 2º ordem é que por meio de uma resposta ao degrau, verifique-se o comportamento da curva. Assim, torna-se possível estimar a função de transferência que leva àquela resposta.

Com a função de transferência da planta em mãos, é possível, além de realizar simulações, observar a resposta em malha aberta e projetar controladores adequados para levar a uma determinada resposta de interesse.

## 2.5 Controle

Para o controle de temperatura é importante uma resposta compatível com a dinâmica do sistema térmico, que é lenta. O conhecimento das técnicas é importante para estabelecer um resultado esperado adequado. Com a técnica de implementação abordada a seguir, pode-se implementar as ações de controle seguindo os conceitos abordados, de modo a ajustar a potência do compressor para atingir a referência estipulada.

Foram utilizadas diferentes técnicas de controle e de implementação, incluindo uma abordagem clássica.

### 2.5.1 Controle Liga-Desliga

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.

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].

Outra utilização do controlador é em sistemas com histerese, conforme visto na figura 2.7.

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

Um exemplo de onda deste tipo de controlador pode ser visto na figura 2.8.

Note que o sinal de controle fica constantemente ligando e desligando, daí o nome do controlador. No entanto, esse tipo de acionamento acaba por gerar um grande consumo energético e um conforto térmico menos adequado, uma vez que a temperatura oscila entre uma faixa de temperatura aceitável.

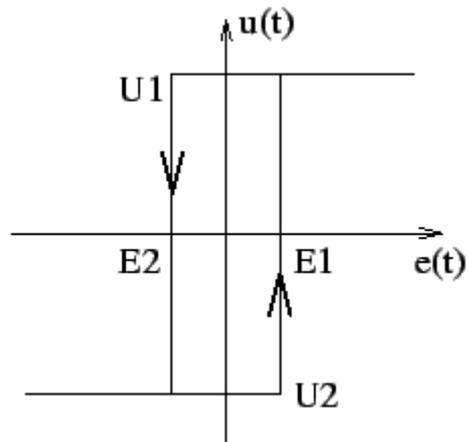


Figura 2.7: Exemplo de histerese [5].

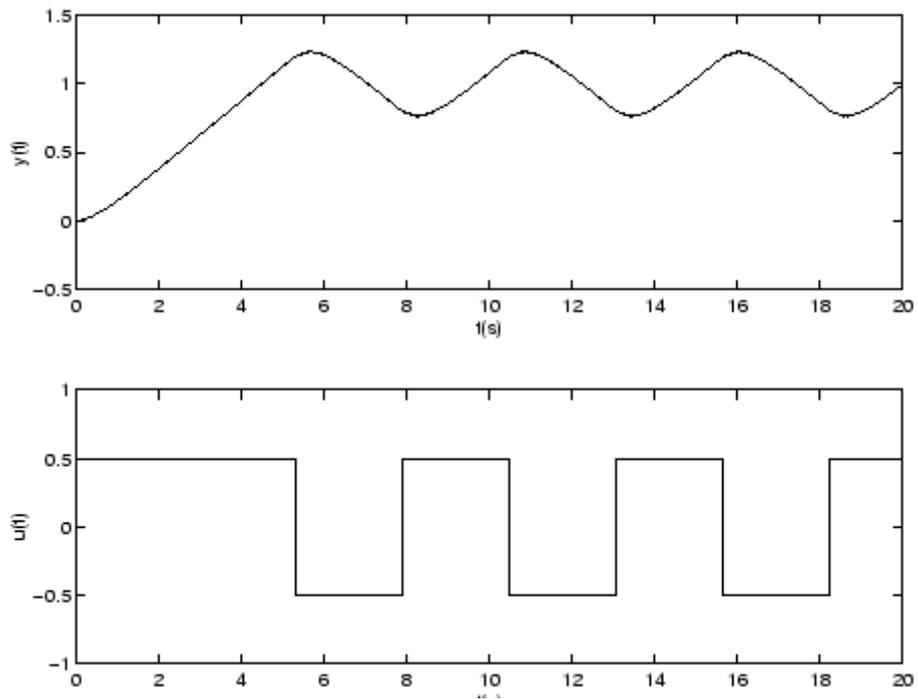


Figura 2.8: Exemplo de ondas referentes ao controlador Liga-Desliga [5].

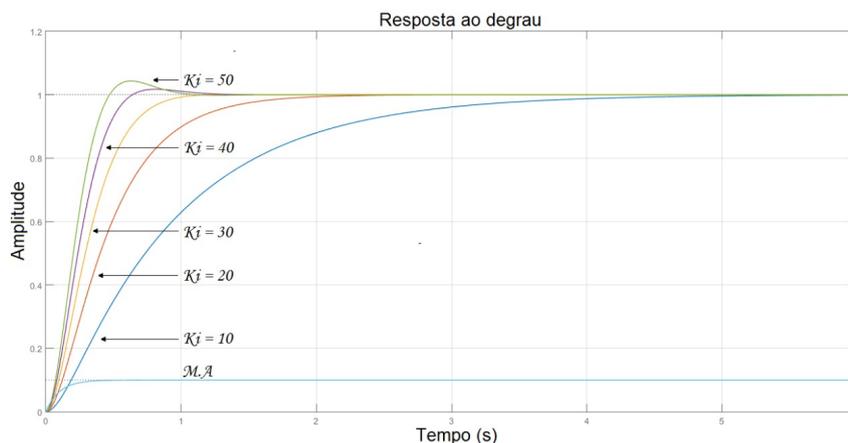


Figura 2.9: Exemplo de ondas referentes ao controlador PI [6]

### 2.5.2 Controle PI

De modo a substituir o tradicional controlador liga-desliga, utiliza-se um controlador mais eficiente, capaz de fornecer melhores respostas e menores erros, para isso, propõe-se a utilização de um controlador proporcional-integral. Esse tipo de controlador aumenta o tipo do sistema graças ao canal integral e faz com que sistemas do tipo 0 consigam seguir um sinal de referência do tipo degrau com erro zero. Contudo, a ação integral por si só pode piorar a estabilidade relativa do sistema, logo utiliza-se em conjunto, um ajuste proporcional do ganho. Com isso, a lei de controle, é dada pela equação (2.3):

$$u(t) = K(e(t) + \frac{1}{T_i} \int_0^t e(\tau)). \quad (2.3)$$

Com o controlador PI, tem-se uma resposta similar à da figura 2.9.

Com um projeto adequado, espera-se de um controlador PI uma resposta que acompanhe a referência, sem grandes oscilações e de forma rápida. Além de garantir um conforto térmico mais adequado, gera economia de energia.

### 2.5.3 Controle Antecipativo

O controlador antecipativo, também conhecido como controle *feedforward*, tem como intuito o tratamento de perturbações de modo que estas não influenciem o resultado do processo. Em técnicas de controle por realimentação negativa, a medida que distúrbios  $w(t)$  diferentes entram no processo, a variável controlada  $c(t)$  desvia-se do ponto fixo e a realimentação negativa é compensada com base na variável de erro  $e(t)$ . A vantagem do controle por realimentação é a sua simplicidade, porém, sua desvantagem é que para rejeitar perturbações, a saída do sistema deve antes divergir da referência, caracterizando-se por uma ação corretiva.

Já o controle *feedforward*, tem como conceito base a rejeição de perturbações antes de estas afetarem a variável controlada. Especificamente, para o controle antecipativo requer uma medição

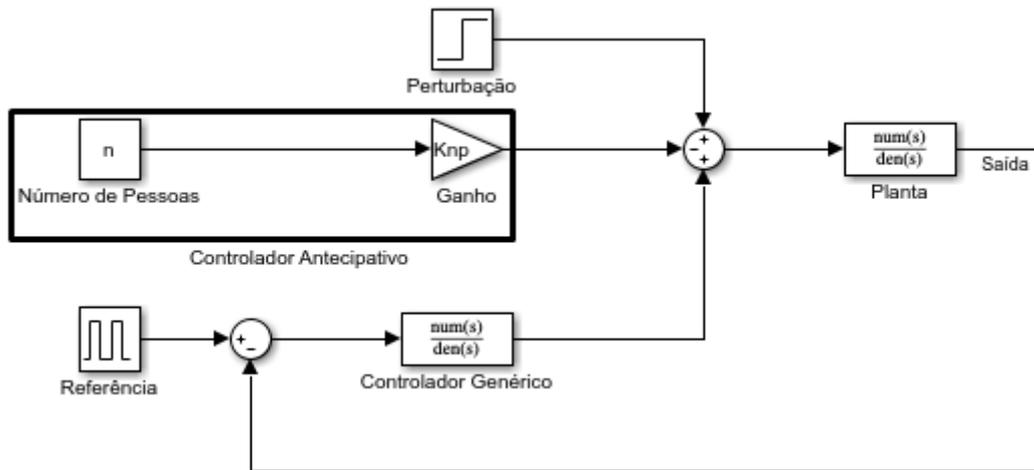


Figura 2.10: Diagrama de blocos de um sistema com Controle Antecipativo.

ou estimação dos distúrbios, antes que estes entrem na planta e, com base nesse cálculo, compensa-se a planta com esse valor negativamente de modo a cancelar a influência da perturbação, ou seja, caracterizando-se por uma ação de controle antecipativa. Essa abordagem é muito útil na área de controle de processos, que por lidar com sistemas de fluidos, por vezes apresenta uma dinâmica lenta e, que sob efeito de perturbações pode gerar maiores erros durante um longo período de tempo [33].

Uma diagrama de blocos que representa uma malha de controle com um controlador antecipativo pode ser observado na figura 2.10.

O controlador antecipativo quando comparado a um controlador por realimentação possui a mesma resposta para casos sem perturbação, a grande diferença, contudo, está no modo como as perturbações são rejeitadas. Na figura 2.11, é possível verificar a grande diferença do comportamento entre os dois tipos de controladores [34], o caso em questão aborda uma situação de dinâmica térmica em níveis de líquidos. Ambos os controladores são capazes de fornecer a mesma resposta, contudo, o controlador antecipativo se mostra superior quando lidando com perturbações.

Do ponto de vista de um sistema térmico, a mesma lógica pode ser aplicada, já que o sistema térmico também possui uma dinâmica lenta. Com a informação adquirida em tempo real, o Controlador Antecipativo é capaz de ajustar a potência do compressor de modo a garantir que a temperatura se mantenha na referência, mesmo com uma perturbação incidente na planta. Vale ressaltar que o controle ainda depende de um controlador PI, contudo, com a compensação antecipativa dos distúrbios.

#### 2.5.4 Modulação de Largura de Pulso - PWM

Modulação por Largura de Pulso é a base para o controle em sistemas de eletrônica de potência. O conceito teórico de tempo de subida e tempo de descida nulos representa uma maneira moderna de controlar dispositivos de energia.

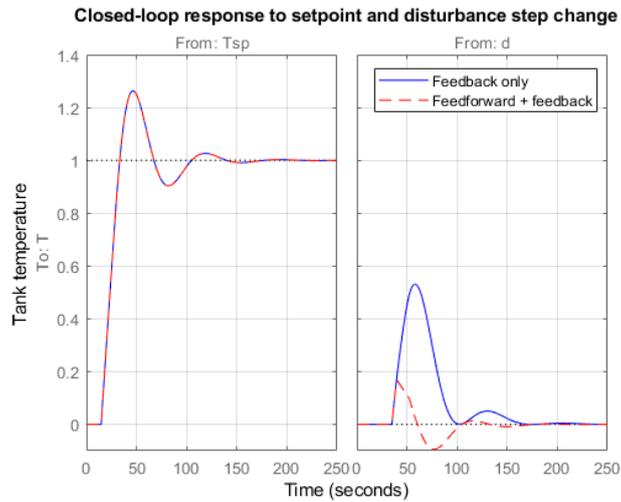


Figura 2.11: Comparação entre respostas com e sem perturbações para os controladores por realimentação e controle antecipativo.

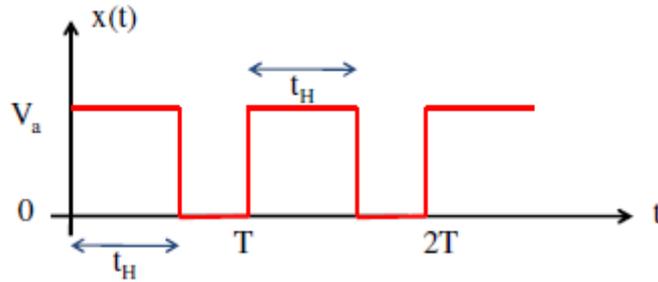


Figura 2.12: Exemplo de onda PWM.

Com exceção de alguns conversores ressonantes, a grande maioria dos circuitos eletrônicos de potência é controlada por sinais PWM de várias formas. As bordas de subida e descida rápidas garantem que os dispositivos semicondutores de potência sejam ligados ou desligados o mais rápido possível para minimizar o tempo de transição de comutação e as perdas de comutação associadas, se mostrando uma técnica eficaz e com baixas perdas de potência durante a transmissão [10].

A técnica do PWM é feita variando-se a largura do pulso e também o intervalo de modo a gerar ciclos ativos diferentes, sendo possível controlar a potência média aplicada a uma carga. Um sinal PWM de frequência constante pode ser reproduzido simplesmente realizando uma comparação entre o sinal de referência  $r(t)$  e o sinal de saída  $c(t)$ . A saída PWM pode ser representada de acordo com a seguinte equação:

$$b_{pwm}(t) = \text{sgn}[r(t) - c(t)], \quad (2.4)$$

sendo  $\text{sgn}$  o sinal da função [35]. Um exemplo de PWM pode ser visto na figura 2.12.

Note que com base nessa figura, o ciclo de trabalho (*duty cycle w*) é dado por:

$$w = \frac{t_H}{T} 100\%. \quad (2.5)$$

Dado o caráter periódico do sinal, este pode ser decomposto em uma série de Fourier. Sendo a componente DC representada por:

$$DC = \frac{1}{T} \int_0^T x(t) dt, \quad (2.6)$$

enquanto a segunda componente principal é a componente fundamental e é dada por:

$$freq_{fundamental} = \frac{1}{T} Hz. \quad (2.7)$$

A componente DC varia de 0 a  $V_A$  conforme o ciclo de trabalho varia de 0 a 100%.

Como diversos motores têm um comportamento de um filtro passa-baixas, caso estes sejam excitados por um sinal PWM, tal que  $\frac{1}{T} \gg \frac{1}{\tau}$ , sendo  $\tau$  a constante de tempo do atuador, graças ao caráter passa-baixas, a velocidade será resultado da componente DC do sinal PWM. E, como a velocidade pode ser dada através do valor médio do sinal PWM, pode-se ajustar esta variando o ciclo de trabalho do sinal [36].

## 2.6 Trabalhos Anteriores

Já foram desenvolvidos outros trabalhos anteriormente visando a climatização da sala de reuniões do LARA. Pode-se citar a contribuição de Heyder Araújo e Marcella Melo [10] que desenvolveram um sistema de controle adaptativo da temperatura da sala de reuniões, utilizando informações como as temperaturas das salas vizinhas para melhorar a resposta do sistema.

Outros exemplos são o trabalho de Daniel Boaventura [37] e Artur Araújo [14] que utilizaram sistemas embarcados para fazer a automação do sistema de climatização. O Daniel Boaventura foi o primeiro a utilizar o OpenHab para fazer o supervisor dessa automação. Enquanto que o Artur Araújo implementou um controle TITO (duas entradas e duas saídas) no sistema de climatização híbrido conseguindo controlar duas variáveis: a temperatura e a umidade.

Todos esses projetos possuem um objetivo em comum: melhorar a eficiência energética dos sistemas de climatização, sem diminuir o conforto térmico dos seres humanos usuários do sistema. Demonstrando que podem ser exploradas as mais diversas técnicas de controle para alcançar esse objetivo.

Esses projetos atingiram resultados positivos e contribuíram como uma base fundamental para esse trabalho. Porém, todos eles obtiveram resultados muito sensíveis à perturbações térmicas causadas por mudanças bruscas no número de pessoas na sala. Ao entrar ou sair pessoas do ambiente, as respostas dos sistemas são degradadas e demoram a melhorar. Por isso, esse trabalho propõe um controle antecipativo que diminua o efeito dessas perturbações na variável controlada, ou seja, na temperatura.

O próximo capítulo cita os materiais, ferramentas e *softwares* utilizados para o desenvolvimento do projeto. Além de ressaltar o objetivo, o modo de funcionamento e as especificações de cada

um.

## Capítulo 3

# Materiais e Ferramentas

### 3.1 *Raspberry Pi*

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 3 Model B* mostrado na figura 3.1. Suas especificações gerais são:

- Processador: *Quad Core 1.2 GHz Broadcom BCM2837 64 bit CPU*;
- Alimentação: *Socket Micro USB 5V 2,5A*;
- 1 GB de memória *RAM*;
- 4 portas *USB 2.0*;
- 15 pinos *MIPI* interface serial da câmera (*CSI 2*);
- *HDMI* para saída de vídeo;
- Porta micro *SD* para carregar o Sistema Operacional (SO) e salvar os dados;
- Dimensões: 85 x 56 x 17 mm.

Para esse projeto, o sistema operacional utilizado no *Raspberry Pi* foi o *Raspbian* versão 4.14. O *Raspbian* é um sistema operacional gratuito baseado no *Debian* e otimizado para a arquitetura de *hardware* do *Raspberry Pi*.

O *Raspberry Pi* é utilizado como o ponto central do sistema de automação projetado.



Figura 3.1: *Raspberry Pi 3 Model B* [7].

## 3.2 Câmera

Assim como o *Raspberry Pi*, a câmera utilizada no projeto é um produto oficial da Fundação *Raspberry Pi*. Mais especificamente, utilizou-se o Módulo Câmera *Raspberry Pi* v2 de 8-megapixel e pode ser vista na figura 3.2. Algumas especificações relevantes para o projeto são:

- Sensor: Sony IMX219;
- Resolução: 8MP;
- Imagem:  $3280 \times 2464$  pixels;
- Vídeo: 1080p30, 720p60 e 640x480p60/90;
- Distância focal: 3.04 mm;
- Raio focal: 2,0;
- Campo de visão horizontal de  $62,2^\circ$ ;
- Campo de visão vertical de  $48,8^\circ$ ;
- Formatos de imagem: JPEG (*accelerated*), JPEG + RAW, GIF, BMP, PNG, YUV420, RGB888;
- Formato de vídeo: raw h.264 (*accelerated*);
- Dimensões: 25 x 24 x 9mm;
- Comunicação via CSI 2.

Sua conexão com o *Raspberry Pi* é simples e prática, visto que o *Raspberry Pi* utilizado já possui o conector CSI 2 próprio para essa troca de dados. Além disso, o sistema operacional *Raspbian* permite a comunicação com a câmera sem a necessidade de instalar *softwares* adicionais.

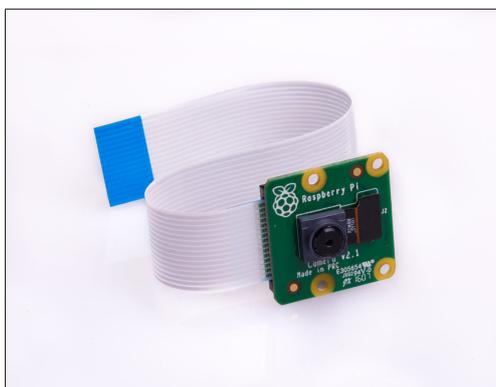


Figura 3.2: Módulo Câmera *Raspberry Pi* v2 [8].



Figura 3.3: *Arduino Mega* [9].

### 3.3 *Arduino*

O *Arduino* é uma placa de prototipagem de *hardware* livre e de placa única. A placa trabalha com um microcontrolador do tipo *Atmel AVR*. O *hardware* dá suporte a diversas formas de entrada e saída de dados de uma maneira simples e amigável. As capacidades de memória e de processamento variam de acordo com o modelo escolhido.

A linguagem utilizada é programada através de um ambiente de desenvolvimento integrado (*IDE*) próprio do *Arduino* e é baseada no *Wiring*, uma variação de *C/C++*. A placa conta com um *bootloader*, que permite a gravação do programa no *chip* de memória.

A figura 3.3 mostra o *Arduino Mega*, esta placa foi a escolhida para o projeto dado o poder computacional com uma maior capacidade de processamento do que sua versão Uno, de memória, além de um número maior de portas digitais e analógicas. A seguir é mostrado algumas especificações relevantes da placa.

- Processador: *ATmega 2560*;
- Memória *flash* 256 KB
- *EEPROM* 4 KB
- Memória *SRAM* 8 KB

- Alimentação 7-12V 1A;
- 54 pinos digitais, onde 14 podem ser *PWM*;
- 16 pinos analógicos;
- 4 *UARTs*;
- Cristal de 16 MHz;
- Porta *USB 2.0*;
- Conector *ICSP*;
- Dimensões: 101,6 x 53,4 mm.

Neste trabalho o *Arduino* foi utilizado como entrada para a leitura de dados dos sensores e para acionamento dos atuadores.

### 3.4 Módulos Relé de Estado Sólido

Para o acionamento de atuadores, uma boa forma é a utilização de chaves estáticas. Estas conseguem prover para cargas que exijam maior potência o chaveamento frente a altas correntes. Nesse sentido, a utilização de um relé se mostra bastante comum.

Comumente, existem dois tipos de relés: os eletromecânicos (*EMR*) e os de estado sólido (*SSR*) [38]. Os *SSRs* estão cada vez mais ganhando espaço, visto que são componentes mais robustos, livres de acionamento mecânico e imunes a ruído. No entanto, a utilização de *SSRs* se mostra inviável para alguns casos.

Em um *SSR*, o eletroímã é substituído por um *LED* que é interligado ao receptor óptico e ao receber o estímulo elétrico, faz com que uma pastilha de cristal semicondutora passe a conduzir, controlando um dispositivo de potência como, por exemplo: um transistor de potência, um *SCR*, um *TRIAC*, entre outros. O funcionamento do *SSR*, pode ser visto na figura 3.4.

Seu diferencial é não possuir componentes mecânicos ou móveis integrados, consequentemente não há desgaste por atrito, resultando em uma maior vida útil. Assim como os relés tradicionais, são capazes de chavear altas correntes e tensões, podendo ser DC ou AC.

Para a aplicação desejada, utilizou-se o módulo relé *SSR* de 4 canais, que pode ser visto na figura 3.5. As principais especificações são:

- Alimentação individual de cada canal: 5 V 12,5 mA;
- Saída: Suporte de 240 VAC , 2 A;
- Dimensões: 25 x 34 x 25 mm.

A ideia é que a ativação do relé via *Arduino* a partir do sinal de controle acione o compressor do ar-condicionado para o controle da temperatura.

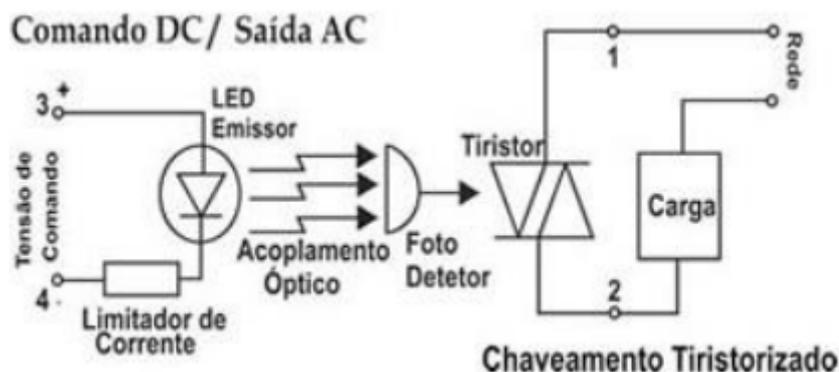


Figura 3.4: Circuito interno de um *SSR* [10].

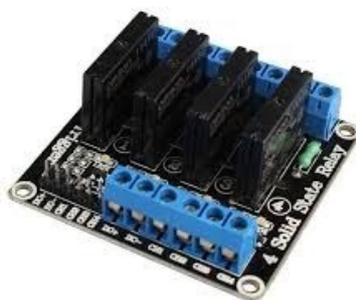


Figura 3.5: Módulo relé *SSR* de 4 canais.

### 3.5 Sensor de Temperatura e Umidade SHT71

O *SHT71*, mostrado na figura 3.6, é um sensor de temperatura e umidade desenvolvido pela empresa *Sensirion*. Ele integra os elementos sensores com elementos processadores de sinal e, assim, permite uma saída de alta potência. Como resultado, obtém-se uma saída confiável, tempo de resposta rápido, além de ser insensível a interferências durante o envio. Além disso, o sensor conta com um módulo de memória de calibração já configurado de fábrica.

O sensor conta com um elemento sensor capacitivo para medição da umidade relativa. Além de um sensor de *gap* de energia para medição da temperatura. Dentre as suas especificações gerais tem-se:

- Tensão de alimentação: entre 2,4V e 5,5V;
- Resolução do conversor analógico/digital (A/D): configurável podendo variar de 8 bits à 14 bits.

Além dessas especificações gerais do sensor também é importante verificar as informações relativas à medição de temperatura. Essas informações estão disponíveis na tabela 3.1. Nela podemos ver os dados de resolução, acurácia, repetibilidade, faixa de operação e tempo de resposta.

O sensor *SHT71* possui, ainda, um circuito para a interface serial *I<sup>2</sup>C* entre ele e o micro-

Tabela 3.1: Especificações de Temperatura do Sensor *SHT71*.

Parâmetro	Mínimo	Típico	Máximo	Unidade
Resolução	0,04	0,01	0,01	°C
	12	14	14	bit
Acurácia	-	±0.4	-	°C
Repetibilidade	-	±0.1	-	°C
Faixa de Operação	-40	-	123,8	°C
Tempo de Resposta	5		30	s

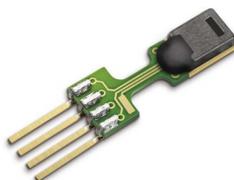


Figura 3.6: Sensor de temperatura e umidade *SHT71* [11].

controlador utilizado para leitura dos dados, o *Arduino*. Esse circuito pode ser visto na figura 3.7.

Foi utilizado somente um sensor *SHT71*: para medir a temperatura e umidade no interior da sala de reuniões do LARA.

### 3.6 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 principais do sensor estão disponíveis na tabela 3.2. Pode-se ver que a sua tensão de alimentação, possibilita que ele seja utilizado e alimentado pelo *Arduino*. Outro fator determinante para sua escolha é a sua boa distância de transmissão de dados que atinge cerca de

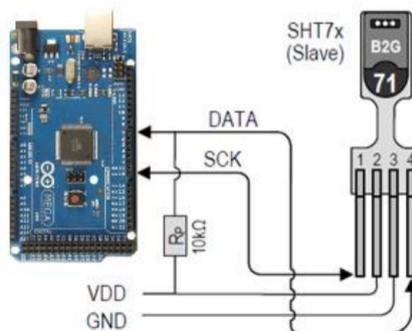


Figura 3.7: Circuito para a interface serial entre o sensor e o *Arduino* [10].

Tabela 3.2: Especificações do Sensor *DHT22*. [19]

<b>Tensão de Alimentação</b>		3,3 ~6 V
<b>Sinal de Saída</b>		Digital
<b>Faixa de Operação</b>	<b>Umidade</b>	0 ~100% RH
	<b>Temperatura</b>	-40 ~80 °C
<b>Acurácia</b>	<b>Umidade</b>	± 2% RH
	<b>Temperatura</b>	± 0.5 °C
<b>Período Médio</b>		2 s

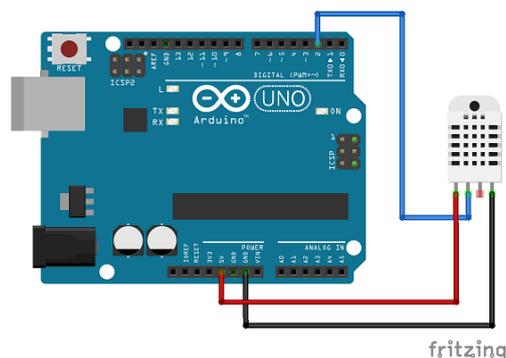


Figura 3.8: Diagrama para conexão do sensor *DHT22* com o *Arduino* [12].

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.

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.

Na figura 3.8, pode-se ver o diagrama de conexão do sensor com a placa *Arduino*. A conexão é simples, basta conectar o pino 1 (pino de alimentação) à porta *VCC* do *Arduino*, o pino 2 (pino de dados) à uma porta digital e o pino 4 (pino de referência) à porta *GND* do *Arduino*.

Foram utilizados dois sensores *DHT22* no projeto: um instalado na sala vizinha à sala de reuniões do LARA e um externo (fora do prédio SG11).

### 3.7 Módulo Sensor Magnético (*Reed Switch*)

O módulo sensor magnético, mostrado na figura 3.9, possui um *Reed Switch*. Ele funciona como uma chave que abre ou fecha seus contatos conforme a presença ou ausência de um campo magnético.

Esse módulo foi utilizado no projeto para detectar a abertura e o fechamento da porta. Ele foi fixado na parede próximo a um ímã que foi colocado na extremidade da porta. Portanto, quando



Figura 3.9: Sensor Magnético [13].

a porta estiver fechada o ímã estará próximo ao sensor e o circuito será fechado. Já quando a porta estiver aberta, o sinal será baixo.

### 3.8 *OpenCV*

O *OpenCV* é uma biblioteca aberta para aplicações de visão computacional e de aprendizagem de máquina. Esse *software* tem interface com diferentes linguagens de programação como: *C++*, *Python* e *Java* [39]. Uma das grandes vantagens do *OpenCV* é que ele foi desenvolvido com foco na eficiência computacional e, portanto, pode ser utilizado em aplicações tempo-real, ou seja, o código rodaria e responderia a mudanças em curtos intervalos de tempo.

Nesse projeto será utilizada a biblioteca *OpenCV* para linguagem de programação *Python*. O *OpenCV* foi instalado e configurado no *Raspberry Pi* seguindo o procedimento indicado no capítulo II.4 do anexo.

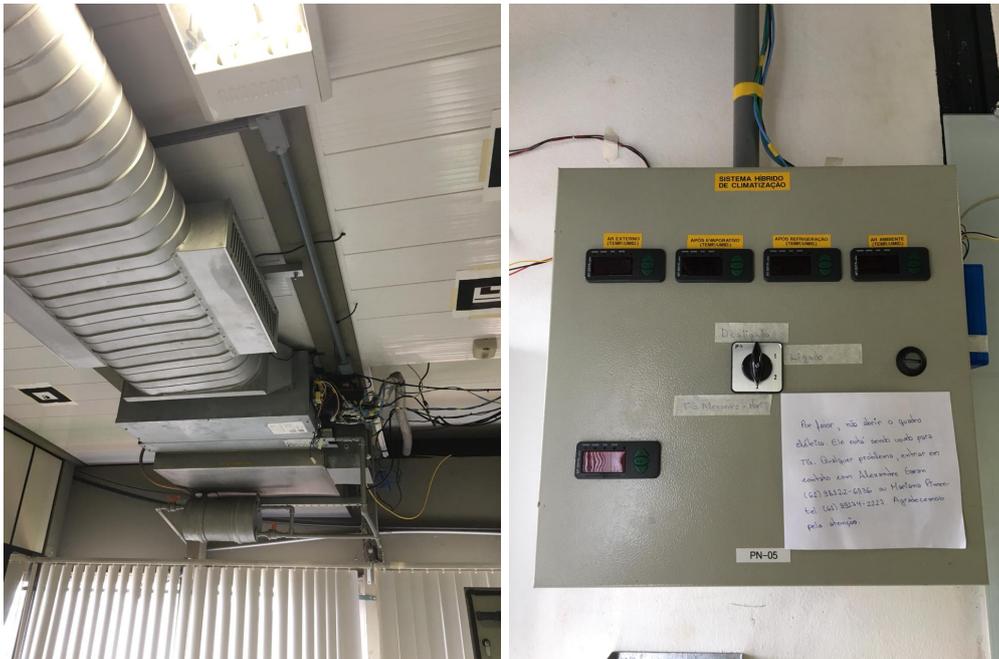
### 3.9 *Matlab e Simulink*

O *software Matlab*, que vem de *Matrix Laboratory*, consiste em um programa amplamente utilizado para operações numéricas, modelagens e simulações de sistemas. 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 as limitações relacionadas ao uso do *Matlab* no *Raspberry Pi*, nesse trabalho o *Matlab* e o *Simulink* foram utilizados apenas para simulação e modelagem do sistema. Depois que foram feitas as simulações, as malhas de controle foram implementadas em *Python* para serem executadas pelo *Raspberry Pi*.

### 3.10 *SQLite*

O *SQLite* é uma biblioteca que implementa um banco de dados embutido. Ao utilizar o *SQLite*, pode-se acessar um banco de dados em linguagem de consulta estruturada (*SQL*) sem a necessidade de executar um sistema de gerenciamento de banco de dados (SGBD).



(a) Ar condicionado da sala de reunião.

(b) Painel de Controle.

Figura 3.10: Sistema de climatização híbrido.

Por ser um banco de dados que oferece maior simplicidade, o *SQLite* é fortemente recomendado para sistemas embarcados onde o recurso é limitado. Por isso, esse foi o banco de dados relacional escolhido para ser utilizado na aplicação. O procedimento de instalação é descrito em detalhes no capítulo II.3 do apêndice.

### 3.11 Sistema de Climatização

Como atuador do sistema, tem-se o sistema de climatização de ar da sala de reuniões do LARA, que será utilizado em seu modelo de operação convencional. O ar-condicionado pode ser visto na figura 3.10(a), enquanto que o painel de controle, pode ser visto na figura 3.10(b).

O funcionamento convencional do sistema de climatização consiste num ciclo de refrigeração fechado onde circula um fluido refrigerante a fim de manter continuamente sua evaporação, baseado na compressão. O ciclo de refrigeração é composto, de forma simplificada, basicamente de um compressor, um condensador, uma válvula de expansão e um evaporador. A figura 3.11 ilustra a estrutura básica de um ciclo de refrigeração.

Pode-se analisar o fluxo começando pelo compressor que succiona o fluido refrigerante à baixa pressão e baixa temperatura e o comprime, elevando assim sua temperatura e pressão. Ao chegar à válvula de expansão, o fluido que na fase de compressão passou de gás para líquido perde pressão e baixa sua temperatura. O fluido, agora à baixa pressão e temperatura, passa pelo evaporador e muda novamente de fase ao retirar calor do ambiente que se deseja resfriar, tornando-se vapor à baixa pressão e temperatura.

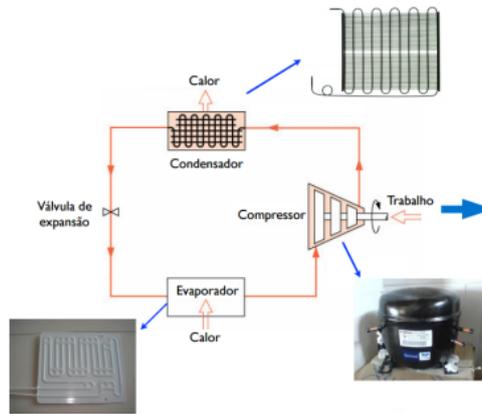


Figura 3.11: Ciclo de refrigeração [14].

Este vapor volta a ser succionado pelo compressor, repetindo-se o ciclo. O compressor é o principal responsável pelo alto consumo de energia elétrica do ar condicionado convencional [14].

### 3.12 Visão Geral

Em resumo, o sistema contará com uma câmera para monitorar o número de pessoas em um dado cômodo a partir de algoritmos de visão computacional implementados utilizando a linguagem de programação *Python* associada a biblioteca *OpenCV*.

Com o número de pessoas, é realizada uma estimativa de carga térmica, que passa a ser uma entrada de perturbação conhecida na malha de controle.

Informações adicionais como a temperatura em pontos estratégicos são obtidas com a utilização de sensores de temperatura *SHT71* e *DHT22*, que enviam dados digitais a um *Arduino Mega*, responsável pela leitura dos dados e acionamento dos atuadores.

O sistema projetado possui uma central de processamento, o *Raspberry Pi*. Esse processador é responsável pelo controle propriamente dito do sistema, que consiste, de forma resumida, em uma compensação da perturbação estimada associado ao clássico controlador PI.

A ação de controle gerada pelo código em *Python* executado no *Raspberry Pi* faz com que o *Arduino* acione um *SSR*, que por sua vez, atua sobre o compressor de ar-condicionado. O controle, na prática se dá através de um sinal *PWM* que ajusta a potência do compressor de modo a garantir maior eficiência energética e conforto térmico.

Os resultados obtidos são salvos no *SQLite*. Esse *software* é o banco de dados selecionado para aplicação.

A figura 3.12 demonstra o diagrama de conexão entre os dispositivos e ferramentas utilizadas.

O próximo capítulo aborda os procedimentos e as metodologias seguidos para a realização do trabalho. Em especial, o estudo sobre o ambiente utilizado, algoritmo de visão computacional para a contagem do número de ocupantes na sala e as técnicas de controle utilizadas.

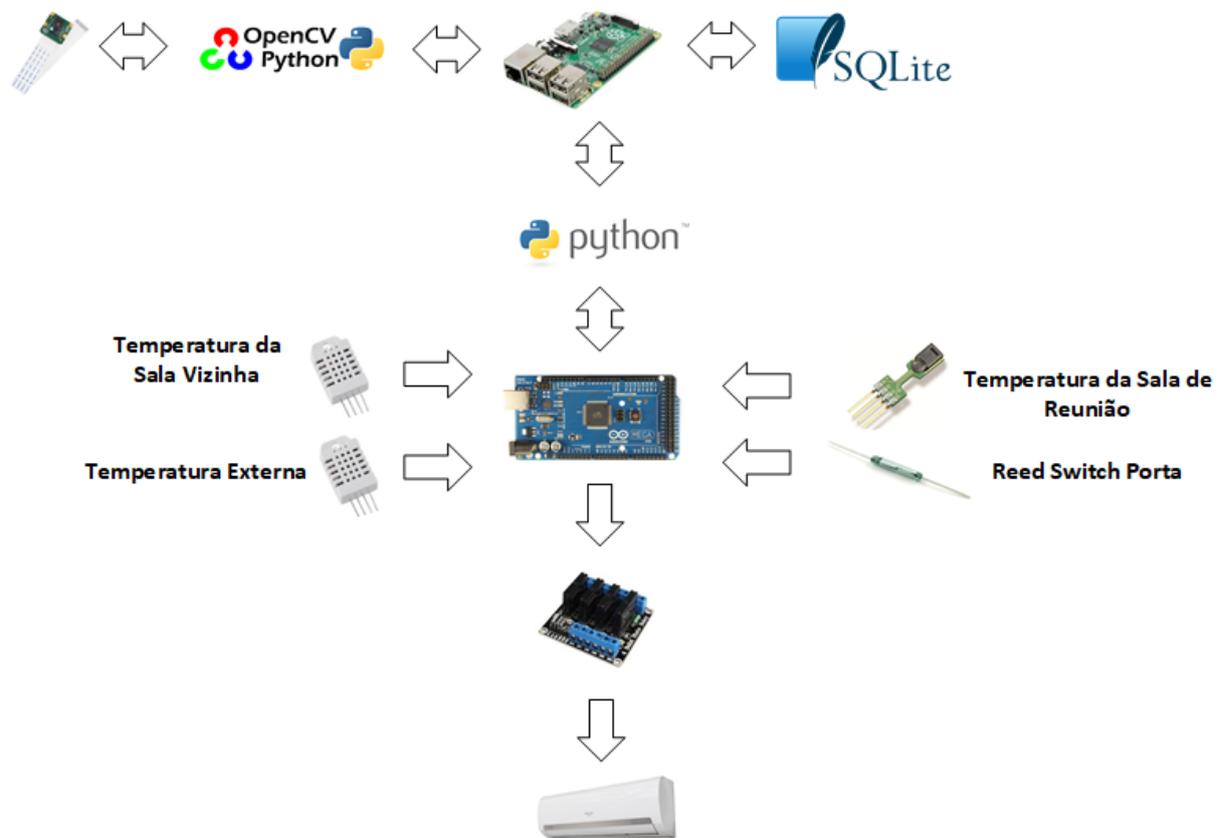


Figura 3.12: Diagrama de Conexão dos Materiais e Ferramentas.

# Capítulo 4

## Métodos

### 4.1 Estudo do Ambiente

O ambiente de trabalho selecionado para a realização do trabalho foi a sala de reunião do LARA. O LARA é dividido em seis ambientes e o ambiente utilizado pode ser melhor visualizado na figura 4.1.

Já o ambiente real da sala de reunião pode ser visto na figura 4.2, é válido notar que a sala possui uma mesa ampla e comporta confortavelmente, em média, 8 ocupantes.

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, ou seja, não há um bom isolamento térmico.

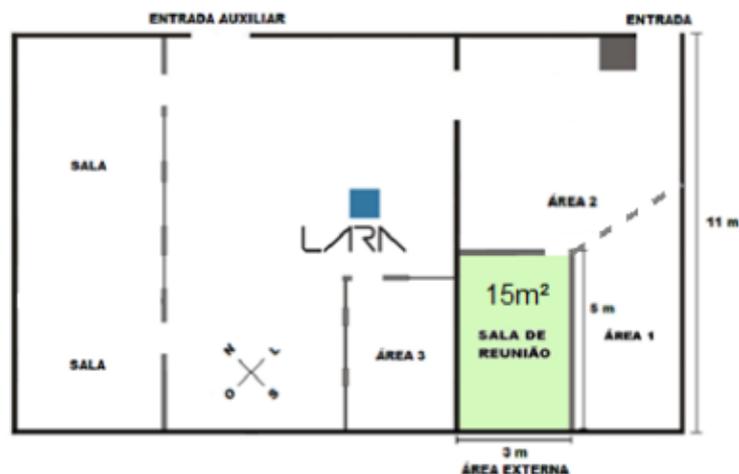


Figura 4.1: Planta do ambiente LARA [15].



Figura 4.2: Ambiente da sala de reunião do LARA.

## 4.2 Identificação de Pessoas

A técnica escolhida para a identificação de pessoas em vídeos é a técnica de subtração de fundo. Por alguns motivos simples que afetariam a resposta dos outros dois métodos propostos em um ambiente como a sala de reunião do LARA.

O primeiro motivo são as mudanças de poses que as pessoas assumem, em uma sala de reunião é esperado que as pessoas estejam em pé ou sentadas, o que pode dificultar a identificação por HOG. Este algoritmo seria mais recomendado em um ambiente em que as pessoas aparecem em pé sempre ou na maior parte do tempo.

O segundo motivo é a alta possibilidade de oclusão no ambiente utilizado. Em uma sala de reunião, tem-se mesas, cadeiras e outros móveis típicos de um ambiente fechado que, frequentemente, podem impedir que a pessoa seja vista por completo dependendo do ângulo da câmera. A oclusão da pessoa poderia, então, afetar o resultado do HOG.

Além disso, uma forma de contornar esse problema seria utilizar uma vista superior, mas isso também afetaria a resposta tanto para HOG quanto para o algoritmo Viola-Jones. O HOG, por trabalhar com a forma das pessoas, apresentaria dificuldades para identificá-las em uma vista superior em que só seria visto o topo das cabeças. Já o Viola-Jones não teria acesso a faces das pessoas, o que é necessário para identificação.

Por esses motivos, escolheu-se trabalhar com uma vista superior aliada a um subtrator de fundo. O algoritmo utilizado para o projeto foi adaptado do trabalho de Federico Mejia [4].

Ao utilizar essa técnica, obtém-se a informação das partes estáticas e em movimento em um sequência de imagens. As partes em movimento são representadas por tons claros, que tendem ao

branco, enquanto que as partes estáticas são representadas pela cor preta.

Em seguida, torna-se necessário fazer algumas transformações morfológicas nos *frames*, antes de partir para a identificação de pessoas. As transformações morfológicas estudadas são conhecidas como erosão, dilatação, abertura e fechamento.

De uma maneira simplificada, a erosão expande a porção preta da imagem em uma porção branca. Já a dilatação expande a porção branca em uma porção preta. A abertura é basicamente uma dilatação após uma erosão. Enquanto que o fechamento é uma erosão após uma dilatação.

Ao utilizar a transformação morfológica após o subtrator de fundo associado a um método para tornar a imagem binária utilizando um dado limite é possível retirar as sombras cinzas da imagem, ou seja, o ruído.

Em seguida, pode-se partir para identificação dos contornos das imagens. O OpenCV conta com funções para fazer essa identificação. Então, é possível desenhar os contornos já obtidos nos *frames* para facilitar a visualização.

O próximo passo é definir se um contorno obtido é ou não é uma pessoa. Uma maneira simples de fazer essa definição é padronizar uma área mínima interior ao contorno para ser considerado uma pessoa.

A próxima etapa é o rastreamento da pessoa. Para fazer isso é necessário atribuir um código identificador (ID) para cada pessoa identificada e armazenar as informações de posição. Para isso foi criada uma classe para representar a pessoa e seus atributos.

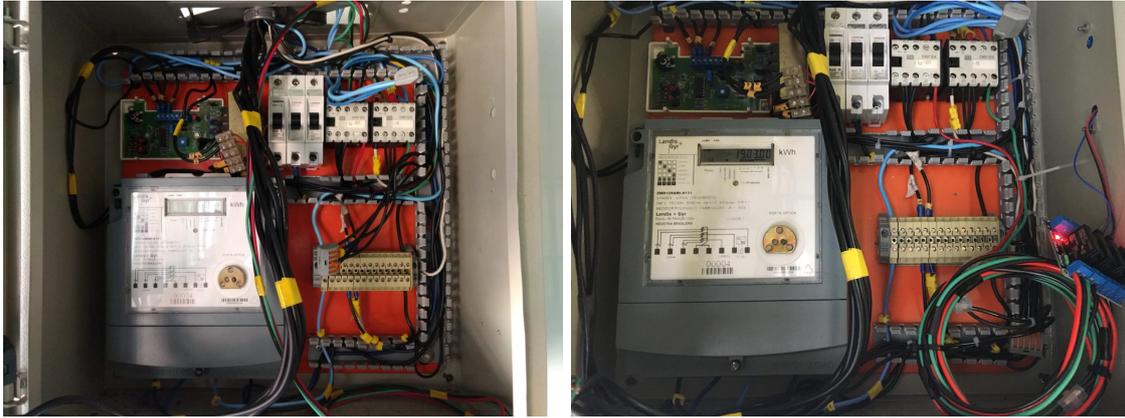
Conforme a pessoa move-se na sala, pode-se obter a informação direção, ou seja, se a pessoa está indo para cima ou para baixo na imagem ao analisar as informações de posição armazenadas.

Agora, pode-se definir se a pessoa está entrando ou saído da sala. Para isso definem-se duas linhas na imagem. Uma linha representando a linha de saída e uma representando a linha de entrada.

Quando a pessoa cruza uma dessas linhas se ela esta na direção correspondente a ação. Por exemplo, se a pessoa está cruzando a linha de saída da sala na direção de saída. Se isso ocorrer o contador deve ser decrementado. Para o caso inverso, a pessoa cruzar a linha de entrada na direção de entrada, deve-se incrementar o contador.

Também pode-se definir mais duas linhas. Essas linhas representam quando o programa pode parar o rastreamento e liberar a memória utilizada.

O algoritmo foi testado, em um primeiro momento, em dois vídeos disponíveis na internet. Após a câmera ser instalada na sala de reunião, passou-se para os testes utilizando um *stream* de vídeo em tempo-real. Para isso, foram necessárias algumas adaptações. Por exemplo, as duas linhas que definem a entrada e saída foram consideradas na mesma altura da imagem. Além disso, a área mínima necessária para ser classificado como uma pessoa também foi ajustada.



(a) Estado do quadro elétrico quando o trabalho foi iniciado. (b) Estado do quadro elétrico com circuito operante.

Figura 4.3: Quadro elétrico do sistema.

### 4.3 Instalação Elétrica

Quando os trabalhos foram inicializados, o quadro elétrico não apresentava uma estrutura clara de instalação e o ar-condicionado se encontrava inoperante, nem mesmo seu modo original de funcionamento estava disponível. O quadro antes das mudanças realizadas pode ser visto na figura 4.3(a).

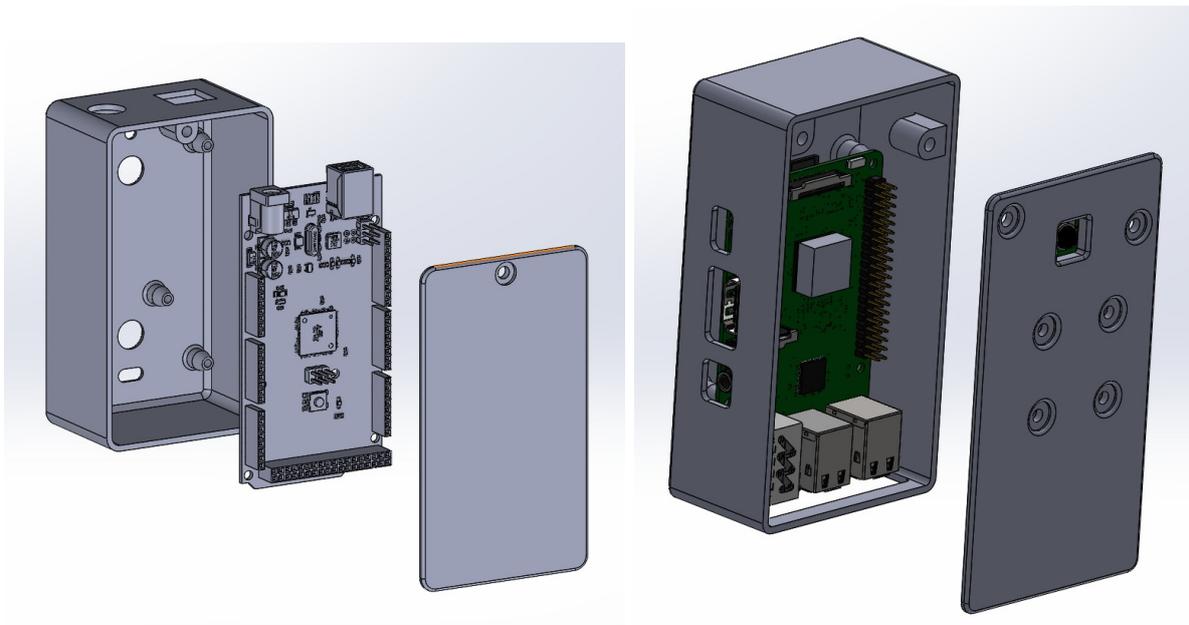
Devido ao estado em que o sistema se encontrava, buscou-se os diagramas elétricos do trabalho original [40] numa tentativa de corrigir as prováveis falhas no sistema.

Contudo, mesmo seguindo-se as especificações de instalação para o quadro elétrico o sistema não só deixou de funcionar, como derrubava o disjuntor sempre que ligado.

Em uma segunda tentativa, após uma conversa com o supervisor técnico do SG-11, percebeu-se que algumas mudanças na ligação com o compressor foram modificadas por alunos que fizeram trabalhos no ar-condicionado anteriormente. Com o diagrama elétrico presente em [41], um dos últimos alunos que trabalharam com o quadro, percebeu-se que alguns dos componentes citados não se encontravam no quadro elétrico, podendo ser a possível causa do não funcionamento.

Uma vez que os componentes foram instalados, o ar-condicionado passou a funcionar normalmente no modo tradicional e pronto para ser implementado o controle antecipativo. Para isso, foram utilizadas diferentes posições da chave seletora presente no quadro. O quadro após as modificações, pode ser visto na figura 4.3(b).

O quadro elétrico conta ainda com um medidor de energia utilizado para a verificação do gasto energético.



(a) Módulo *Arduino*, sensores e relé.

(b) Módulo *Raspberry Pi* e câmera.

Figura 4.4: Modelagem para impressão 3D.

## 4.4 Instalação dos Sensores e Atuadores

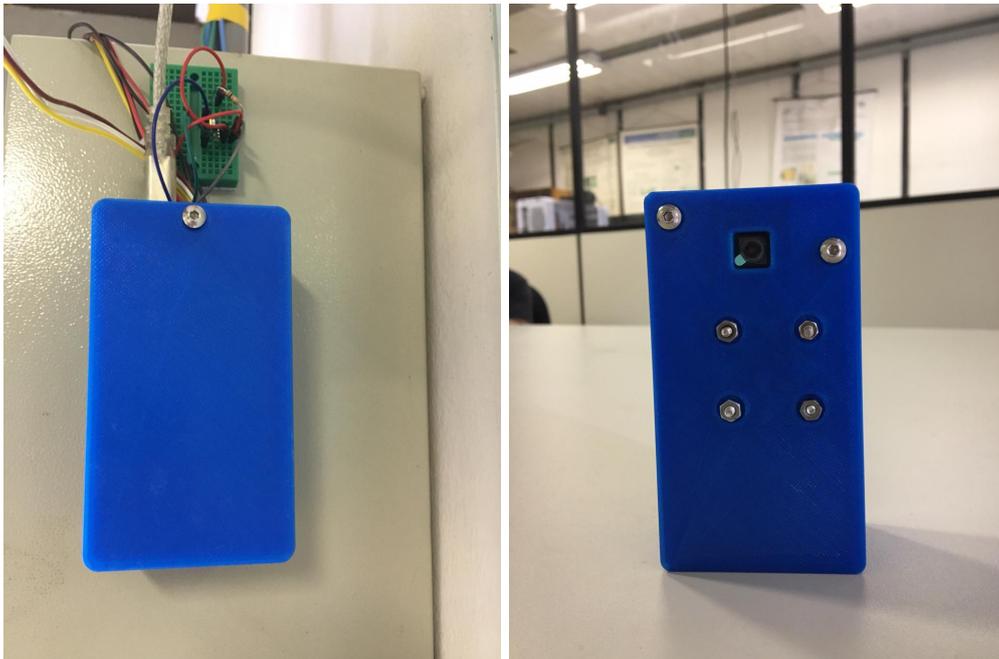
Finalmente, com o sistema plenamente funcional, o passo seguinte consistiu na instalação dos componentes necessários para o funcionamento do controlador proposto. A ideia básica, foi instalar quatro sensores: o primeiro dentro da sala de reunião, o segundo na sala vizinha, o terceiro no lado de fora da janela e o último, na porta.

Dividiu-se os componentes de controle e atuação em dois: o primeiro contendo o *Raspberry Pi* e a câmera e outro contendo o *Arduino*, os sensores e o relé. Em seguida, projetou-se duas caixas para impressão 3D em *Solid Works*, de modo a tornar a instalação mais simples e visualmente mais bonita. Na figura 4.4(a), pode-se ver a modelagem da caixinha para o módulo do *Arduino*, enquanto que na figura 4.4(b), a caixinha para o módulo do *Raspberry Pi*.

Com os modelos prontos, a impressão 3D foi feita e os componentes foram devidamente encapsulados. A figura 4.5(a) mostra a caixa do primeiro módulo devidamente acoplada ao quadro elétrico, enquanto a figura 4.5(b), mostra a caixa para o segundo módulo.

## 4.5 Identificação do Modelo

Para a identificação da planta, inicialmente aplicou-se um degrau de referência durante um período de 3h e desligando por 1h durante o período de um dia completo para que o regime transitório também fosse analisado.



(a) Encapsulamento para o módulo *Arduino*, (b) Encapsulamento para o módulo *Raspberry Pi* e câmara.

Figura 4.5: Encapsulamentos impressos.

#### 4.5.1 Identificação do Sistema - *Matlab*

Para a identificação da planta, utilizou-se a *toolbox* de identificação de modelos dinâmicos *System Identification*. Utilizou-se 70% dos dados colhidos ao longo de um dia como treino e os 30% restante como validação. A configuração pode ser conferida na figura 4.6.

A ideia da ferramenta é fornecer curvas de treino e curvas para validação, selecionando-se os métodos desejados. Como resultado, tem-se uma função de transferência de saída que aproxima as curvas de entrada.

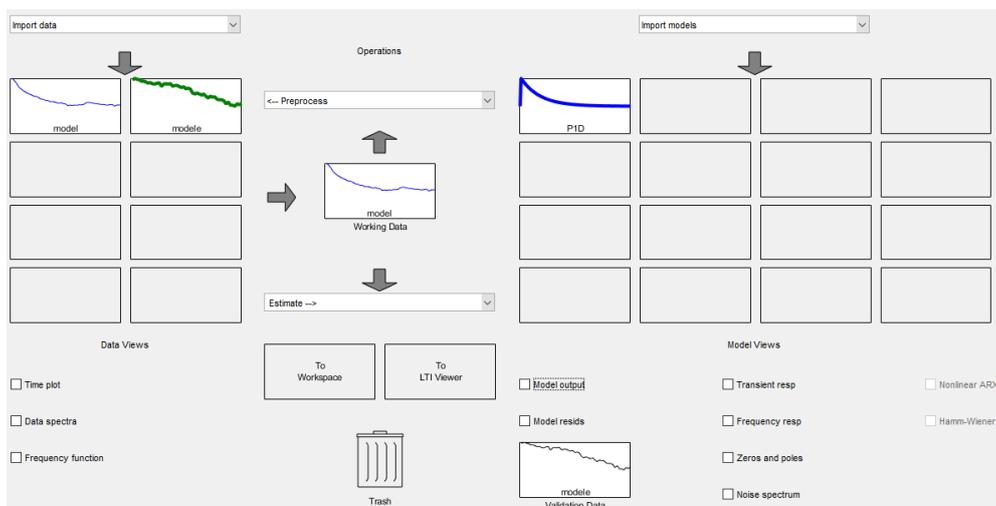


Figura 4.6: *Toolbox* de identificação do sistema (*Matlab*).

### 4.5.1.1 Simulação do Sistema em Malha Aberta

Para a simulação do sistema em malha aberta foi utilizado como base o modelo *simulink* desenvolvido por [14]. Esse modelo foi adaptado para incluir as perturbações causadas pelo número de pessoas. Além disso, os parâmetros foram modificados a fim de se adequar melhor a planta identificada. Vale ressaltar, que as modificações e testes foram feitos utilizando o *Matlab 2018a Update 2*.

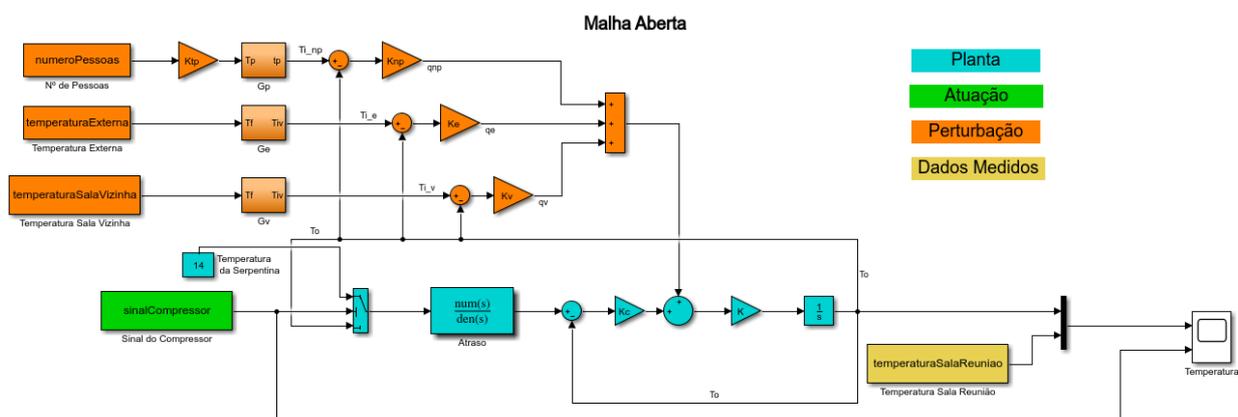


Figura 4.7: Modelo *simulink* para simulação do sistema em malha aberta.

A figura 4.7 mostra o modelo *simulink* utilizado para as simulações. Pode-se ver que o sinal do compressor determina se a temperatura de referência do ar-condicionado será a temperatura da serpentina ( $T = 14^{\circ}\text{C}$ ) ou se será a temperatura atual da sala ( $T = T_0$ ). Ou seja, o sinal do compressor define se o ar-condicionado irá ligar para resfriar a sala ou se irá desligar e a tendência será aquecer a sala.

Em azul, tem-se a planta do ar-condicionado. Ele é modelado como um atraso seguido do sistema de primeira ordem. Essa configuração é utilizada para se trabalhar com entradas e saídas de temperatura e no meio, com a informação de fluxo de calor, necessária para que a influência das perturbações pudessem ser analisadas.

Em laranja, por sua vez, pode-se ver as perturbações consideradas no sistema térmico. Elas são as influências das pessoas, da temperatura externa e da temperatura da sala vizinha. Existe um atraso associado a cada uma delas. Além disso, vale lembrar que o fluxo de calor é proporcional a diferença entre a temperatura do outro ambiente ou da pessoa e a temperatura atual da sala.

Por fim, em amarelo estão os dados reais medidos para comparação.

## 4.6 Abordagens de Controle

Nesta trabalho foram utilizadas três diferentes abordagens de controle. Foram projetado os clássicos controladores liga-desliga e PI para utilizar como base de comparação. Além destes, foram projetados um controlador antecipativo que deve diminuir ou eliminar a influência da perturbação

gerada pela presença de pessoas.

Vale ressaltar que utilizou-se para o PWM, um período de 2 minutos, com ciclos de trabalhos mínimos de 40 s.

#### 4.6.1 Controlador Liga-Desliga

O algoritmo desenvolvido para implementar o controlador liga-desliga é extremamente simples. Basta comparar a temperatura medida com a temperatura máxima e mínima pré-definidas. O controlador segue a regra:

- ligar se  $y > y_{max}$ ;
- desligar se  $y < y_{min}$ ;
- manter o estado anterior caso  $y_{min} > y > y_{max}$ .

Vale ressaltar ainda que realizaram-se testes com faixas grandes de valores máximos e mínimos, entretanto, isso fazia com que a resposta do sistema não ficasse adequada, uma vez que este possuía grandes variações em torno da referência. Inicialmente, utilizou-se uma faixa de  $\pm 0.5^{\circ}\text{C}$  para os testes de 24h, todavia, observou-se que os demais controladores operavam em variações próximas a  $\pm 0.3^{\circ}\text{C}$ , portanto, utilizou-se essa faixa para obter uma comparação justa nos testes finais de 8h.

##### 4.6.1.1 Simulação do Controlador Liga-Desliga

A figura 4.8 mostra a implementação do controlador liga-desliga no *template* para simulação.

Em verde, tem-se o controlador. Pode-se ver que a temperatura de referência é configurada como  $T_{ref} = 23^{\circ}\text{C}$ . Calcula-se o erro entre a temperatura desejada e a saída  $e$ , dependendo do valor do erro, o controlador é ou não acionado.

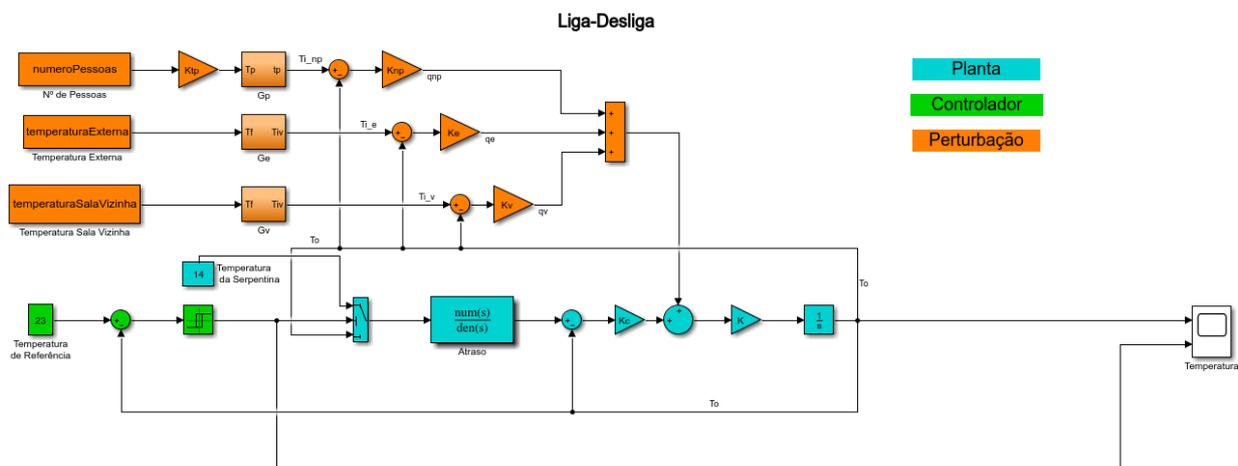


Figura 4.8: Modelo *simulink* para simulação do sistema com o controlador liga-desliga.

A implementação deste controlador se deu com base em um intervalo diferencial de  $\pm 0,3^{\circ}\text{C}$ .

## 4.6.2 Controlador PI

Para a implementação do controlador PI, utilizou-se a ferramenta *sisotool* do *Matlab*. Nela, realizou-se a sintonia do controlador por meio do método da Resposta Robusta no Tempo. Idealmente, esse método fornece um controlador com velocidade máxima na resposta e com regime transiente sem oscilações, todavia, esta abordagem se torna ineficiente em sistemas reais, graças às não-linearidades do sistema e limitações físicas dos atuadores. Então, decidiu-se por adotar os requisitos de modo a garantir uma resposta mais lenta e robusta, ou seja, com um regime transiente mais suave, fazendo com que o atuador seja menos solicitado, preservando sua vida útil; já o tempo não foi um fator de grande preocupação, pois a dinâmica lenta do sistema dá margem para um controle com uma resposta não tão rápida. As configurações para a especificação podem ser vistas na figura 4.9.

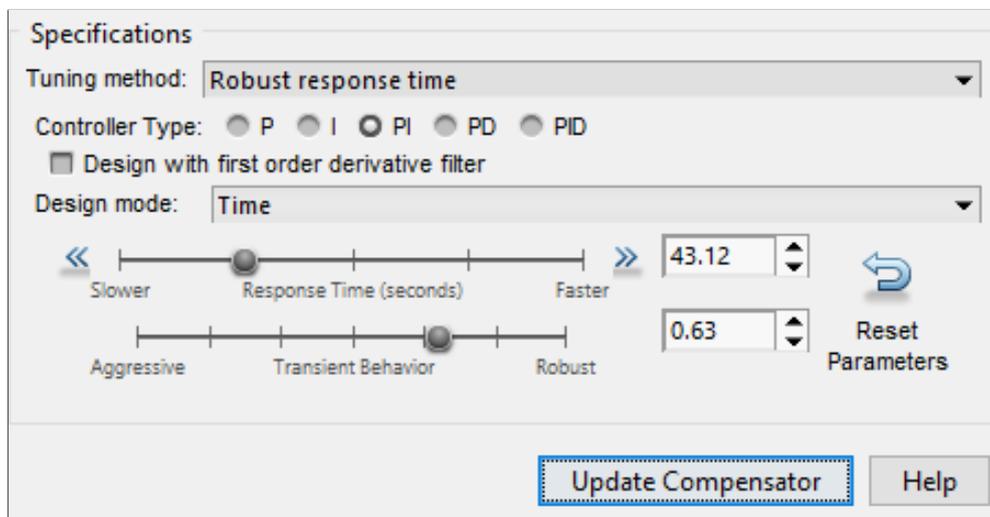


Figura 4.9: Especificação para a sintonia do controlador PI realizada no *Matlab*.

Posteriormente, o controlador projetado foi implementado em *Python* de modo a ser executado no *Raspberry Pi* e o sinal de controle calculado é enviado ao *Arduino* para acionamento do relé. Utilizou-se a biblioteca *Simple-PID* documentada em [42], na qual, escolheram-se os ganhos proporcionais e integrais e o valor da referência e retornava-se o sinal de controle a ser enviado para a planta.

### 4.6.2.1 Simulação do Controlador PI

O modelo *simulink* utilizado para simular o comportamento do sistema com o controlador PI pode ser visto na figura 4.10.

Primeiramente, calcula-se o erro entre a temperatura de saída e a referência. A partir desse erro, tem-se o sinal de controle gerado pelo PI. Devido a algumas limitações do sistema é necessário um bloco de saturação e um bloco de ganhos para converter o sinal de controle para um sinal que

possa ser usado como entrada do bloco *PWM*. O sinal de saída do *PWM* é o sinal enviado ao compressor.

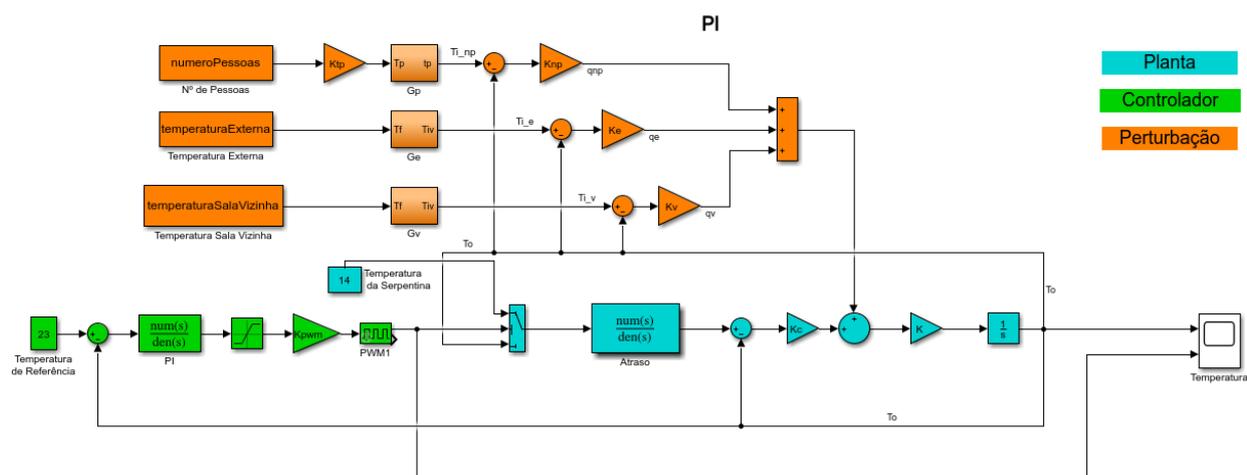


Figura 4.10: Modelo *simulink* para simulação do sistema com o controlador PI.

#### 4.6.2.2 Controle PI com Correção do Ponto de Operação para Perturbações de Temperatura

Visando tratar o problema gerado pela perturbação da temperatura quando estas se encontravam altas, resultando em um deslocamento da referência, implementou-se um controlador PI que atua normalmente dentro de uma faixa de temperatura entre 22,7°C e 23,3°C, caso a temperatura saia desta faixa para baixo, desliga-se o compressor até que a temperatura retorne ao ponto de operação, já se a temperatura sai desta faixa para cima, satura o compressor até que o ponto de operação seja novamente atingido, então, passa-se a utilizar-se o PI normalmente. Este novo método tem por finalidade diminuir o erro da resposta, entretanto, possivelmente, resultará em um maior gasto energético.

#### 4.6.3 Controlador Antecipativo

Já para o controle antecipativo, utilizaram-se primeiramente as informações acerca do número de pessoas, para isso, utilizou-se a câmera instalada no laboratório aliada aos algoritmos de visão computacional, abordados na seção 2.3.3.

Com a informação do número de pessoas em mãos, realizaram-se os cálculos expressos na seção 2.2.1, para estimar a influência das pessoas na carga térmica. Do valor encontrado, subtraía-se do sinal de controle de modo a cancelar a perturbação antes que esta afetasse o funcionamento do sistema. Para o controle em malha fechada, utilizou-se o controlador PI.

Uma vez que o sistema estava instalado, implementou-se uma funcionalidade de apenas ligar o ar-condicionado quando pessoas eram detectadas na sala de reuniões, ou seja, para situações em que a sala estivesse vazia, o sistema desligava.

### 4.6.3.1 Simulação do Controlador Antecipativo

Na figura 4.11, vê-se o modelo *simulink* utilizado para a simulação do controlador antecipativo. Pode-se ver que mantém a estrutura do controlador PI. Além disso, a partir da informação do número de pessoas, pode-se estimar a influência térmica. Basta multiplicar o número de pessoas pela carga térmica calculada por pessoa. Esse sinal irá diminuir ou eliminar o efeito perturbação causada pela presença das pessoas.

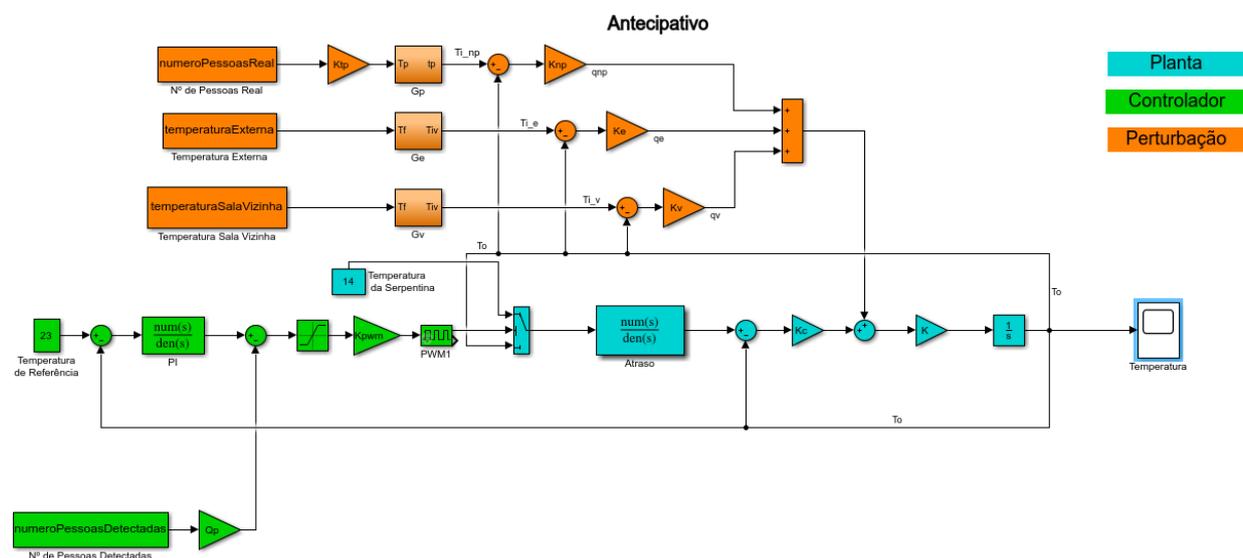


Figura 4.11: Modelo *simulink* para simulação do sistema com o controlador antecipativo.

## 4.7 Ensaios no Sistema Real

Foi necessário definir um conjunto de ensaios a serem realizados e seus requisitos, a fim de tornar a comparação mais justa e completa. Os ensaios foram:

- Ensaios de 1 dia: ensaios realizados durante 24 horas repetindo ciclos de 3 horas ligado e 1 hora desligado.
  - Liga-desliga;
  - PI com pouca influência do número de pessoas;
  - PI com muita influência do número de pessoas e
  - Antecipativo.
- Ensaios de 8 horas: ensaios visando simular o comportamento do ar-condicionado no dia-a-dia normal da sala de reunião do LARA. O ar-condicionado deve ficar ligado das 9h até as 17h.
  - Liga-desliga;

- PI;
- Antecipativo com mudança de referência quando não há pessoas na sala e
- Antecipativo sem mudança de referência quando não há pessoas na sala.

Para a realização de testes no sistema real, definiu-se alguns pontos a serem observados durante o experimento e outros a serem anotados para fins de comparação posterior. A cada ensaio, anotava-se a potência atual do medidor de energia  $e$ , ao final, a potência final registrada. A diferença entre as duas potências fornecia o gasto energético que o teste em questão demandou.

Outras análises a serem feitas eram a verificação da temperatura externa, temperatura da sala vizinha e o número de pessoas, para isso, desenvolveu-se uma métrica para estimar um valor baseado nessas variáveis de modo a tornar possível uma análise mais justa entre testes em diferentes dias. Isto ocorre pois a temperatura externa não é uma variável controlável e a temperatura da sala vizinha é apenas parcialmente controlada, já que o ar-condicionado acaba podendo ser utilizado por outros usuários do LARA. Logo, temperaturas mais frias contribuem para que o sistema chegue mais facilmente à referência, ou ao menos influenciam pouco; enquanto dias mais quentes têm como consequência uma maior exigência do sistema de refrigeração.

A métrica foi construída utilizando-se como base as integrais das temperaturas externa e da sala vizinha no dia e da integral do número de pessoas no dia. Foi criado o chamado Índice de Balanço Térmico (IBT), que correlaciona as variáveis de perturbação com base nas integrais e nos ganhos de influência estimados em simulação. O IBT é dado pela equação (4.1), mostrada a seguir:

$$IBT = \frac{K_e \int_{t_0}^t (T_e - T_{ref})dt + K_v \int_{t_0}^t (T_v - T_{ref})dt + K_{np} \int_{t_0}^t N_p (T_p - T_{ref})dt}{t - t_0}, \quad (4.1)$$

onde  $T_e$  é a temperatura externa,  $T_v$ , a temperatura da sala vizinha,  $T_p$ , a temperatura média de uma pessoa e  $N_p$ , o número de pessoas. Já  $K_e$ ,  $K_v$  e  $K_{np}$  são os ganhos adimensionais que representam a influência da perturbação na dinâmica da sala, ou seja, quanto maior esses ganhos, maior será o peso da determinada perturbação na temperatura da sala.

Também foi analisado o erro do sistema utilizando a fórmula da raiz do erro médio quadrático (RMSE), mostrada na equação (4.2):

$$RMSE = \sqrt{\frac{\sum_{i=1}^N (T_{ref} - T_{medida})^2}{N}}, \quad (4.2)$$

onde  $T_{ref}$  é a temperatura de referência para o sistema de controle, no trabalho em questão, esta é de 23°C; já a  $T_{medida}$  é a temperatura medida na sala de reunião durante o experimento.

Finalmente, realizou-se um comparativo de melhoria do controlador proposto. Os cálculos das melhorias no erro e energética em termos percentuais foram realizados com base, respectivamente, na equação (4.3):

$$M_{RMSE}\% = \frac{RMSE_{tradicional} - RMSE_{antecipativo}}{RMSE_{tradicional}} \times 100, \quad (4.3)$$

e na equação (4.4):

$$M_{energetica}\% = \frac{E_{tradicional} - E_{antecipativo}}{E_{tradicional}} \times 100, \quad (4.4)$$

onde  $M_{RMSE}\%$  é a melhoria percentual no erro quadrático médio do sistema,  $RMSE_{ref}$  é o RMSE do controlador com o qual se deseja comparar, enquanto  $RMSE_{Antecipativo}$  é o RMSE do controlador proposto. Já a  $M_{Energetica}\%$  é a melhoria percentual na economia energética,  $E_{ref}$  é a potência gasta do controlador com o qual se deseja comparar, enquanto  $E_{Antecipativo}$  é a potência do controlador Antecipativo.

# Capítulo 5

## Resultados

### 5.1 Identificação de Pessoas

Como não foi possível, em um primeiro momento, fazer a instalação do *Raspberry Pi* e da câmera na sala de reunião do LARA, utilizou-se dois vídeos distintos para fazer os testes da identificação e contagem do número de pessoas.

O primeiro vídeo, retirado de [43], corresponde a um ambiente aberto com alto fluxo de pessoas caminhando em diversas direções. O vídeo possui uma resolução de 640x480 *pixels* e uma taxa de *frames* igual à 24 fps. Para fins de análise, este vídeo foi denominado BD1.

Já o segundo vídeo, retirado de [44], é mais compatível com a nossa aplicação visto que é posicionada no corredor de um ambiente fechado. Sua resolução é de 160x128 *pixels* e sua taxa de *frames* é 25 fps. Este vídeo foi denominado BD2.

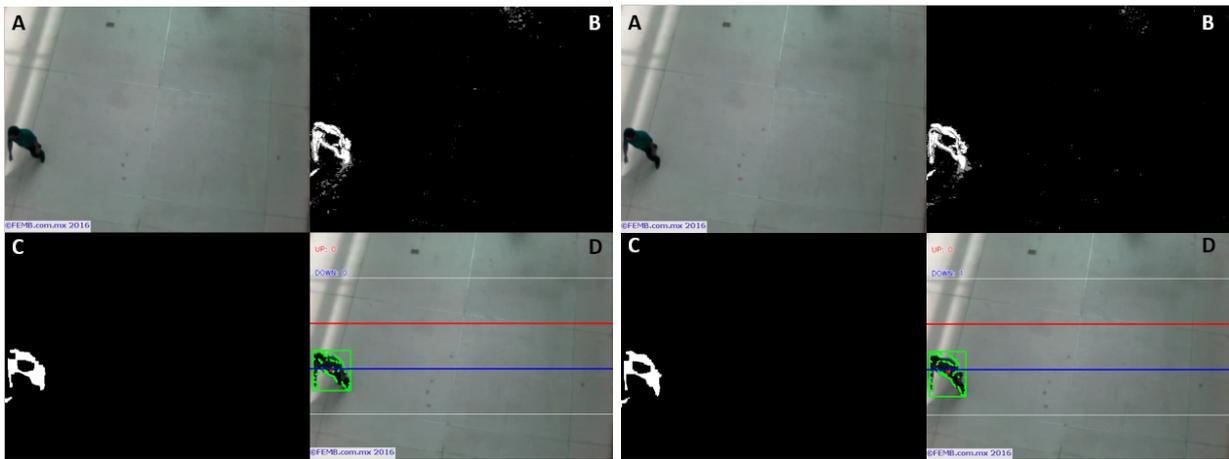
O BD2 conta com imagens com uma resolução menor do que o BD1, porém apresenta uma maior taxa de *frames* por segundo.

As figuras 5.1(a) até 5.2(b) representam alguns exemplos de resultados obtidos para a primeira base de dados. Já as figuras 5.3(a) até 5.4(c) representam alguns exemplos obtidos para a segunda base.

Os exemplos selecionados envolvem casos de acerto, assim como casos de erro. Além disso, cada figura apresenta quatro quadrantes. O quadrante A é a imagem original antes do processamento. Já o quadrante B representa a imagem após a subtração de fundo. Vale lembrar que a parte estática da imagem é representada por preto, a parte alterada é representada por branco e, em cinza, tem-se a sombra produzida pelo movimento. Ao aplicar os filtros e as transformações, elimina-se a parte cinza e o resultado é o quadrante C.

Por fim, no quadrante D, tem-se a imagem com uma análise mais completa. Para a contagem são traçadas quatro linhas:

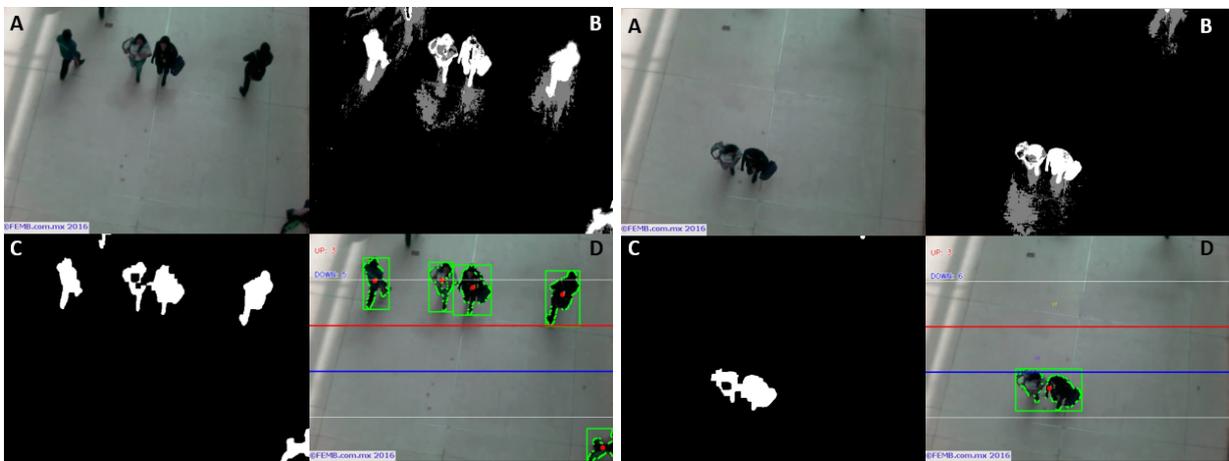
- linha vermelha: quando uma pessoa ultrapassa a linha vermelha na direção certa, ou seja, indo para cima, incrementa-se o contador *UP*;



(a) Quadro anterior a contagem.

(b) Quadro posterior a contagem.

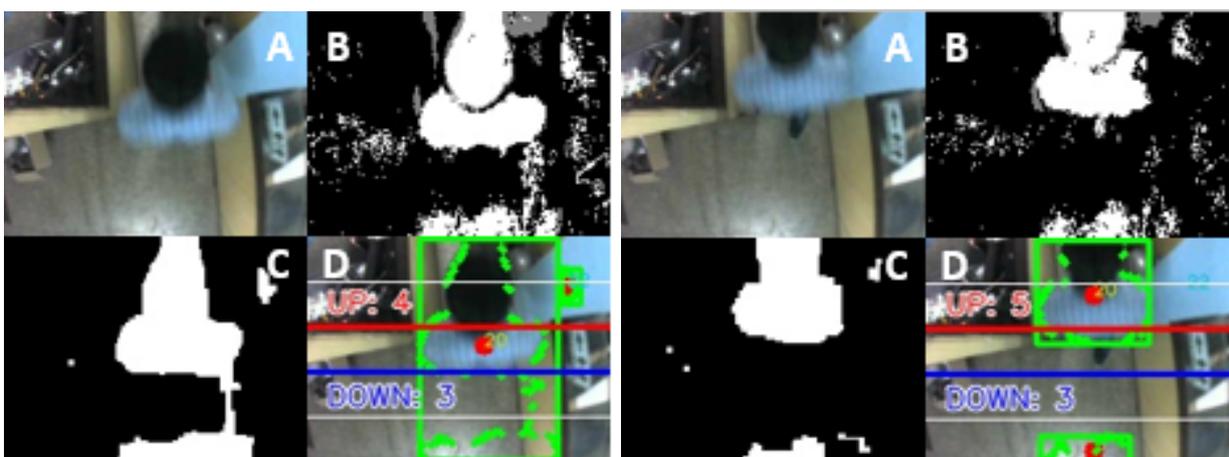
Figura 5.1: Contagem correta no vídeo BD1 (ambiente aberto).



(a) Quadro anterior a contagem.

(b) Quadro posterior a contagem.

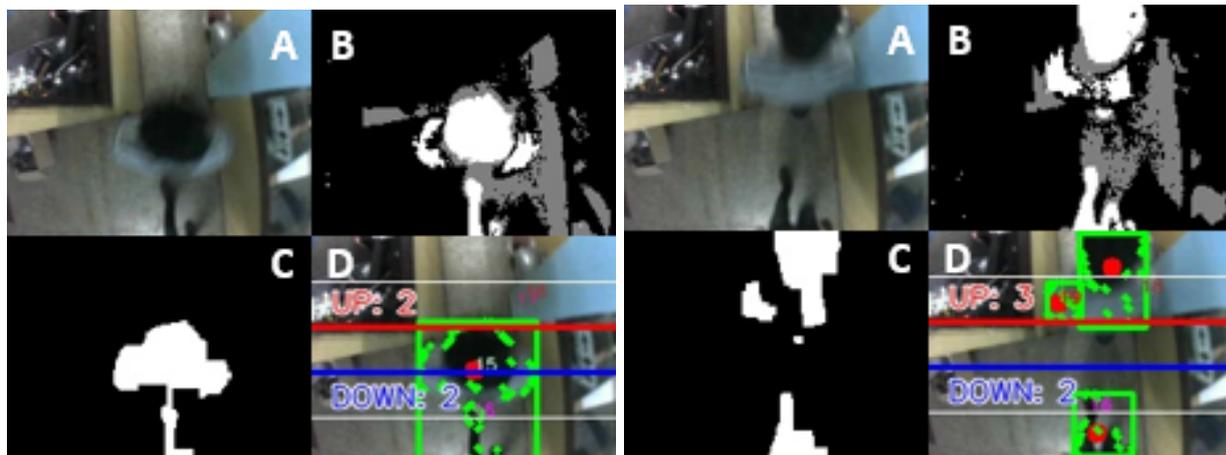
Figura 5.2: Contagem incorreta no vídeo BD1 (ambiente aberto).



(a) Quadro anterior a contagem.

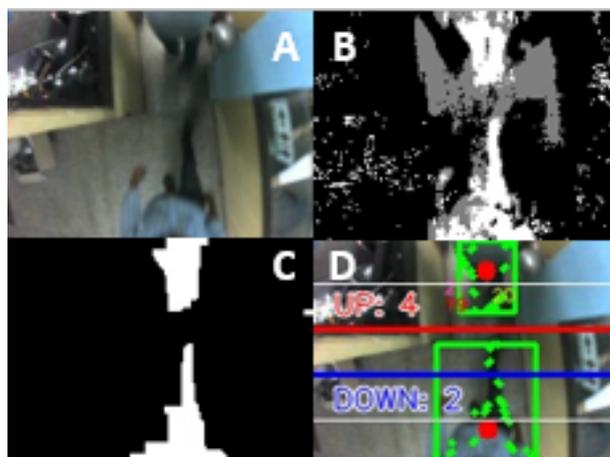
(b) Quadro posterior a contagem.

Figura 5.3: Contagem correta no vídeo BD2 (ambiente fechado).



(a) Quadro anterior a contagem.

(b) Quadro posterior a contagem.



(c) Quadro com contagem incorreta.

Figura 5.4: Contagem incorreta no vídeo BD2 (ambiente fechado).

- linha azul: quando uma pessoa ultrapassa a linha azul na direção certa, ou seja, indo para baixo, incrementa-se o contador *DOWN* e
- linhas brancas: quando uma pessoa sai do interior das linhas brancas, ela pode deixar de ser rastreada e o espaço de memória deve ser liberado.

Além disso, o contorno da pessoa é desenhado em verde. Assim, como o retângulo para representá-la. O ponto vermelho no interior do quadrado é o ponto utilizado para definir a localização e seguir a trajetória.

As figuras 5.1(a) e 5.1(b) mostram um caso de contagem correta do número de pessoas. As figuras, representam *frames* distintos em um ambiente aberto. Note que com a passagem da pessoa indo da parte superior da imagem para a parte inferior, há o incremento do contador *DOWN*.

Ou seja, a figura 5.1(a) representa um quadro anterior a passagem da pessoa pela linha azul. Pode-se ver que os valores dos contadores são  $DOWN = 0$  e  $UP = 0$ . Já a figura 5.1(b) corresponde a um quadro posterior a passagem, logo os valores dos contadores são  $DOWN = 1$  e  $UP = 0$ .

Ainda utilizando o BD1, percebe-se que apesar de ser acurado, o código ainda tinha casos de erro, como observado nas figuras 5.2(a) e 5.2(b). Observe que nestas imagens, duas pessoas lado a lado saem da parte superior e vão para a parte inferior, no entanto, instantes antes de passarem pela faixa de detecção, o algoritmo considera que as duas pessoas são uma única, incrementando unitariamente o contador *DOWN*.

O segundo conjunto de resultados diz respeito ao BD2, representando um ambiente fechado. As figuras 5.3(a) e 5.3(b) ilustram um caso de contagem correta. Perceba que uma pessoa é identificada se movendo da parte inferior da imagem para a parte superior, resultando em um incremento do contador *UP*.

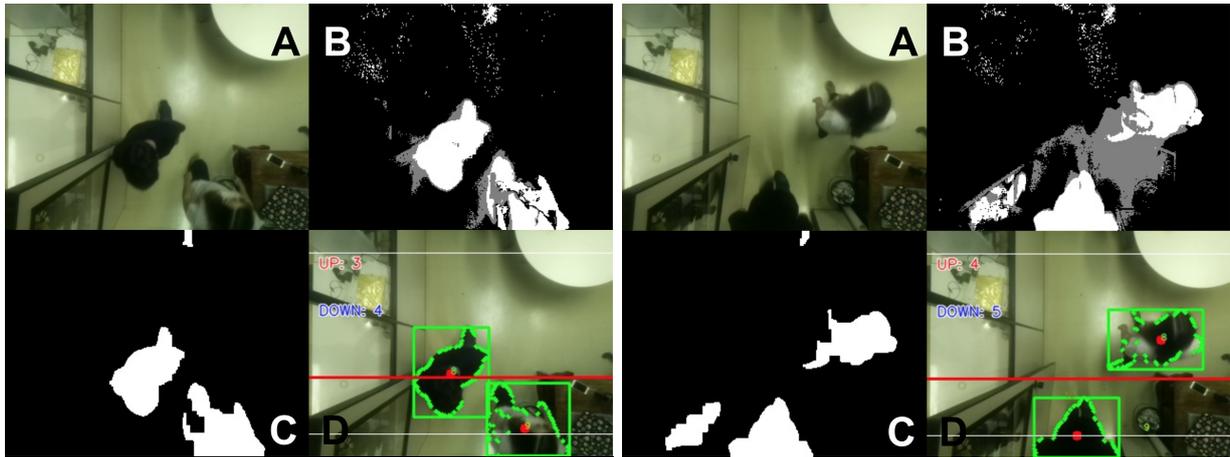
Contudo, para esse tipo de caso, erros pontuais também ocorrem. Nas figuras 5.4(a), 5.4(b) e 5.4(c) uma pessoa se desloca da parte inferior para a parte superior, no entanto, o algoritmo identifica outros objetos que não são pessoas gerando uma contagem incorreta do contador *DOWN*.

A taxa de acerto no BD1 foi maior. Isso ocorreu devido à baixa resolução dos quadros de BD2.

Depois que o *Raspberry Pi* e a câmera foram devidamente instalados, pode-se realizar os testes com o *stream* de vídeo da sala de reunião em tempo-real. Alguns resultados podem ser vistos na figura 5.5(a) e 5.5(b). Nesse exemplo, tem-se um pessoa entrando na sala e outra pessoa saindo da sala ao mesmo momento. A figura 5.5(a) é anterior à contagem das pessoas, enquanto que a figura 5.5(b) é posterior. Ao compará-las, vê-se que os contadores *UP* e *DOWN* foram ambos incrementados conforme o esperado.

Os resultados obtidos no ambiente da sala de reunião também não foram completamente robustos e resilientes. O *software* apresentou alguns problemas devido a presença de sombras que às vezes eram reconhecidas erroneamente como pessoas.

Além disso, algumas situações específicas já mencionas continuaram a gerar falhas de contagem. Como, por exemplo, duas pessoas muito próximas serem reconhecidas como uma única pessoa.



(a) Quadro anterior a contagem.

(b) Quadro posterior a contagem.

Figura 5.5: Contagem em tempo-real na sala de reuniões do LARA.

## 5.2 Identificação da Planta

A fim de se fazer a identificação da planta, aplicou-se uma sequência de degraus no compressor para ver o comportamento da planta no regime transitório e permanente. Os dados de identificação podem ser vistos nas figuras 5.6(a), 5.6(b) e 5.6(c).

A figura 5.6(a) mostra a temperatura medida na sala de reunião, assim como a sua média móvel para facilitar a visualização. Já na figura 5.6(b) é possível ver a temperatura da sala vizinha e a temperatura externa. Na figura 5.6(c), vê-se o acionamento do compressor, ou seja, os momentos em que o ar-condicionado está ligado ( $signal = 1$ ) ou desligado ( $signal = 0$ ).

Com os dados em mãos, foi possível notar que o sistema possuía uma resposta de primeira ordem, conforme o esperado e que o sistema tendia resfriar até uma temperatura aproximada de  $14^{\circ}\text{C}$ . Esse valor coincide com a temperatura da serpentina do aparelho de ar-condicionado.

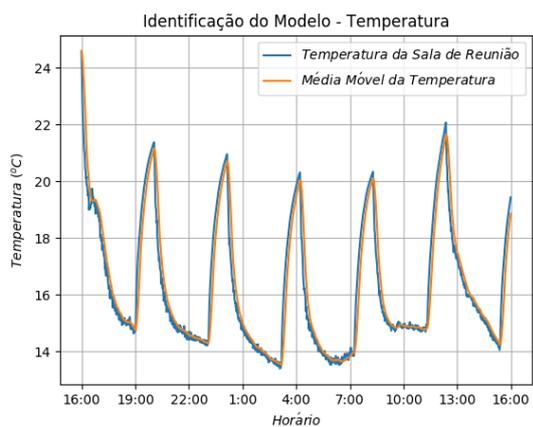
Também foi possível notar uma relação entre a temperatura mínima atingida a cada acionamento e as temperaturas dos outros ambientes.

### 5.2.1 Identificação do Sistema - *Matlab*

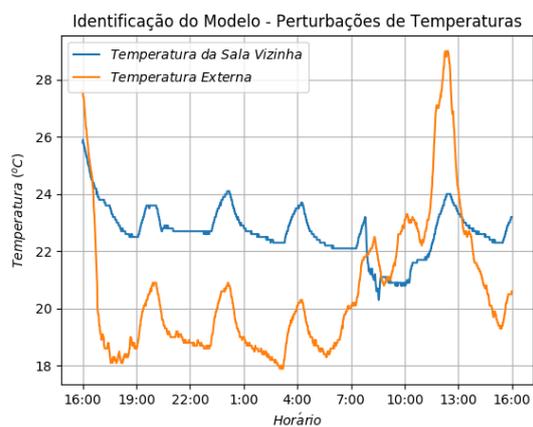
A função de transferência identificada utilizando a *toolbox* do *Matlab* é uma função de transferência de primeira ordem com atraso:

$$G(s) = \frac{K}{\tau s + 1} e^{-\theta s}, \quad (5.1)$$

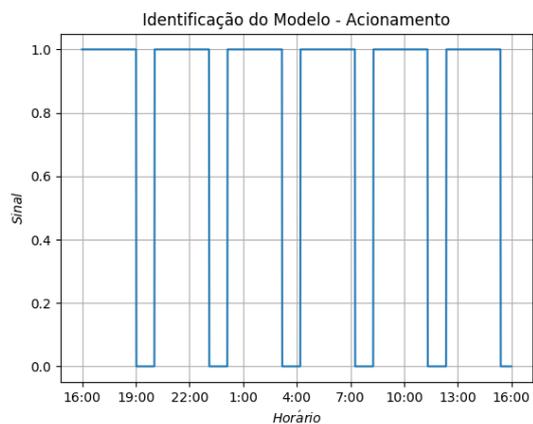
onde  $K = 0,36783$ ,  $\tau = 798.84 \text{ s}$  ( $13,314 \text{ min}$ ) e  $\theta = 3.9 \text{ s}$  ( $0,065 \text{ min}$ ). Substituindo os valores encontrados tem-se a equação (5.2):



(a) Temperatura da Sala de Reuniões.



(b) Temperaturas Externas e da Sala Vizinha.



(c) Acionamento do Compressor.

Figura 5.6: Dados coletados para identificação da planta.

$$G(s) = \frac{0,36783}{798,84s + 1} e^{-3,9s}. \quad (5.2)$$

Com isso, vê-se que a contante de tempo do sistema é, aproximadamente, 13 *min*. Além disso, existe um atraso próximo a 4 *s*, entre o envio do sinal ao compressor e o início do resfriamento do sistema. O modelo de primeira ordem identificado apresentou um *fit* de 95,38%.

O modelo do sistema adotado neste projeto é de uma entrada e uma saída (SISO), mas vale lembrar que o comportamento do sistema varia conforme as mudanças na temperatura externa, na temperatura da sala vizinha e de outras perturbações. Um dia com as temperaturas dos outros ambientes mais baixas são identificados com uma contante de tempo mais rápida, por exemplo. O ideal, então, seria fazer uma nova identificação a cada utilização do sistema.

### 5.2.2 Simulação do Sistema em Malha Aberta

O resultado da simulação do modelo mostrado na figura 4.7 pode ser visto na figura 5.7, onde a curva azul mostra o valor de temperatura da sala de reunião simulado, a curva vermelha mostra a temperatura real da sala de reunião e nas curvas roxa e amarela pode-se ver, respectivamente, as temperaturas externa e da sala vizinha.

Pode-se ver a semelhança entre a curva simulada e a curva real da temperatura da sala de reunião.

A partir dessa identificação, tem-se os ganhos do fluxo de calor das pessoas ( $K_{np}$ ), do ambiente externo ( $K_e$ ) e da sala vizinha ( $K_v$ ) para a sala de reunião, com base na equação (5.3):

$$\begin{cases} K_e = 1,4, \\ K_v = 3,3, \\ K_{np} = 0,1. \end{cases} \quad (5.3)$$

Com isso, pode-se concluir que a maior influência é dada pela diferença de temperatura entre a sala de reunião e as salas vizinhas. Em seguida, é pelo ambiente externo. Por último, tem-se a diferença causada pelo fluxo de calor entre uma pessoa e o ambiente sala de reunião. Mas vale lembrar que esse é o ganho devido a uma única pessoa, conforme mais pessoas entram na sala essa perturbação térmica aumenta e passa a se tornar mais significativa.

Além dos ganhos, conseguiu-se estimar os atrasos das respectivas perturbações, podendo ser vistos na equação (5.4):

$$\begin{cases} a_e = 328,80 \text{ s}, \\ a_v = 414,00 \text{ s}, \\ a_{np} = 720 \text{ s}. \end{cases} \quad (5.4)$$

Onde  $a_e$ ,  $a_v$  e  $a_{np}$  representam os atrasos associados as perturbações do ambiente externo, da sala vizinha e devido as pessoas, respectivamente.

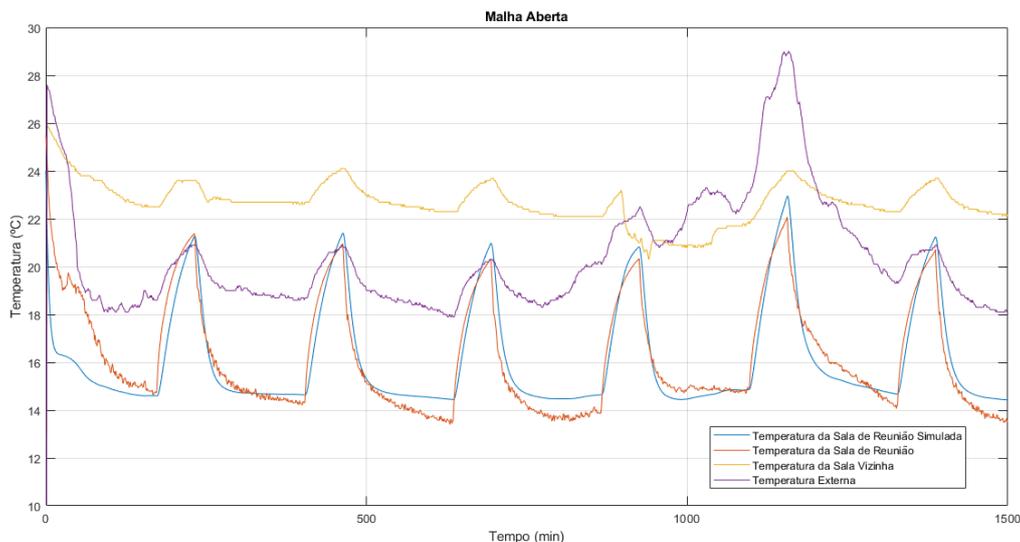


Figura 5.7: Resultado da simulação do sistema em malha aberta e temperaturas reais medidas.

## 5.3 Controlador Liga-Desliga

O controlador liga-desliga foi implementado em simulação utilizando o *Matlab* e o *Simulink*. Posteriormente, foi implementado no sistema real por um código em *Python* executado no *Raspberry Pi*.

### 5.3.1 Simulação do Controlador Liga-Desliga

O resultado da simulação do modelo com o controlador liga-desliga pode ser visto na figura 5.8. No gráfico, pode-se ver tanto as temperaturas como o número de pessoas.

A temperatura da sala de reunião se comporta como o esperado, considerando o intervalo diferencial de  $\pm 0,3$  °C. Quando a temperatura é elevada para 23,3 °C, o ar-condicionado liga até que atinja 22,7 °C. Quando o limite inferior de 22,7 °C é alcançado, o compressor desliga.

É interessante notar que quando o número de pessoas é maior, a temperatura se comporta um pouco diferente. Quando o compressor está desligado, o aumento da temperatura é mais rápido e quando o compressor está ligado, a redução da temperatura é mais lenta. Isso ocorre, porque a perturbação térmica demanda mais do ar-condicionado.

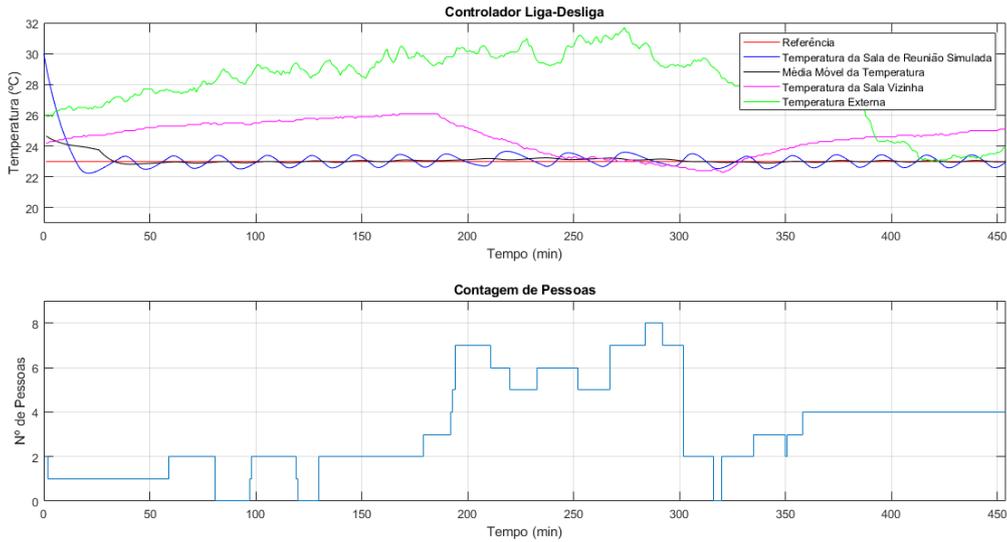


Figura 5.8: Resultado com o controlador liga-desliga.

### 5.3.2 Controlador Liga-Desliga no Sistema Real

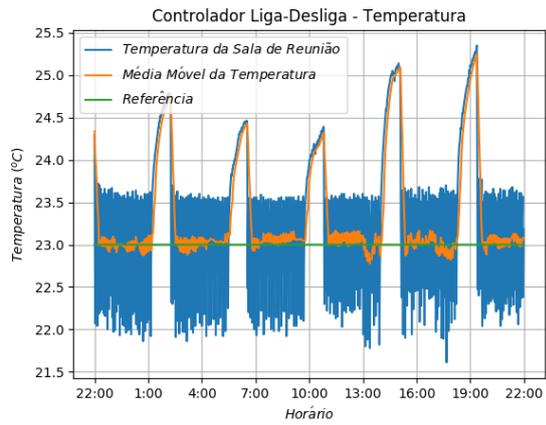
Para a implementação prática do controlador liga-desliga foram feitos testes de um dia (24 horas) e outros de 8 horas para analisar o seu efeito. Os testes de 24h serviam para analisar o comportamento em meio a diferentes situações de perturbação (temperaturas externas e ocupantes), analisando-se regimes transitórios, vez que o sistema permanecia ligado por 3h e desligado por 1h. Enquanto que o teste de 8h, tinha como propósito, avaliar o desempenho do controlador numa situação típica de uma sala de reunião em horário comercial, permanecendo ligado durante todo o período.

#### 5.3.2.1 Testes de 24h

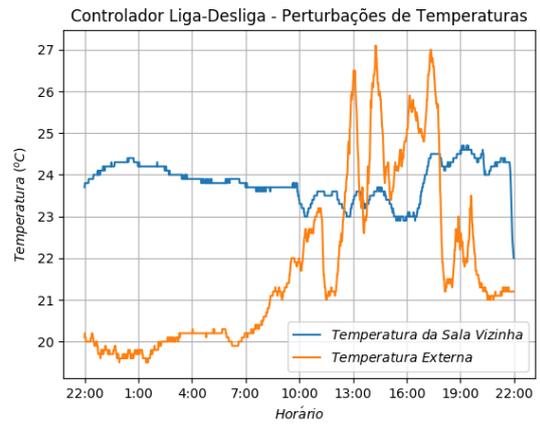
Na figura 5.9(a), pode-se ver o comportamento da temperatura da sala de reunião. É possível observar que a média móvel da temperatura está próxima à referência em todos os ciclos.

Percebe-se que temperatura variou pouco mais do que  $0,5\text{ }^{\circ}\text{C}$  acima e abaixo da referência. Isso ocorre porque no sistema real há uma limitação, não pode-se desligar e ligar o atuador muito rapidamente. Ou seja, tem-se que considerar um tempo mínimo obtido experimentalmente para que possa inverter o estado para não danificar o compressor a longo prazo. Esse tempo mínimo que o compressor deve permanecer entre um estado e o outro é de  $40\text{ s}$ . No algoritmo foi utilizado um período de  $27\text{ s}$ , porém devido a dinâmica lenta do sistema térmico, observou-se que o sinal para o atuador não mudava em apenas um período.

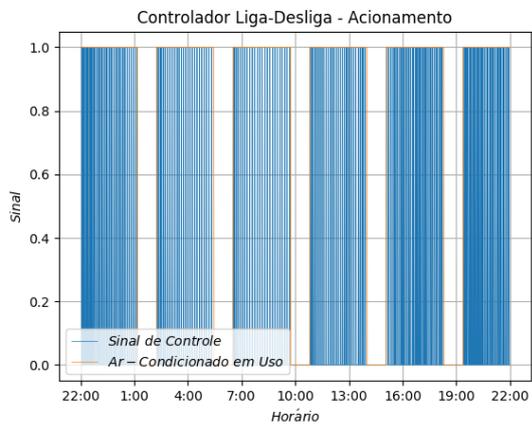
Já na figura 5.9(b), pode-se ver a temperatura externa e da sala vizinha durante o experimento. É possível observar que mesmo nos picos dessas temperaturas, o controlador liga-desliga ainda conseguiu se aproximar da referência.



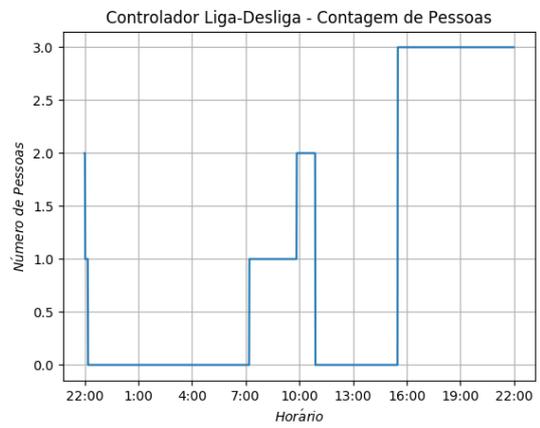
(a) Temperatura da Sala de Reuniões.



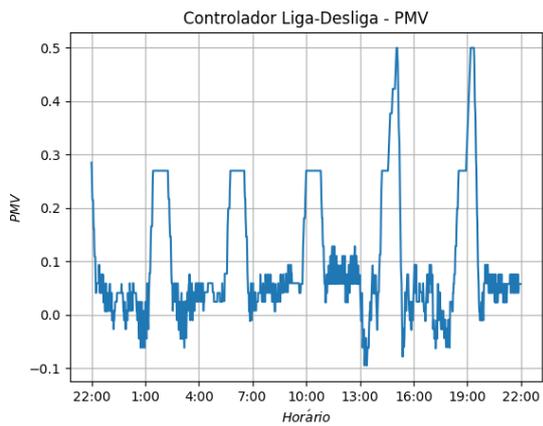
(b) Temperaturas Externas e da Sala Vizinha.



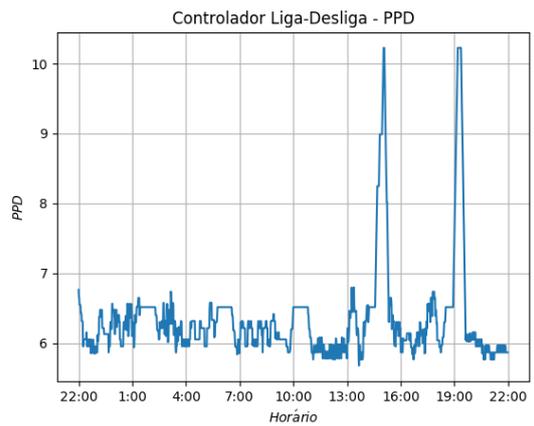
(c) Acionamento do Compressor.



(d) Número de Pessoas.



(e) Índice PMV



(f) Índice PPD

Figura 5.9: Controlador liga-desliga (24h).

A figura 5.9(c) mostra o sinal de acionamento do compressor assim como o estado do sistema, ou seja, mostra se o ar-condicionado está ligado ou desligado. O sistema é ligado por 3 horas e, em seguida, desligado por 1 hora para possibilitar a análise do regime permanente e transitório.

Em seguida, a figura 5.9(d), mostra a contagem de pessoas. Pode-se ver que a sala não foi muito utilizada durante esse experimento, atingindo um valor máximo de três pessoas durante o final da tarde.

Porém ao relacionar o número de pessoas com a temperatura da sala não fica muito claro o efeito dessa perturbação no sistema. Independente do número de pessoas presentes, a saída acompanhou a referência.

A figura 5.9(e) mostra variação do índice PMV ao longo do dia. Para definição dos valores de PMV utilizou-se o anexo E da norma ISO7730 [31]. Pode-se ver que enquanto o sistema estava ligado e em regime permanente, o valor mais distante do zero atingido foi próximo à 0,12.

O valor do índice PPD pode ser calculado a partir do PMV. O resultado está disponível na figura 5.9(f). Como explicado anteriormente, o PPD mostra a porcentagem prevista de pessoas insatisfeitas. Ao analisar o gráfico, pode-se concluir que, quando o sistema estava ligado e em regime permanente, o valor não ultrapassou a faixa de 7%.

Ao final do experimento, foi coletada a informação de gasto energético que foi 13,37 *kWh*. Também foi feito o cálculo do índice IBT pela equação (4.1) e estimou-se que a dificuldade do ensaio foi 3,95 °C. Além disso, o RMSE dado pela equação (4.2) é 0,51 °C.

### 5.3.3 Testes de 8h

Com o sistema sempre ligado durante um período de 8h, pôde-se verificar na figura 5.10(a) que o controlador seguiu bem a referência, mantendo um erro de aproximadamente  $\pm 0,5^\circ\text{C}$ . Pode-se notar ainda um desvio da referência por volta das 16h.

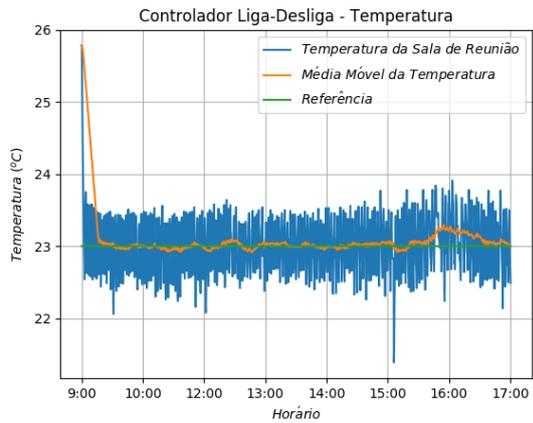
Já na figura 5.10(b), é possível verificar as perturbações de temperatura, em particular, próximo às 16h, percebe-se um pico da temperatura externa, chegando a temperaturas de  $29^\circ\text{C}$ . Esse pico explica a dificuldade encontrada pelo controlador por volta desse período.

Quando se verifica o acionamento do sistema na figura 5.10(c), pode-se perceber que o sinal de controle se manteve em 1 por muito tempo, indicando uma saturação do controlador.

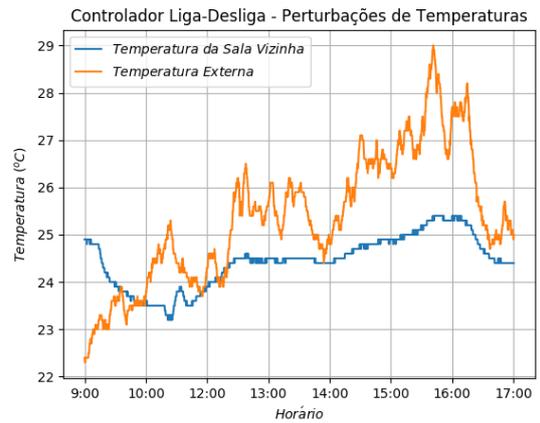
A contagem de pessoas está presente na figura 5.10(d), onde é possível verificar uma limitação de uso da sala por apenas 2 pessoas. Portanto, é possível concluir que essa não é uma perturbação grande o suficiente para efetivamente atrapalhar o sistema.

Nas figuras 5.10(e) e 5.10(f), pode-se ver que ambos os índices mantiveram-se baixos ao longo do ensaio, com exceção do período próximo às 16h.

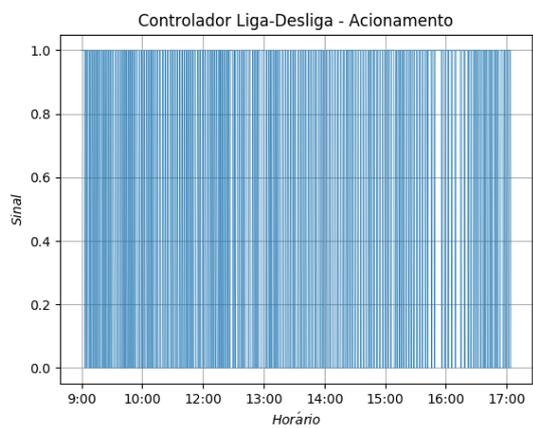
Então, analisaram-se os resultados finais do experimento. O gasto energético foi de 7,93 *kWh*. Calculou-se ainda o IBT com base na equação (4.1), com um valor de 8,63. Já o RMSE, calculado por (4.2), resultou em 0,38. Vale notar que a grande diferença do RMSE do teste de 24h para o



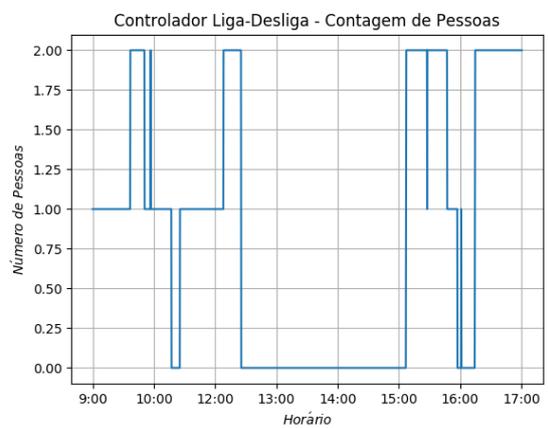
(a) Temperatura da Sala de Reuniões.



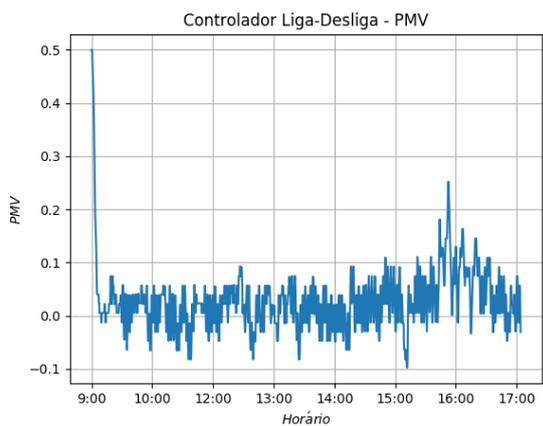
(b) Temperaturas Externas e da Sala Vizinha.



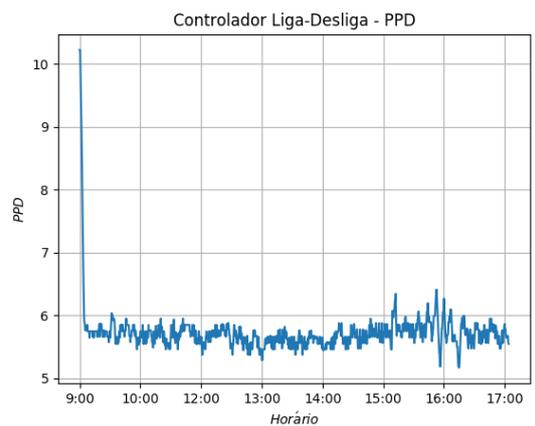
(c) Acionamento do Compressor.



(d) Número de Pessoas.



(e) Índice PMV



(f) Índice PPD

Figura 5.10: Controlador liga-desliga (8h).

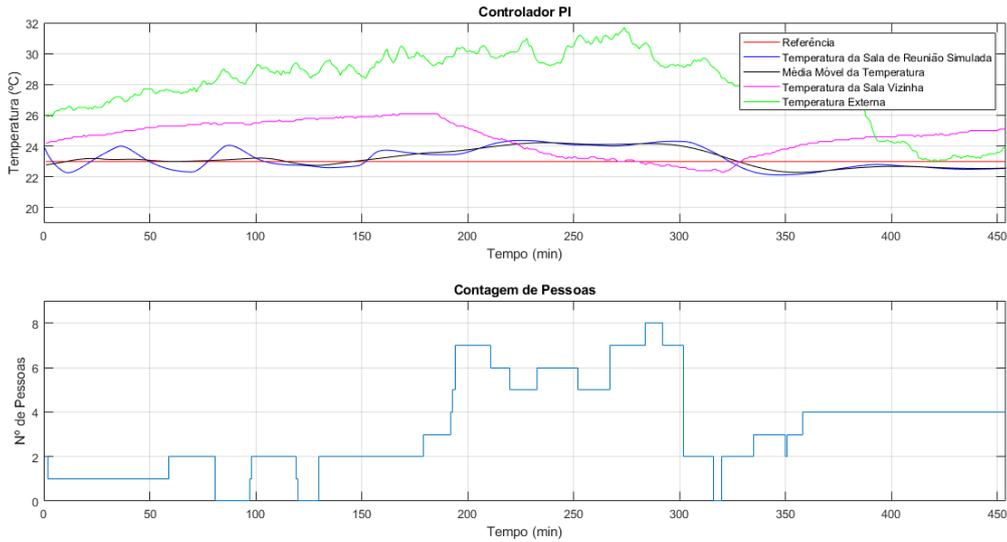


Figura 5.11: Resultado da simulação do sistema com o controlador PI.

teste de 8h é devido a mudança do intervalo diferencial.

## 5.4 Controlador PI

O controlador PI foi simulado utilizando o modelo mostrado na figura 4.10 e, em seguida, foi implementado no código em *Python* a fim de analisar o seu desempenho no sistema real.

O controlador PI projetado pode ser visto na equação (5.5).

$$G_c(s) = \frac{0,12s + 1,56}{s}. \quad (5.5)$$

### 5.4.1 Simulação do Controlador PI

O resultado do sistema com o controlador PI pode ser visto na figura 5.11. Pode-se ver que, até aproximadamente 170 minutos, a temperatura oscila em torno da referência. Neste período, nota-se também que o número de pessoas varia entre zero e duas, ou seja, a perturbação devido as pessoas é relativamente baixa.

Quando o intervalo de tempo está entre os 170 minutos e os 300 minutos, o número de pessoas é elevado e atinge um número máximo de oito pessoas na sala de reunião. Essa perturbação faz com que a temperatura se desloque e não consiga atingir a referência.

### 5.4.2 Controlador PI no Sistema Real

Para o teste no sistema real da sala de reunião do laboratório, realizou-se testes de 24h e 8h com uma alta e uma baixa perturbação do número de pessoas. A ideia aqui é que fosse possível

verificar a influência da perturbação devido ao número de pessoas, além de uma resposta adequada com um gasto energético reduzido.

#### 5.4.2.1 Testes de 24h

Inicialmente, realizaram-se os testes com uma perturbação do número de pessoas muito pequena. Ou seja, o número de pessoas na sala de reunião se mantinha sempre baixo de modo a evitar uma perturbação grande causada pela carga térmica das pessoas. O controle da temperatura da sala pode ser visto na figura 5.12(a).

Ao analisar o comportamento da curva, é possível verificar que a média móvel da temperatura não atingiu a referência nos horários de 18h às 21h e 7h às 18h. Como o controlador PI possui um canal integral que deveria zerar o erro, esse comportamento deve ser consequência de uma perturbação ou do tempo entre o cálculo do sinal de controle e a aplicação deste no compressor, devido ao *PWM*. Contudo, ao observar-se o erro, verifica um erro em geral próximo a  $0,3^{\circ}\text{C}$ , só ficando próximo a  $0,5^{\circ}\text{C}$  por volta das 12h.

Em seguida, pode-se observar as curvas relativas às temperaturas externa e da sala vizinha na figura 5.12(b).

Aqui pode-se verificar uma perturbação grande devido à temperatura externa por volta das 12h, atingindo picos de  $30^{\circ}\text{C}$ , o que pode explicar o deslocamento vertical da curva da temperatura da sala e aumento do erro.

Já a figura 5.12(c), mostra o acionamento do compressor, que se dá por meio do ciclo de trabalho do *PWM*, mais uma vez, verifica-se que o estado do sistema se mantém 3h ligado e 1h desligado.

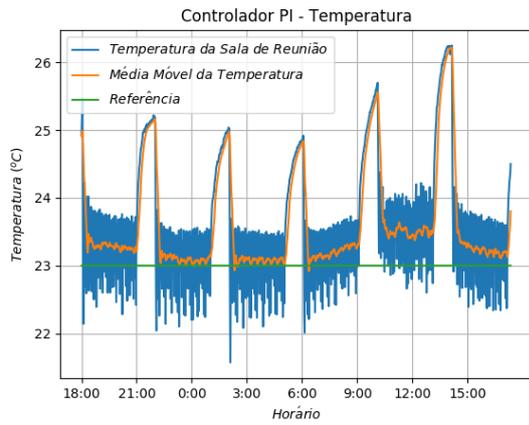
Para confirmar que a perturbação devido ao número de pessoas é baixa, podemos verificar na figura 5.12(d) que o número de pessoas durante todo o dia não passou de duas pessoas ao mesmo tempo na sala.

Para verificar o conforto térmico, utilizou-se os índices PMV e PPD. O índice PMV pode ser visto na figura 5.12(e), já o PPD, pode ser visto na figura 5.12(f).

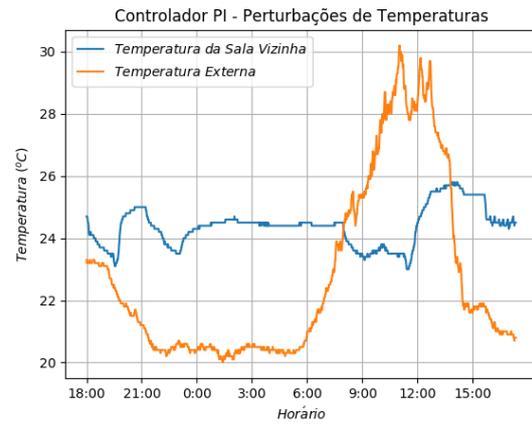
É possível verificar que o índice PMV atingiu um máximo 0,2 quando o ar-condicionado estava ligado e o sistema já estava em regime permanente. Já o PPD indica que a porcentagem de pessoas insatisfeitas foi aproximadamente 6%.

Ao fim do experimento, verificou-se o gasto energético e a dificuldade do dia para a realização do ensaio. O gasto energético foi de  $7,9\text{ kWh}$ , enquanto que o IBT foi calculado com base na equação (4.1), resultando em 5,47. O erro calculado foi de  $0,49^{\circ}\text{C}$  a partir da fórmula do RMSE mostrada na equação (4.2).

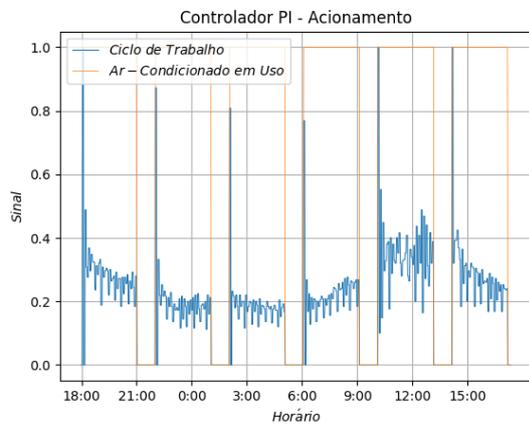
Em seguida, realizaram-se os ensaios com uma perturbação com alto número de pessoas para causar perturbação, ou seja, colocando cerca de seis pessoas na sala ao mesmo tempo, simulando um ambiente de reunião. Segundo as simulações, um número superior a quatro pessoas já se mostrava uma perturbação capaz de deslocar a curva de temperatura da sala de reunião para



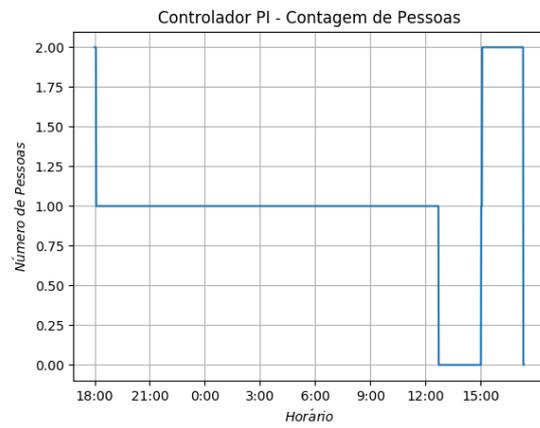
(a) Temperatura da Sala de Reuniões.



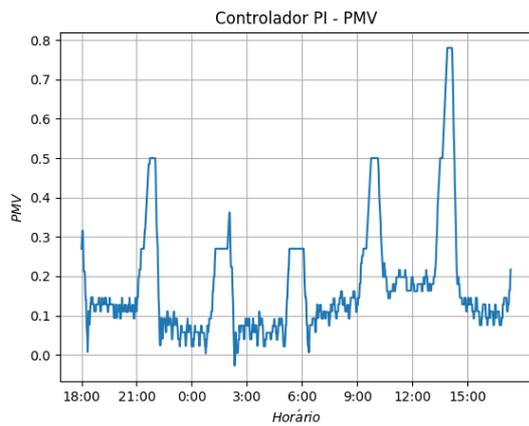
(b) Temperaturas Externas e da Sala Vizinha.



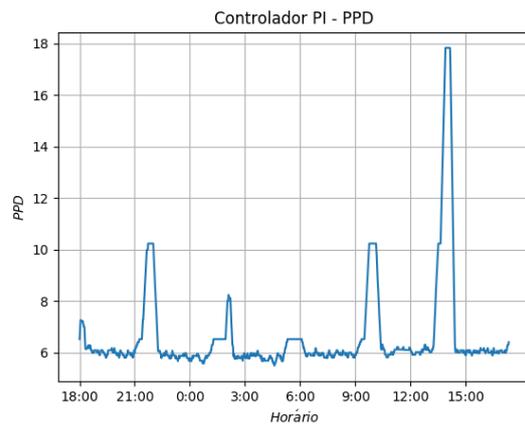
(c) Acionamento do Compressor.



(d) Número de Pessoas.



(e) Índice PMV



(f) Índice PPD

Figura 5.12: Controlador PI (24h) com baixo número de pessoas.

cima em um valor significativo.

O gráfico com a temperatura e a média móvel pode ser visto na figura 5.13(a). Percebe-se que a curva em geral, atingiu bem a referência, todavia, entre os horários de 9h e 15h, percebe-se um erro superior a  $0,5^{\circ}\text{C}$ , podendo ser considerado alto.

Para verificar o motivo que levou a tal erro, pode-se verificar na figura 5.13(b) as temperaturas externa e da sala vizinha ao longo do dia. É possível verificar que a temperatura se manteve amena, com picos de no máximo  $25^{\circ}\text{C}$ , o que não explica o aumento do erro na temperatura da sala.

Pode-se observar na figura 5.13(c) o sinal de controle, o ciclo de trabalho e o comportamento do estado do sistema, ligando durante 3h e desligando na 1h seguinte.

Finalmente, quando analisada a quantidade de pessoas durante o dia, verifica-se uma grande quantidade de pessoas na sala entre o período de 9h e 15h, mantendo até um número de nove pessoas ao mesmo tempo na sala. Dessa forma, é possível confirmar que esta é a perturbação dominante na situação em questão, o que comprova na prática que é uma perturbação que pode ser tratada de forma a melhorar o desempenho do sistema.

Os índices PMV e PPD foram calculados para ilustrar o conforto térmico do ambiente. A figura 5.13(e) mostra o valor do PMV ao longo do experimento. É fácil notar que no período de maiores perturbações tanto com a temperatura externa e com a sala vizinha quanto de número de pessoas, o índice PMV apresentou pior performance. Houve um pico por volta das 15h, por exemplo, em que o PMV atingiu 0,23 aproximadamente.

A figura 5.13(f) mostra o comportamento do índice PPD. Pode-se ver que, em um primeiro momento com muito perturbação, a predição do número de pessoas insatisfeitas é maior. Conforme as perturbações diminuem, o predição do número de insatisfeitos decai.

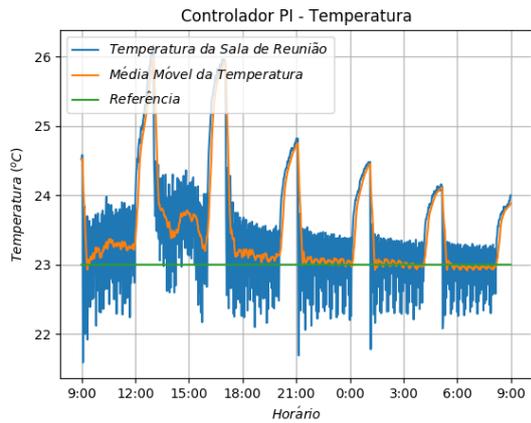
Em seguida, mediu-se o gasto energético e calculou-se o índice de dificuldade do teste. O gasto energético foi de  $6,77\text{kWh}$ , enquanto que o IBT foi calculado com base em (4.1), dando 3,37. Vale ressaltar que apesar do IBT ter dado um valor abaixo da média, este ainda representa um caso onde muitas pessoas serviram de perturbação para o sistema, todavia, as temperaturas externa e da sala vizinha, se mantiveram relativamente baixas. Já o erro quadrático médio calculado para esse ensaio foi  $0,49^{\circ}\text{C}$ .

#### 5.4.2.2 Testes de 8h

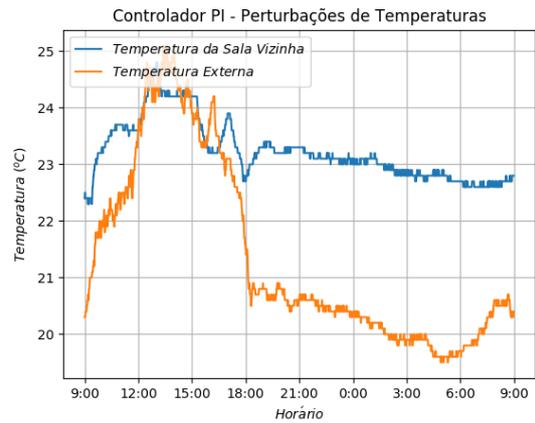
Como visto na seção 5.4.2.1, as perturbações de temperatura são agravantes no comportamento do sistema e atrapalham a resposta do controlador PI. Portanto, para contornar esta situação, implementou-se o controlador PI descrito em 4.6.2.2.

É possível notar na figura 5.14(a) que com a mudança feita, agora o erro passa ser bem menor, com a média móvel variando cerca de  $0,3^{\circ}\text{C}$ . Contudo, percebe-se um fenômeno de que o controlador está errando para baixo na primeira metade do experimento.

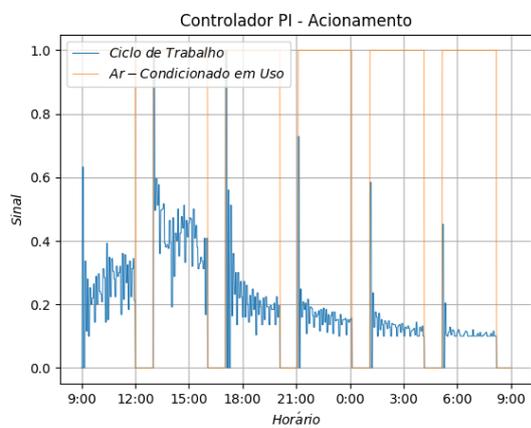
Quando verifica-se a figura 5.14(b), percebe-se uma temperatura externa e da sala vizinha inferiores ou muito próximas à referência na primeira metade do ensaio, o que fazia com que o



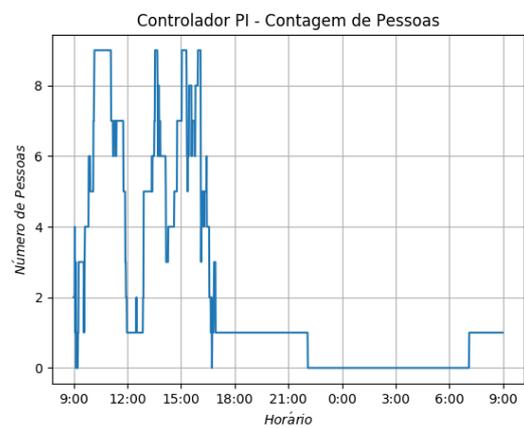
(a) Temperatura da Sala de Reuniões.



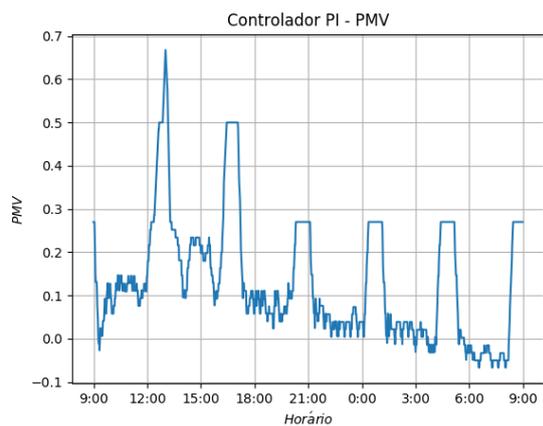
(b) Temperaturas Externas e da Sala Vizinha.



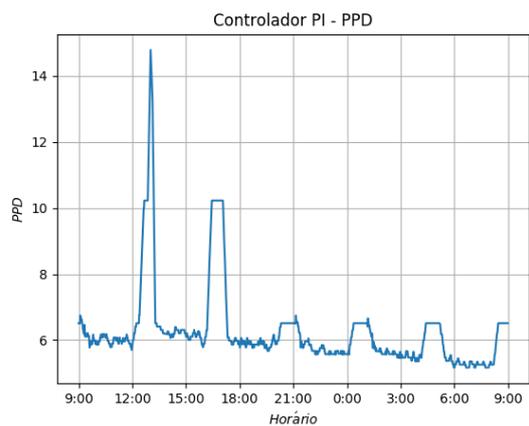
(c) Acionamento do Compressor.



(d) Número de Pessoas.

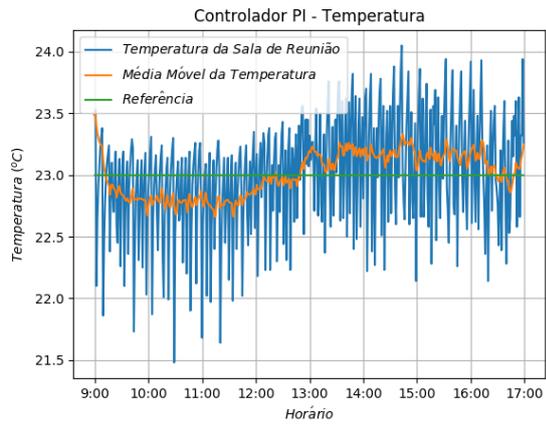


(e) Índice PMV

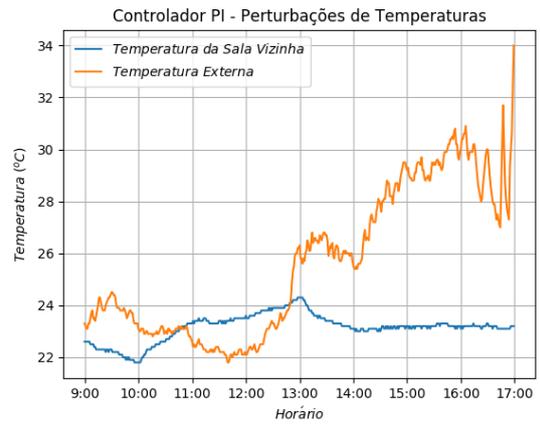


(f) Índice PPD

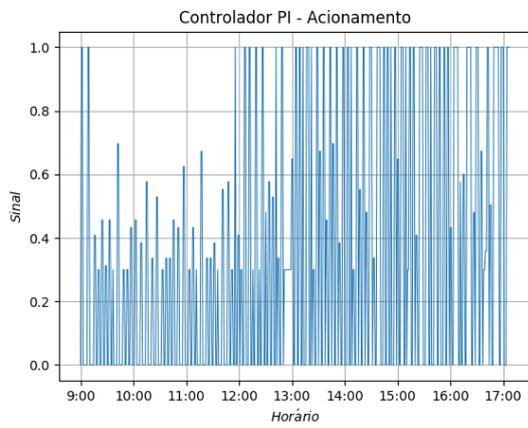
Figura 5.13: Controlador PI (24h) com alto número de pessoas.



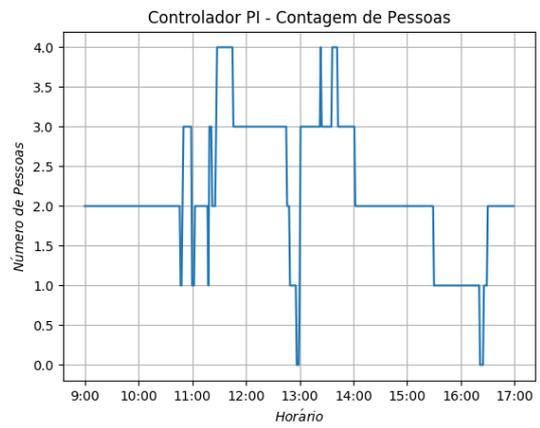
(a) Temperatura da Sala de Reuniões.



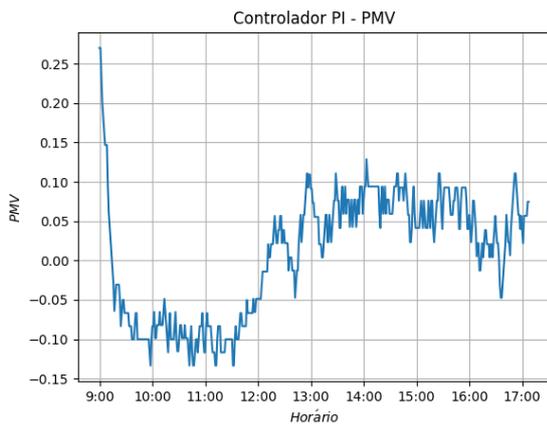
(b) Temperaturas Externas e da Sala Vizinha.



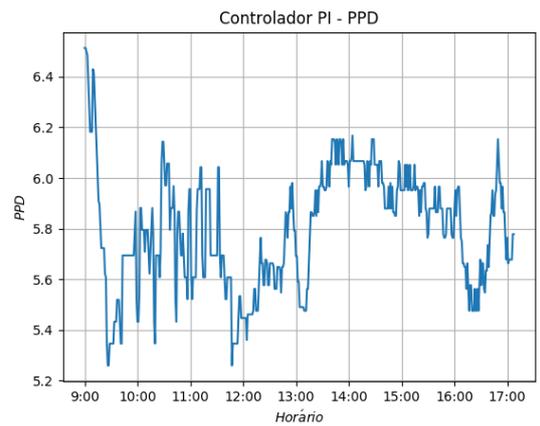
(c) Acionamento do Compressor.



(d) Número de Pessoas.



(e) Índice PMV



(f) Índice PPD

Figura 5.14: Controlador PI (8h).

sistema resfriasse rapidamente quando ligado, prejudicando o erro do sistema, uma vez que a utilização do ar-condicionado se mostra mais adequada para dias mais quentes. Contudo, a partir das 13h, a temperatura externa passou a aumentar substancialmente até o fim do experimento.

O acionamento do sistema pode ser visto de acordo com o ciclo de trabalho na figura 5.14(c). Através do estado do sistema, é possível conferir que o sistema se mantém ligado durante as 8h de experimento. E que na primeira metade, este aciona o compressor bem menos do que na segunda.

Já em relação ao número de pessoas presente na figura 5.14(d), nota-se um número baixo de pessoas durante o dia, gerando picos de 4 ocupantes. O horário entre 13h e 14h mostra um dos picos do dia, e nota-se que nesse horário, há um erro grande na temperatura da sala e perturbações de temperatura não tão grandes. Esse fenômeno é um indício da perturbação gerada pelo número de pessoas influenciando a resposta.

Ao observar a figura 5.14(e), nota-se que durante a primeira metade do ensaio o índice PMV assumiu valores negativos, enquanto que na segunda metade o PMV aumentou chegando a valores próximos à 0,14. A figura 5.14(f) mostra que a porcentagem predita de insatisfeitos atingiu valores entre 5,2% e 6,2% em regime permanente.

Finalmente, mediu-se o gasto energético e calculou-se o índice de dificuldade do teste. O gasto foi de 6,16 *kWh*. Pelo cálculo da equação (4.1), conclui-se que a dificuldade estimada do ensaio foi 2,7. Vale ressaltar que o IBT só é considerado para perturbações de temperatura superiores a 23°C, portanto, a primeira metade praticamente não influenciou neste índice. Já o RMSE calculado foi igual a 0,42 °C, todavia, o erro se torna difícil de se relacionar com o baixo IBT, devido a baixa representatividade.

## 5.5 Controlador Antecipativo

O modelo utilizado para simulação do controlador antecipativo foi apresentado na figura 4.11. Após a simulação, o controlador antecipativo foi implementado para executar no *Raspberry Pi*.

### 5.5.1 Simulação do Controlador Antecipativo

O resultado da simulação com o controlador antecipativo está disponível na figura 5.15. Pode-se notar que o comportamento é bem parecido com o comportamento do PI quando não há um número grande de pessoas na sala, ou seja, quando há até umas três pessoas.

Porém, conforme o número de pessoas aumenta, o PI não mostra-se incapaz de acompanhar a referência enquanto que o antecipativo mostra-se capaz. Pode-se analisar o período entre 170 e 300 minutos, período em que o número de pessoas foi até oito, e perceber que o controlador antecipativo acompanhou a referência. A temperatura da sala vizinha diminuiu nesse período, o que também foi um fator que ajudou em seu desempenho.

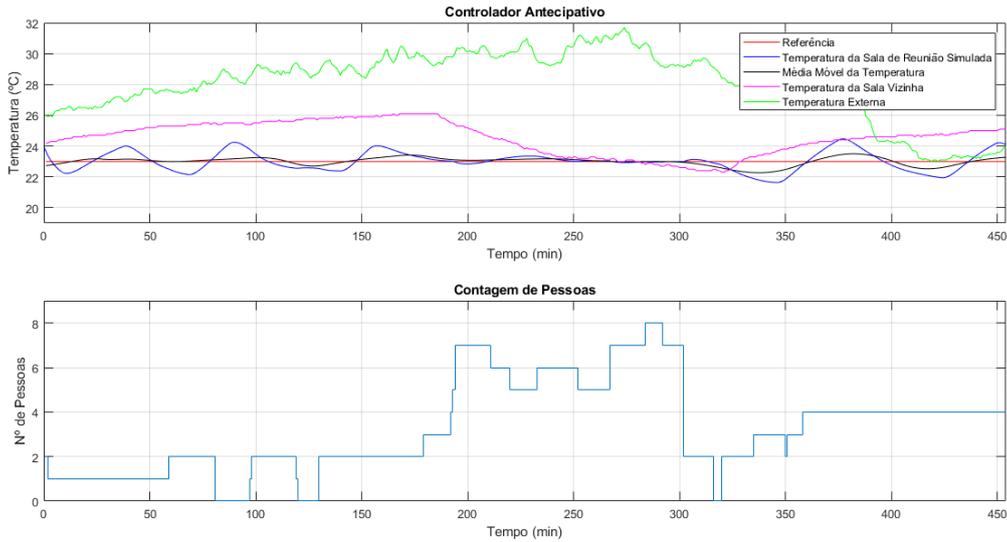


Figura 5.15: Resultado da simulação do sistema com o controlador antecipativo.

## 5.5.2 Controlador Antecipativo no Sistema Real

Foram feitos dois tipos testes do controlador antecipativo: testes de 24h e testes de 8h.

### 5.5.2.1 Testes de 24h

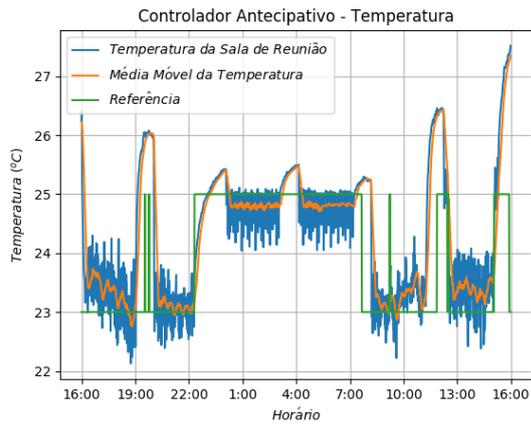
Os testes de 24h foram realizados com ciclos de acionamento. O ar-condicionado é ligado por 3h e depois permanece desligado por 1h, em seguida, repete o ciclo até completar as 24h.

A figura 5.16(a) mostra a temperatura da sala de reunião durante o experimento. Uma das grandes vantagens do controlador desenvolvido é que pode-se saber quando não há pessoas na sala. Então, quando a sala está vazia, pode-se elevar a referência e, conseqüentemente, reduzir o gasto energético ou pode-se, até mesmo, desligar o ar-condicionado. Na figura 5.16(a), vê-se que a referência foi elevada para 25 °C quando não havia ninguém na sala de reunião.

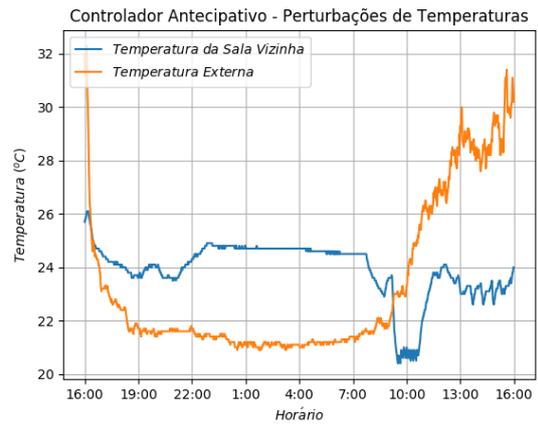
A figura 5.16(b) mostra o comportamento das temperaturas externa e da sala vizinha durante o ensaio. Já o acionamento do atuador pode ser visto na figura 5.16(c). Nela pode-se ver que quando não há pessoas na sala e a referência é elevada, o ciclo de trabalho diminui conforme o esperado.

O número de pessoas pode ser visto na figura 5.16(d). Nesse gráfico, vê-se que entre 16h e 20h assim como entre 9h e 15h foram os períodos de maior utilização da sala, ou seja, períodos com maior número de pessoas.

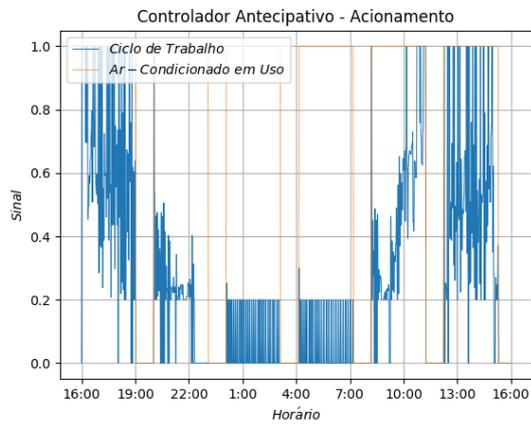
Ao associar a informação do número de pessoas com o gráfico já apresentado na figura 5.16(a), pode-se ver que entre 16h e 19h, apesar do grande número de pessoas, a saída se aproxima da referência. Já entre 13h e 17h, a temperatura fica deslocada da referência provavelmente devido a perturbações das temperaturas externa e da sala vizinha.



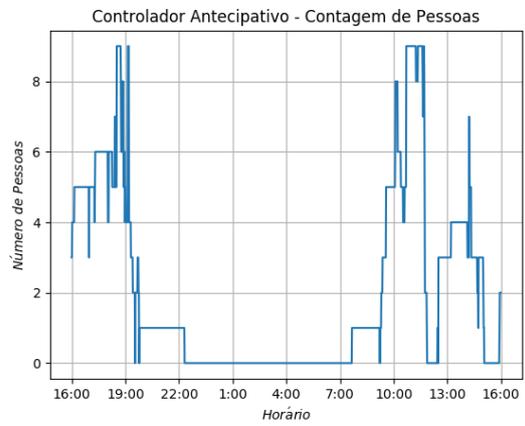
(a) Temperatura da Sala de Reuniões.



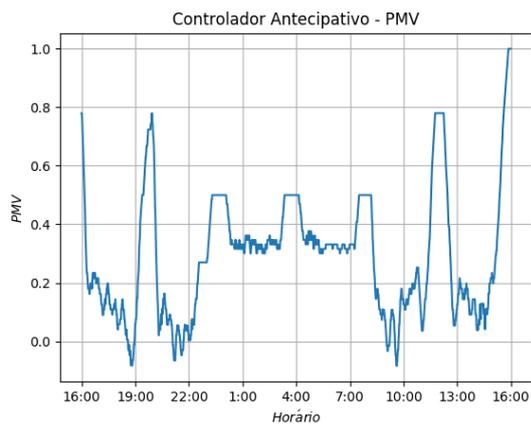
(b) Temperaturas Externas e da Sala Vizinha.



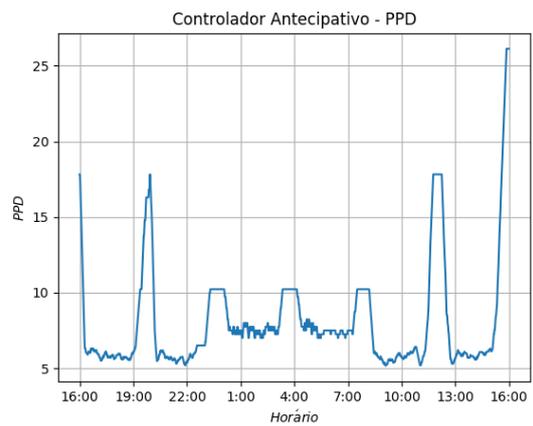
(c) Acionamento do Compressor.



(d) Número de Pessoas.



(e) Índice PMV



(f) Índice PPD

Figura 5.16: Controlador antecipativo (24h). Quando a sala está em uso, a referência é 23°C. Caso contrário, a referência é 25°C.

Percebeu-se, então, a necessidade de corrigir o ponto de operação do sistema para esses casos de alta perturbação de temperaturas externa e da sala vizinha conforme havia sido feito para o controlador PI.

Pode-se ver na figura 5.16(e), o comportamento do índice PMV ao longo do experimento. Vê-se que o seu valor não ultrapassou o limite de 0,2 durante o estado permanente. Já a mudança de referência quando o número de pessoas é zero acaba por aumentar o PMV para 0,4, mas não representa a real sensibilidade do sistema.

Já a figura 5.16(f), mostra a predição de pessoas insatisfeitas. Pode-se notar que o experimento obteve um resultado positivo visto que na maior parte do tempo esse índice apresentou valores inferiores a 6%.

Ao final do ensaio, o gasto energético foi 8,62 *kWh*. Além disso, foi utilizada a equação (4.1) para calcular o IBT, a dificuldade estimada do dia foi 6,3. Já o RMSE foi de 0,48.

### 5.5.2.2 Testes de 8h

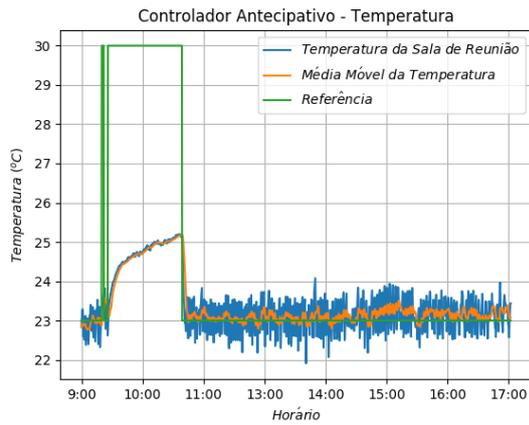
O teste de 8h contém apenas um ciclo de acionamento e tem como objetivo observar o efeito do controlador antecipativo no dia-a-dia normal da sala de reunião. O ar-condicionado foi ligado às 9h e desligado às 17h.

Pode-se ver, na figura 5.17(a), que a temperatura da sala de reunião seguiu a referência definida de 23 °C sempre que haviam pessoas na sala. Segundo a figura 5.17(d), das 09h20 até 11h40 não haviam pessoas na sala, então, a referência foi modificada para 30 °C com o objetivo de reduzir o gasto energético. Ao relacionar com a figura 5.17(c), que mostra o ciclo de trabalho do PWM, verifica-se que o sistema não é acionado durante o período mencionado.

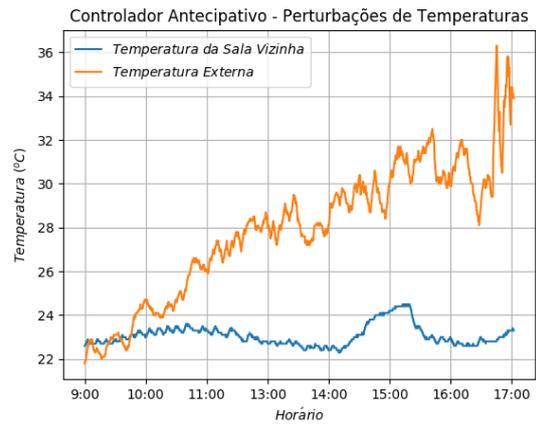
É importante notar também que a correção do ponto de operação reduziu o efeito das perturbações de temperatura externa e da temperatura da sala do Lara na saída do sistema. Mesmo em períodos com temperaturas mais altas, a saída do sistema ainda seguiu, de forma aproximada, a referência. A partir das 14h50, vê-se um leve deslocamento da curva de temperatura a cima da referência, que se deve a um aumento da temperatura externa e da temperatura da sala vizinha. Porém, esse deslocamento não ultrapassa 0,5 °C.

Ao relacionar as informações de número de pessoas, mostrada na figura 5.17(d), com a temperatura da sala de reunião, mostrada na figura 5.17(a), vê-se que a influência do número de pessoas no sistema é praticamente nulo. Mesmo nos picos de quantidade de pessoas, que chegava a conter 6 ou 7 pessoas na sala ao mesmo tempo, o sistema seguiu a referência conforme especificado.

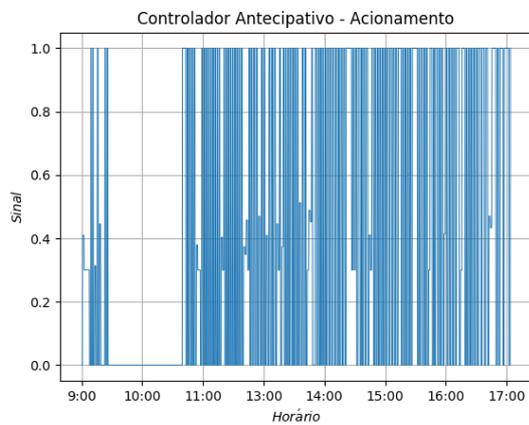
A figura 5.17(e) mostra o comportamento do índice PMV e a figura 5.17(f) mostra o comportamento do índice PPD. Vê-se que o resultado do PMV foi muito bom, não ultrapassando um módulo de 0,15. Ao observar o gráfico do PPD pode-se ver que a porcentagem predita de pessoas insatisfeitas foi entre 5% e 6% no regime permanente. Pode-se observar ainda uma aumento nos dois índices por volta das 10h, resultado da mudança de referência devido ao número de ocupantes igual a zero.



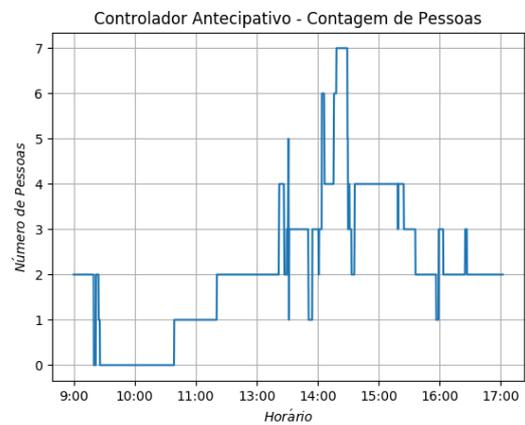
(a) Temperatura da Sala de Reuniões.



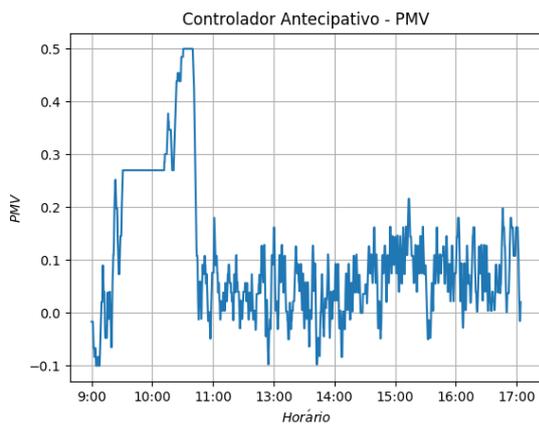
(b) Temperaturas Externas e da Sala Vizinha.



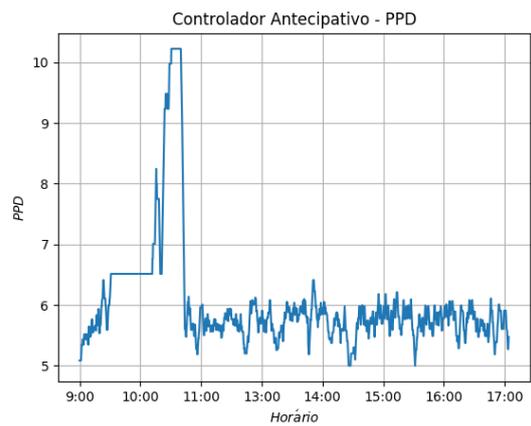
(c) Acionamento do Compressor.



(d) Número de Pessoas.



(e) Índice PMV



(f) Índice PPD

Figura 5.17: Controlador antecipativo (8h). Quando a sala está em uso, a referência é 23°C. Caso contrário, a referência é 30°C.

Tabela 5.1: Comparação entre os Tipos de Controladores.

Controlador	Período (h)	Consumo (kWh)	IBT	RMSE
L/D	24	13,37	3,95	0,51
PI - Baixo N <sup>o</sup> de Pessoas	24	7,9	5,47	0,49
PI - Alto N <sup>o</sup> de Pessoas	24	6,77	3,36	0,49
Antecipativo	24	8,62	6,30	0,48
L/D	8	7,92	8,63	0,42
PI	8	6,16	2,70	0,42
Antecipativo - Sem Mud. de Referência	8	6,00	6,06	0,35
Antecipativo	8	5,81	8,71	0,37

Ao fim do experimento, mediu-se o gasto energético. Esse gasto foi de 5,81 kWh. Pelo cálculo da equação (4.1), conclui-se que a dificuldade estimada do ensaio foi 8,71, sendo o ensaio mais difícil realizado. Além disso, obteve-se um RMS de 0,37.

É importante diferenciar se o gasto energético foi reduzido devido ao controlador antecipativo ou devido ao aumento da temperatura de referência quando não há pessoas na sala. Portanto, também foi feito um ensaio sem considerar esse diferencial do sistema desenvolvido.

O resultado desse ensaio está disponível na figura 5.18(a). Ao relacioná-la com a figura 5.18(b), vê-se uma forte influência da temperatura externa e da sala vizinha na saída.

Esse ensaio utilizou 6,00 kWh. Além disso, o ensaio obteve um IBT igual a 6,07. Com um RMSE de 0,35 °C.

## 5.6 Comparação

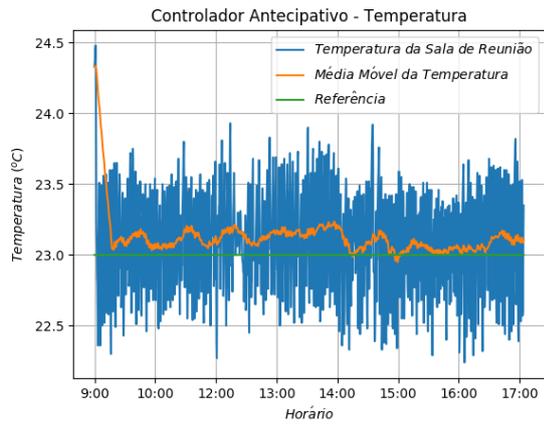
Uma vez que todos os resultados foram obtidos, decidiu-se criar uma tabela para simplificar a comparação entre os diferentes tipos de controladores desenvolvidos. O comparativo pode ser visto na tabela 5.1.

Pode-se notar que o tradicional controlador liga-desliga apresentou-se inferior aos controladores PI e Antecipativo tanto em relação ao RMSE quanto ao gasto energético.

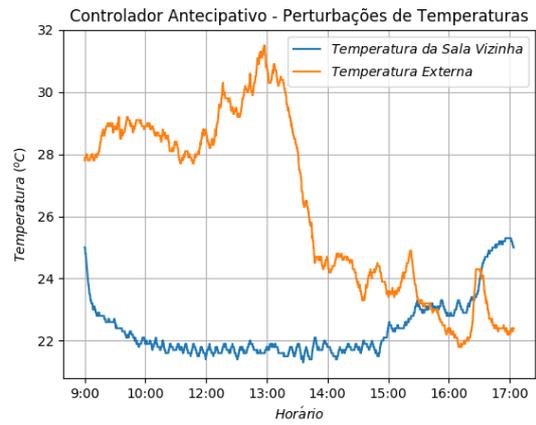
Quando comparados os dois controladores PI de 24h, verifica-se que apesar da perturbação do número de pessoas ser maior em um deles, há uma grande diferença entre os IBTs, indicando que o dia do PI com baixo número de pessoas foi mais difícil do que o com alto número. Isso explica a diferença de gasto energético e do RMSE, que tendia a ser maior para o segundo caso.

Tanto nos ensaios de 24h quanto nos de 8h, o controlador antecipativo mostra uma melhoria significativa no RMSE mesmo sob um dia difícil, indicado pelo seu IBT. Entretanto, mostrou um gasto energético superior ao PI no experimento de 24h. Isso pode ser explicado devido ao alto nível de perturbação do dia.

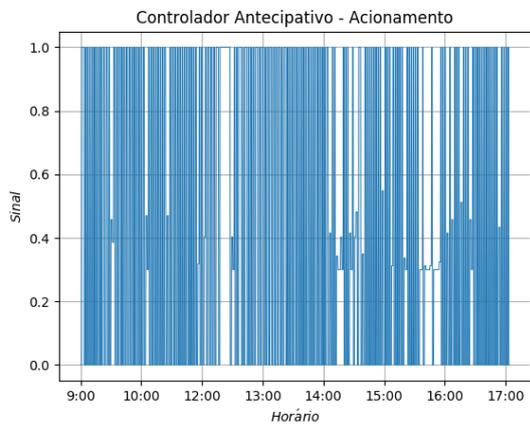
Com as correções finais, os experimentos de 8h foram feitos e neles é possível verificar uma diminuição expressiva no RMSE em todos os casos. É válido notar que o controlador antecipativo



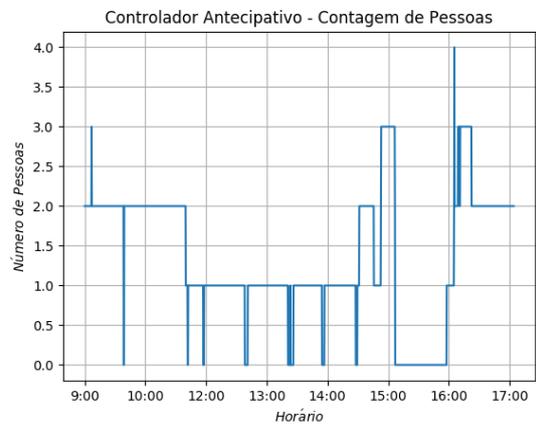
(a) Temperatura da Sala de Reuniões.



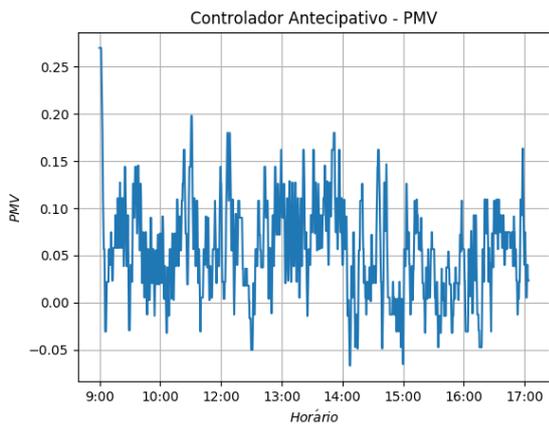
(b) Temperaturas Externas e da Sala Vizinha.



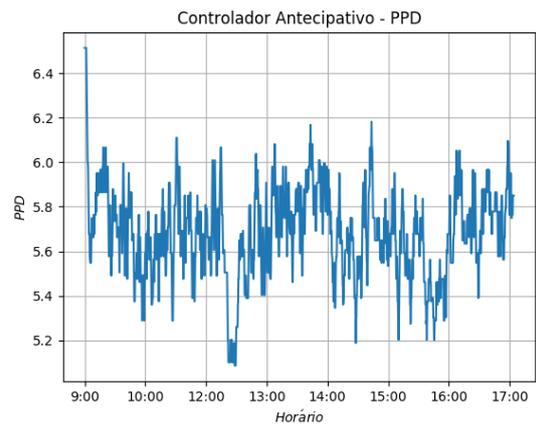
(c) Acionamento do Compressor.



(d) Número de Pessoas.



(e) Índice PMV



(f) Índice PPD

Figura 5.18: Controlador antecipativo (8h). Sem mudança de referência conforme o número de ocupantes da sala.

Tabela 5.2: Melhoria Percentual do Controle Antecipativo.

<b>Melhoria do Controle Antecipativo sem Mud. de Referência</b>		
	<b>Melhoria no Erro</b>	<b>Melhoria Energética</b>
<b>Liga-Desliga</b>	16,66%	24,24%
<b>PI</b>	16,66%	2,6%
<b>Melhoria do Controle Antecipativo com Mud. de Referência</b>		
	<b>Melhoria no Erro</b>	<b>Melhoria Energética</b>
<b>Liga-Desliga</b>	11,90%	26,64%
<b>PI</b>	11,90%	5,8%

se mostrou extremamente eficiente, vez que sob um dia difícil, conseguiu um menor RMSE e um menor gasto energético.

Finalmente, montou-se a tabela 5.2 para demonstrar a melhoria que a técnica proposta efetivamente gerou no sistema. Para os resultados percentuais, utilizou-se as equações (4.3) e (4.4). Além disso, utilizou-se os ensaios de 8h, que são mais representativos para a aplicação real do modelo proposto.

Com os dados percentuais, é possível verificar que o controle antecipativo possui uma grande melhoria percentual, mesmo com o mais alto IBT, conseguindo erros e gastos energéticos menores quando comparado com os demais controladores tradicionais, principalmente para os casos com alto número de pessoas.

A próxima seção aborda as conclusões sobre o trabalho desenvolvido, além das sugestões para trabalhos futuros.

# Capítulo 6

## Conclusão

Aliar uma melhoria no conforto térmico a uma redução do gasto energético é um dos grandes desafios da automação residencial e dos prédios inteligentes. Visando atingir esses objetivos, são estudadas e desenvolvidas várias técnicas de controle.

A proposta principal deste trabalho é desenvolver um controlador antecipativo que, baseado na perturbação térmica causada pelas pessoas presentes no ambiente, controla o sistema de refrigeração ar-condicionado. Além disso, comparar o novo controlador com os controladores clássicos é fundamental para mensurar as suas vantagens.

A primeira contribuição do projeto foi um algoritmo de contagem do número de pessoas. Esse algoritmo utiliza técnicas de visão computacional e processamento de imagens para analisar a entrada e a saída de pessoas da sala. O algoritmo demonstrou-se acurado para situações normais, ou seja, em que as pessoas não estão muito próximas umas das outras e quando a entrada não é muito rápida.

Outra contribuição foi projetar e implementar um controlador antecipativo. O controlador antecipativo associa as informações de carga térmica e do número de pessoas, mantendo um alto desempenho mesmo com grande variação da ocupação.

Ao compará-lo com os controladores liga-desliga e PI, vê-se que ele obteve uma melhora trazendo uma redução significativa tanto no erro quanto no gasto energético, mesmo quando submetido a um ensaio mais severo que os demais. Se mostrando uma solução interessante para ambientes prediais, reduzindo o custo energético e propiciando maior conforto térmico.

### 6.1 Sugestões para Trabalhos Futuros

Uma das sugestões para os trabalhos futuros é implementar um sistema supervisório, para armazenar séries históricas e auxiliar na visualização dos dados. Outro ponto importante é a portabilidade que esse tipo de sistema pode oferecer, tornando possível que uma informação de interesse sobre o sistema seja obtida por qualquer dispositivo conectado à rede do supervisório. O *OpenHAB* (*Open Home Automation Bus*) é um exemplo de sistema supervisório aberto focado

em automação residencial. Sua função principal é a integração de diferentes sistemas de forma a produzir uma solução interoperável. Utilizando o *OpenHAB* pode-se implementar diferentes regras de automação a partir de uma interface gráfica amigável [45].

Outra sugestão é a utilização de redes neurais artificiais (RNA) para auxiliar na detecção e contagem de pessoas. Por vezes, notou-se que o sistema utilizado de detecção por subtração de fundo apresentava erros pontuais. Um exemplo de erro é quando duas pessoas entram muito próximas. Outro é quando entra-se na sala com algum objeto longe do corpo, como uma cadeira.

Uma possibilidade é a utilização de um estado da arte na área de inteligência computacional, como o *YOLO* (*You Only Look Once*) [46], que consiste num identificador de objetos em tempo real com um processamento de imagens superior à média. A rede neural foi estudada e implementada em um computador, mas por limitações de tempo e de *hardware*, não chegou a ser testada no *Raspberry Pi*, mas apresentava resultados interessantes para a contagem de pessoas.

Porém, para possibilitar o uso destes tipos de técnicas de redes neurais no *Raspberry Pi* faz-se necessário o uso de aceleradores de *hardware*. Isso ocorre porque o sistema térmico deve ser controlado em tempo-real e o *hardware* do *Raspberry* não é suficiente, mesmo que já utilizando unicamente o modelo já treinado.

Além disso, o algoritmo de subtração de fundo pode se tornar mais robusto se for implementada alguma técnica para evitar a detecção de fantasmas. Esses fantasmas são falsos positivos que ocorrem, principalmente, devido aos movimentos das sombras, mas também podem ocorrer devido a objetos presentes na imagem do fundo. Existem diversas técnicas para contornar esse problema, alguns exemplos são a utilização da informação de similaridade de cores para detectar e remover os fantasmas [47] ou usufruir de descritores estatísticos para determinar os ruídos em uma sequência de vídeo [48]. Além disso, a subtração de fundo também pode ser associada a outras técnicas clássicas como o HOG, para isso seria necessário fazer uma transformação de coordenadas para analisar a imagem sobre outra perspectiva.

Já pensando em uma melhora do sistema de controle, seria interessante associar o controle antecipativo desenvolvido neste trabalho a um controle adaptativo. O funcionamento do ar-condicionado depende da temperatura externa e da temperatura das salas vizinhas. Uma das grandes dificuldades deste trabalho foi distinguir o comportamento causado pelo número de pessoas na sala do comportamento causado pelo fluxo de calor entre os ambientes. Por isso, um trabalho mais completo que levasse todas essas variáveis em conta demonstraria uma melhora significativa no conforto térmico.

Outra proposta é utilizar o sistema evaporativo. O sistema condicionador de ar disponível na sala de reunião do LARA já conta com um sistema evaporativo para auxiliar no resfriamento da sala e reduzir o gasto energético. Associar o sistema evaporativo com o controlador antecipativo desenvolvido resultaria em uma maior eficiência energética.

Por uma limitação do ar-condicionado disponível no LARA, só existem dois estados: ligado e desligado. Por tanto, para controlar a temperatura é necessário implementar um PWM. Porém, o PWM deve ter um período alto, a fim de não danificar o compressor ao mudar de estado muito

rápido. Apesar do sistema térmico possuir uma dinâmica lenta, o período do PWM ainda influencia nos resultados. Para reduzir ou eliminar esse problema, sugere-se uma modelagem no domínio discreto ou o uso de um ar-condicionado com inversor de frequência em que seja possível um controle contínuo da velocidade do compressor.

# REFERÊNCIAS BIBLIOGRÁFICAS

- [1] LAMBERTS, R. *Conforto e Stress Térmico*. Apostila — Universidade Federal de Santa Catarina: Centro Tecnológico do Departamento de Engenharia Civil, 2011.
- [2] DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: IEEE. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. San Diego, CA, USA, 2005. v. 1, p. 886–893.
- [3] VIOLA, P.; JONES, M. Rapid object detection using a boosted cascade of simple features. In: IEEE. *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. Kauai, HI, USA, 2001. v. 1, p. I–I.
- [4] BARAJAS, F. M. *People Counter*. Janeiro 2017. Site, Disponível em: <https://fedemejia.com/> [Acessado em Julho 2018].
- [5] UFRGS. *Ação Liga-Desliga*. 2018. Site, Disponível em: <http://www.ece.ufrgs.br/~jmgomes/pid/Apostila/apostila/node20.html> [Acessado em Dezembro 2018].
- [6] EMBARCADOS. *Diferenças entre os controladores proporcional e integral*. 2018. Site, Disponível em: <https://www.embarcados.com.br/diferencas-controlador-proporcional-e-integral/> [Acessado em Novembro 2018].
- [7] RASPBERRY PI FOUNDATION. *Raspberry Pi 3 Model B*. Fevereiro 2016. Site, Disponível em: <https://www.raspberrypi.org/products/raspberry-pi-3-model-b/> [Acessado em Maio 2018].
- [8] RASPBERRY PI FOUNDATION. *Camera Module v2*. Abril 2016. Site, Disponível em: <https://www.raspberrypi.org/products/camera-module-v2/> [Acessado em Maio 2018].
- [9] ARDUINO. *Arduino Mega 2560 rev3*. 2018. Site, Disponível em: <https://store.arduino.cc/usa/arduino-mega-2560-rev3> [Acessado em Maio 2018].
- [10] ARAUJO, H. A. S. de; MELO, M. C. C. *Controle Adaptativo da Climatização Predial para Eficiência Energética*. Trabalho de Graduação — Universidade de Brasília: Faculdade de Tecnologia, 2013.
- [11] SENSIRION. *Datasheet SHT7x (SHT71, SHT75)*. 5. ed. Suíça, Dezembro 2011. Datasheet, Disponível em: [http://www.mouser.com/ds/2/682/Sensirion\\_Humidity\\_SHT7x\\_Datasheet\\_V5-469726.pdf](http://www.mouser.com/ds/2/682/Sensirion_Humidity_SHT7x_Datasheet_V5-469726.pdf) [Acessado em Maio 2018].

- [12] ERRIEZ. *DHT22 - AM2302/AM2303 temperature and humidity sensor library for Arduino*. 2018. Site, Disponível em: <https://github.com/Erriez/ESP8266UbidotsSensors/blob/master/lib/ErriezDHT22/README.md> [Acessado em Novembro 2018].
- [13] HU INFINITO. *Módulo Sensor Magnético*. 2018. Site, Disponível em: <http://www.huinfinito.com.br/sensores/20-modulo-sensor-magnetico.html> [Acessado em Novembro 2018].
- [14] ARAUJO, A. M. *Controle TITO via desacoplamento de malhas de climatização híbrida*. Trabalho de Graduação — Universidade de Brasília: Faculdade de Tecnologia, 2016.
- [15] OLIVEIRA, F. A. R. de. *Instrumentação e identificação de um ambiente predial visando controle preditivo do conforto térmico*. Dissertação (Mestrado) — Faculdade de Tecnologia, Universidade de Brasília, 2010.
- [16] CARRIER. *Carrier*. Site, Disponível em: <http://carrierdobrasil.com.br/> [Acessado em Dezembro 2018].
- [17] FILIPEFLOP. *Funcionamento da Câmera Raspberry Pi*. Dezembro 2014. Site, Disponível em: <https://www.filipeflop.com/blog/modulo-camera-raspberry-pi/> [Acessado em Maio 2018].
- [18] ASHRAE. *Standard 55 Thermal Environment Conditions for Human Occupancy*. Atlanta, Georgia, USA, 2004. Norma.
- [19] LIU, T. *Digital-output relative humidity temperature sensor/module DHT22*. 1. ed. USA. Datasheet, Disponível em: <https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf> [Acessado em Novembro 2018].
- [20] PADUA, I. H. D. *Caracterização de edifícios inteligentes: um caso exemplo*. Dissertação (Mestrado) — Pontifícia Universidade Católica de Minas Gerais, Minas Gerais, 2006.
- [21] ROTOPLAST. *A climatização do ambiente de trabalho pode influenciar na produtividade do funcionário?* Site, Disponível em: <http://www.rotoplast.com.br/blog/a-climatizacao-do-ambiente-de-trabalho-pode-influenciar-na-productividade-do-funcionario/> [Acessado em Dezembro 2018].
- [22] FERRER, J. A.; GARRIDO, A. *Eficiência Energética em Edifícios*. 2013. ONUDI Observatório e Energias Renováveis para a América Latina e Caribe.
- [23] COELHO, D. F. B.; CRUZ, V. H. do N. Eficiência energética nas edificações. *São Paulo: Blur*, Artigo Científico — São Paulo, p. 61 –88, 2017.
- [24] IEA, I. E. A. *Annual Report*. 2013. IEA.
- [25] EPE, E. de P. E. *Balanco Energético Nacional 2015: ano base 2014*. 2015. EPE.

- [26] SILVA, M. *SMART BUILDING. A TECNOLOGIA APLICADA NA GESTÃO DO SEU IMÓVEL*. Agosto 2018. Site, Disponível em: <http://blog.bortoliniimoveis.com.br/2018/08/08/smart-building-a-tecnologia-aplicada-na-gestao-do-seu-imovel/> [Acessado em Dezembro 2018].
- [27] HENSEN, J. L. M. *On the thermal interaction of building structure and heating and ventilating system*. Tese (Doutorado) — Technische Universiteit Eindhoven, 1991.
- [28] DJONGYANG, N.; TCHINDA, R.; NJOMO, D. Thermal comfort: A review paper. v. 14, p. 2626–2640, 12 2010.
- [29] VIANNA, P. I. E. *Automação de um sistema de climatização híbrido com foco no processo evaporativo visando conforto térmico e uso racional da energia*. Trabalho de Graduação — Universidade de Brasília: Faculdade de Tecnologia, 2012.
- [30] FANGER, P. O. et al. Thermal comfort. analysis and applications in environmental engineering. *Thermal comfort. Analysis and applications in environmental engineering.*, Copenhagen: Danish Technical Press., 1970.
- [31] ISO. *Ergonomics of the thermal environment - analytical determination and interpretation of thermal comfort using calculation of the PMV e PPD indices and local thermal comfort criteria*. Geneva, Switzerland, 2005. Norma.
- [32] CREDER, H. *Instalação de Ar Condicionado*. Rio de Janeiro: LTC, 2004.
- [33] CORRIPIO, C. A. S. e A. B. *Princípios e Prática do Controle Automático de Processo*. Rio de Janeiro: LTC, 2008.
- [34] MATHWORKS. *Temperature Control in a Heat Exchanger*. 2018. Site, Disponível em: <https://www.mathworks.com/help/control/examples/temperature-control-in-a-heat-exchanger.html#d120e5718> [Acessado em Dezembro 2018].
- [35] SUN, J. *Dynamics and Control of Switched Electronic Systems*. Rensselaer Polytechnic Institute: Department of Electrical, Computer, and Systems Engineering, 2012.
- [36] SOARES JÚNIOR, L. R. *Controle e Acionamento*. 2018. Site, Disponível em: <http://www2.ene.unb.br/lelio/cap12.pdf> [Acessado em Julho 2018].
- [37] BOAVENTURA, D. B. *Automação com OpenHab de sistema de climatização híbrida visando eficiência energética*. Trabalho de Graduação — Universidade de Brasília: Faculdade de Tecnologia, 2016.
- [38] OMRON. *Solid State Relay G3MB*. 1. ed. Estados Unidos. Datasheet, Disponível em: [https://cdn.awsli.com.br/78/78150/arquivos/rele\\_ssr.pdf](https://cdn.awsli.com.br/78/78150/arquivos/rele_ssr.pdf) [Acessado em Maio 2018].
- [39] TEAM, O. *OpenCV*. 2018. Site, Disponível em: <https://opencv.org/> [Acessado em Julho 2018].

- [40] FLORES, J. L. O. *Sistema Híbrido de Climatização Visando Conforto Térmico e Eficiência Energica*. Dissertação (Mestrado) — Faculdade de Tecnologia, Universidade de Brasília, 2009.
- [41] MOTTA, W. B. A. *CONTROLE DE CLIMATIZAÇÃO HÍBRIDA VISANDO A REDUÇÃO DO CONSUMO DE ENERGIA*. Trabalho de Graduação — Universidade de Brasília: Faculdade de Tecnologia, 2018.
- [42] PYPI. *Simple-PID 0.1.4*. 2018. Site, Disponível em: <https://pypi.org/project/simple-pid/> [Acessado em Novembro 2018].
- [43] BARAJAS, F. M. *People Counter Video*. Junho 2016. Site, Disponível em: <https://www.dropbox.com/s/7mbo35zrxdl1aefz/peopleCounter.avi?dl=0> [Acessado em Julho 2018].
- [44] PATIAL, V. *Videos*. Maio 2016. Site, Disponível em: <https://drive.google.com/drive/folders/OB3LatSCwKo2benZyVXhKLXV6ROU> [Acessado em Julho 2018].
- [45] COCULA, J. *Who we are*. Fevereiro 2017. Site, Disponível em: <https://github.com/openhab/openhab1-addons/wiki#overview> [Acessado em Julho 2018].
- [46] REDMON, J.; FARHADI, A. Yolov3: An incremental improvement. *arXiv*, 2018.
- [47] KADIM, Z. et al. Ghost removal for background subtraction using color similarity comparison. p. 24–, 12 2011.
- [48] JUNG, C. Efficient background subtraction and shadow removal for monochromatic video sequences. *Multimedia, IEEE Transactions on*, v. 11, p. 571 – 577, 05 2009.
- [49] BERTOLETI, P. *Primeiros passos com Linux na Raspberry Pi 3*. Agosto 2016. Site, Disponível em: <https://www.embarcados.com.br/primeiros-passos-linux-na-raspberry-pi-3/> [Acessado em Maio 2018].
- [50] RASPBERRY PI FOUNDATION. *NOOBS*. 2018. Site, Disponível em: <https://www.raspberrypi.org/downloads/noobs/> [Acessado em Maio 2018].
- [51] SD ASSOCIATION. *SD Memory Card Formatter*. Site, Disponível em: [https://www.sdcard.org/downloads/formatter\\_4/](https://www.sdcard.org/downloads/formatter_4/) [Acessado em Maio 2018].
- [52] RASPBERRY WEB SERVER. *Set up an SQLite database on a Raspberry Pi*. 2014. Site, Disponível em: <http://raspberrypiwebserver.com/sql-databases/set-up-an-sqlite-database-on-a-raspberry-pi.html> [Acessado em Maio 2018].
- [53] ZETCODE. *SQLite Python tutorial*. Novembro 2014. Site, Disponível em: <http://zetcode.com/db/sqlitepythontutorial/> [Acessado em Maio 2018].
- [54] ROSEBROCK, A. *Install guide: Raspberry Pi 3 + Raspbian Jessie + OpenCV 3*. Abril 2016. Site, Disponível em: <https://www.pyimagesearch.com/2016/04/18/install-guide-raspberry-pi-3-raspbian-jessie-opencv-3/> [Acessado em Junho 2018].

# APÊNDICES

# I. PROGRAMAS UTILIZADOS

## I.1 Programa Principal

---

```
##Controle Antecipativo por Estimativa de Carga Termica em Video
##Codigo Principal
##Trabalho de Graduacao
##Autores:
## Alexandre Saran
## Mariana Pimentel

# coding=utf-8

from peoplecounterlib import *
from sqlitelib import *
from controllerlib import *
import threading
import settings
import time
import datetime
import cv2

def data():
    global runEvent

    database = InitializeDatabase(str(timeHour))
    CreateTable(database)

    initialNumPeople = 2

    settings.initialNumPeople = initialNumPeople

    (tempMeetingRoom, humMeetingRoom, tempLara, tempExternal, doorSignal) = (0,0,0,0,0)

    while runEvent.is_set():
        numPeople = settings.cntUp-settings.cntDown + initialNumPeople
        print(TAG+'Numero de Pessoas: '+str(numPeople))

        # Calculates inicial time
        timestamp = time.time()
        dateTme =
            datetime.datetime.fromtimestamp(timestamp).strftime('%Y-%m-%d_%H-%M-%S')

        # Read temperatura e humidity sensors
```

```

for i in range (10):
    (readOk, tempMeetingRoom, humMeetingRoom, tempLara, tempExternal,
     doorSignal) = readTempHumDoor(tempMeetingRoom, humMeetingRoom, tempLara,
     tempExternal, doorSignal)
    if(readOk):
        break

dutyCycle = settings.dutyCycle

# Read compressor signal
compressorSignal = settings.compressorSignal
isOn = settings.isOn
print(TAG+'Esta Ligado: '+str(isOn))

InsertData(database, dateTime, tempMeetingRoom, humMeetingRoom, tempLara,
tempExternal, doorSignal, numPeople, compressorSignal, isOn, dutyCycle)
time.sleep(15)

def counter(timeHour, SAVE_RESULTS):
    try:
        PeopleCounter(0, 0, str(timeHour), SAVE_RESULTS)
    except KeyboardInterrupt:
        cv2.destroyAllWindows()

def controller():
    global runEvent
    lastOutput = 0
    cont = 1
    while runEvent.is_set():
        (lastOutput, cont) = Controller(lastOutput, cont)

SAVE_RESULTS = False

TAG = '(main) '

# Calculates inicial time
timestamp = time.time()
timeHour = datetime.datetime.fromtimestamp(timestamp).strftime('%Y-%m-%d_%H-%M-%S')

print(TAG+'data/hora: '+ str(timeHour))

settings.init()

runEvent = threading.Event()
runEvent.set()

```

```

threadPeopleCounter = threading.Thread(name='people_counter', target=counter, args =
    (str(timeHour), SAVE_RESULTS))
threadPeopleCounter.daemon = True
threadPeopleCounter.start()

threadPeopleData = threading.Thread(name='data', target=data)
threadPeopleData.start()

threadController = threading.Thread(name='controller', target=controller)
threadController.start()

try:
    while(True):
        time.sleep(0.3)
except KeyboardInterrupt:

    database = InitializeDatabase(str(timeHour))
    ReadTable(database)
    CloseTable(database)

    cv2.destroyAllWindows()

    runEvent.clear()
    threadPeopleCounter.join()
    threadPeopleData.join()
    threadController.join()

    writeRele(0, 30, "(controllerlib) ")

# print(TAG+'Final2: '+str(numberPeople))

print(TAG+'end')

```

---

Código I.1: Programa principal que inicia os processos paralelos de leitura e armazenamento de dados, de contagem do número de pessoas e de controle.

## I.2 Contagem do Número de Pessoas

### I.2.1 Biblioteca para Contagem do Número de Pessoas

---

```

##Controle Antecipativo por Estimativa de Carga Termica em Video
##Biblioteca para Identificacao e Contagem do Numero de Pessoas por Video
##Trabalho de Graduacao
##Autores:
## Alexandre Saran
## Mariana Pimentel

```

```

##Adaptado do Material de: Federico Mejia

# coding=utf-8

from cameralib import *
import settings
import numpy as np
import cv2
import personlib
import time
import datetime
import sys
import os

def PeopleCounter(cntUp, cntDown, name, saveResults):
    TAG = '(peoplecouterlib) '

    # Result's path on Raspberry Pi
    path = '../../Resultados/'+name
    videoName = name

    print(TAG+'caminho: '+ str(path))
    print(TAG+'nome do video: '+ str(videoName))

    if(saveResults):
        if not os.path.exists(path):
            os.makedirs(path)

    # Initialize the camera and grab a reference to the raw camera capture
    (camera,rawCapture) = InicializeCamera()
    settings.camera = camera

    # Allow the camera to warmup
    time.sleep(0.1)

    # Calculate the threshold to define if it is or not a person
    w = 320
    h = 240
    frameArea = h*w
    #areaTH = frameArea/250
    areaTH = frameArea/60
    areaTHSuperior = frameArea/3
    print(TAG+'threshold: '+str(areaTH))

    # Define up and down line
    #lineUp = int(2*(h/5))
    #lineDown = int(3*(h/5))

```

```

lineUp = int(13*(h/20))
lineDown = int(13*(h/20))

# After this lines memory can be free
#upLimit = int(1*(h/5))
#downLimit = int(4*(h/5))
upLimit = int(2*(h/20))
downLimit = int(18*(h/20))

# Calculate important points
print(TAG+"y da linha inferior:"+str(lineDown))
print(TAG+"y da linha superior:"+str(lineUp))
lineDownColor = (255,0,0)
lineUpColor = (0,0,255)

(pt1, pt2, pt3, pt4, pt5, pt6, pt7, pt8) = calculatePoints(w, lineUp, lineDown,
    upLimit, downLimit)
(ptsL1, ptsL2, ptsL3, ptsL4) = calculateLinePoints(pt1, pt2, pt3, pt4, pt5, pt6,
    pt7, pt8)

# Creates the background subtractor
fgbg = cv2.createBackgroundSubtractorMOG2(history = 500, varThreshold = 16,
    detectShadows = True)

# Necessary to apply filters and morfological transforms
kernelOp = np.ones((5,5),np.uint8)
kernelOp2 = np.ones((7,7),np.uint8)
kernelCl = np.ones((11,11),np.uint8)

persons = []
maxPAge = 5
pid = 1

cont = 1
for cap in camera.capture_continuous(rawCapture, format="bgr", use_video_port=True):
# Para um video continuo camera.capture_continuous(rawCapture, format="bgr",
    use_video_port=True)
    settings.cntDown = cntDown
    settings.cntUp = cntUp

    # Read a frame
    frame = cap.array

    for i in persons:
        i.age_one()

#####

```

```

#     PRE-PROCESS     #
#####

# Apply background subtraction
fgmask = fgbg.apply(frame)
fgmask2 = fgbg.apply(frame)

saveSubtractorImages(saveResults, path, videoName, cont, frame, fgmask)

try:
    (mask, mask2) = preProcess(fgmask, fgmask2, saveResults, path, videoName,
                               cont, kernelOp, kernelCl)
except:
    print (TAG+'para cima: '+str(cntUp))
    print (TAG+'para baixo: '+str(cntDown))
    break

#####
#     CONTOURS     #
#####

_, contours0, hierarchy =
    cv2.findContours(mask2,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
for cnt in contours0:
    area = cv2.contourArea(cnt)
    if area > areaTH and area < areaTHSuperior:

        #####
        #     TRACKING     #
        #####

        M = cv2.moments(cnt)
        cx = int(M['m10']/M['m00'])
        cy = int(M['m01']/M['m00'])
        x,y,w,h = cv2.boundingRect(cnt)

        new = True
        if cy in range(upLimit,downLimit):
            for i in persons:
                (i, persons, cntUp, cntDown, stopLoop, new) = defineDirection(i,
                                     cx, cy, w, h, new, cntUp, cntDown, lineUp, lineDown, upLimit,
                                     downLimit, persons, TAG)
            if i.timeOut():
                # If it reaches timeout, remove person from list
                index = persons.index(i)
                persons.pop(index)
                del i

```

```

        if (stopLoop):
            break

    if new == True:
        p = personlib.MyPerson(pid,cx,cy, maxPAge)
        persons.append(p)
        pid += 1

    # Drawing persons
    drawPersons(frame, cx, cy, x, y, w, h, saveResults, path, videoName,
                cont, cnt)

timestamp = time.time()
dateTime =
    datetime.datetime.fromtimestamp(timestamp).strftime('%Y-%m-%d_%H-%M-%S')

dateTimeHour = dateTime.split('_')[1]
hour = dateTimeHour.split('-')[0]

if(cntUp - cntDown + settings.initialNumPeople < 0):
    cntUp = settings.cntUp;
    cntDown = settings.cntDown;

nightHours = ['23', '00', '01', '02', '03', '04', '05', '06']
if(hour in nightHours):
    if(settings.initialNumPeople != cntDown - cntUp):
        cntDown = cntUp + settings.initialNumPeople

# Drawing tracking
drawTrack(frame, persons, cntUp, cntDown, lineDownColor, lineUpColor, ptsL1,
          ptsL2, ptsL3, ptsL4, saveResults, path, videoName, cont)

cont+=1

#If ESC is pressed, stop
k = cv2.waitKey(30) & 0xff
if k == 27:
    camera.close()
    break

rawCapture.truncate()
rawCapture.seek(0)

#####
#   CLEAR   #
#####

```

```

## cap.release()

cv2.destroyAllWindows()

# return cntUp - cntDown

def calculatePoints(w, lineUp, lineDown, upLimit, downLimit):
    pt1 = [0, lineDown];
    pt2 = [w, lineDown];

    pt3 = [0, lineUp];
    pt4 = [w, lineUp];

    pt5 = [0, upLimit];
    pt6 = [w, upLimit];

    pt7 = [0, downLimit];
    pt8 = [w, downLimit];

    return (pt1, pt2, pt3, pt4, pt5, pt6, pt7, pt8)

def calculateLinePoints(pt1, pt2, pt3, pt4, pt5, pt6, pt7, pt8):
    ptsL1 = np.array([pt1,pt2], np.int32)
    ptsL1 = ptsL1.reshape((-1,1,2))

    ptsL2 = np.array([pt3,pt4], np.int32)
    ptsL2 = ptsL2.reshape((-1,1,2))

    ptsL3 = np.array([pt5,pt6], np.int32)
    ptsL3 = ptsL3.reshape((-1,1,2))

    ptsL4 = np.array([pt7,pt8], np.int32)
    ptsL4 = ptsL4.reshape((-1,1,2))

    return (ptsL1, ptsL2, ptsL3, ptsL4)

def drawTrack(frame, persons, cntUp, cntDown, lineDownColor, lineUpColor, ptsL1, ptsL2,
ptsL3, ptsL4, saveResults, path, videoName, cont):
    font = cv2.FONT_HERSHEY_SIMPLEX

    for i in persons:
        cv2.putText(frame,
            str(i.getId()),(i.getX(),i.getY()),font,0.3,i.getRGB(),1,cv2.LINE_AA)

#####
# IMAGES #
#####

```

```

strUp = 'UP: ' + str(cntUp)
strDown = 'DOWN: ' + str(cntDown)
frame = cv2.polylines(frame, [ptsL1], False, lineDownColor, thickness=2)
frame = cv2.polylines(frame, [ptsL2], False, lineUpColor, thickness=2)
frame = cv2.polylines(frame, [ptsL3], False, (255,255,255), thickness=1)
frame = cv2.polylines(frame, [ptsL4], False, (255,255,255), thickness=1)
cv2.putText(frame, strUp, (10,40), font, 0.5, (255,255,255), 2, cv2.LINE_AA)
cv2.putText(frame, strUp, (10,40), font, 0.5, (0,0,255), 1, cv2.LINE_AA)
cv2.putText(frame, strDown, (10,90), font, 0.5, (255,255,255), 2, cv2.LINE_AA)
cv2.putText(frame, strDown, (10,90), font, 0.5, (255,0,0), 1, cv2.LINE_AA)

cv2.imshow('Frame', frame)

if (saveResults):
    nameImgFinal = path + '/' + videoName + '_' + str(cont) + '_final.jpg'
    cv2.imwrite(nameImgFinal, frame)

def drawPersons(frame, cx, cy, x, y, w, h, saveResults, path, videoName, cont, cnt):
    cv2.circle(frame, (cx, cy), 5, (0,0,255), -1)
    img = cv2.rectangle(frame, (x, y), (x+w, y+h), (0,255,0), 2)

    if (saveResults):
        nameImgRectangle = path + '/' + videoName + '_' + str(cont) + '_rectangle.jpg'
        cv2.imwrite(nameImgRectangle, img)

    imgContours = cv2.drawContours(frame, cnt, -1, (0,255,0), 3)

    if (saveResults):
        nameImgCont = path + '/' + videoName + '_' + str(cont) + '_contours.jpg'
        cv2.imwrite(nameImgCont, imgContours)

def saveSubtractorImages(saveResults, path, videoName, cont, frame, fgmask):
    if (saveResults):
        nameImgOriginal = path + '/' + videoName + '_' + str(cont) + '_original.jpg'
        nameImgSub = path + '/' + videoName + '_' + str(cont) + '_subtractor.jpg'
        cv2.imwrite(nameImgOriginal, frame)
        cv2.imwrite(nameImgSub, fgmask)

def preProcess(fgmask, fgmask2, saveResults, path, videoName, cont, kernelOp, kernelCl):

    # Eliminate shadows
    ret, imBin = cv2.threshold(fgmask, 200, 255, cv2.THRESH_BINARY)
    ret, imBin2 = cv2.threshold(fgmask2, 200, 255, cv2.THRESH_BINARY)
    #ret, imBin = cv2.threshold(fgmask, 0, 255, cv2.THRESH_BINARY+cv.THRESH_OTSU)
    #ret, imBin2 = cv2.threshold(fgmask2, 0, 255, cv2.THRESH_BINARY+cv.THRESH_OTSU)

    # Eliminate noise

```

```

mask = cv2.morphologyEx(imBin, cv2.MORPH_OPEN, kernelOp)
mask2 = cv2.morphologyEx(imBin2, cv2.MORPH_OPEN, kernelOp)

# Join white parts
mask = cv2.morphologyEx(mask , cv2.MORPH_CLOSE, kernelCl)
mask2 = cv2.morphologyEx(mask2, cv2.MORPH_CLOSE, kernelCl)

if(saveResults):
    nameImgTransf = path + '/' + videoName + '_' + str(cont) + '_transformation.jpg'
    cv2.imwrite(nameImgTransf,mask)

return (mask, mask2)

def defineDirection(i, cx, cy, w, h, new, cntUp, cntDown, lineUp, lineDown, upLimit,
downLimit, persons, TAG):
    stopLoop = False
    if abs(cx-i.getX()) <= w and abs(cy-i.getY()) <= h:
        # Close to a person already detected
        lineLeft = int(8*(320/10))
        lineRight = int(3*(320/10))

        new = False
        i.updateCoords(cx,cy) #Refresh
        if i.going_UP(lineDown,lineUp,lineLeft,lineRight) == True:
            cntUp += 1;
            print(TAG+str(i.getId()) + ' foi para cima aos '+time.strftime("%c"))
        elif i.going_DOWN(lineDown,lineUp,lineLeft,lineRight) == True:
            cntDown += 1;
            print(TAG+str(i.getId()) + ' foi para baixo aos '+time.strftime("%c"))
        stopLoop = True
    if i.getState() == '1':
        if i.getDir() == 'down' and i.getY() > downLimit:
            i.setDone()
        elif i.getDir() == 'up' and i.getY() < upLimit:
            i.setDone()

    return (i, persons, cntUp, cntDown, stopLoop, new)

```

---

Código I.2: Biblioteca utilizada para contagem do número de pessoas.

## I.2.2 Classe Auxiliar para Representar a Pessoa

---

```

##Identificacao e Contagem do Numero de Pessoas por Video
##Classe para Representar uma Pessoa
##Trabalho de Graduacao 1
##Autores:

```

```

## Alexandre Saran
## Mariana Pimentel
##Adaptado do Material de: Federico Mejia

from random import randint

class MyPerson:
    tracks = []

    def __init__(self, i, xi, yi, max_age):
        self.i = i
        self.x = xi
        self.y = yi
        self.tracks = []
        self.R = randint(0, 255)
        self.G = randint(0, 255)
        self.B = randint(0, 255)
        self.done = False
        self.state = '0'
        self.age = 0
        self.max_age = max_age
        self.dir = None

    def getRGB(self):
        return (self.R, self.G, self.B)

    def getTracks(self):
        return self.tracks

    def getId(self):
        return self.i

    def getState(self):
        return self.state

    def getDir(self):
        return self.dir

    def getX(self):
        return self.x

    def getY(self):
        return self.y

    def updateCoords(self, xn, yn):
        self.age = 0
        self.tracks.append([self.x, self.y])

```

```

self.x = xn
self.y = yn

def setDone(self):
    self.done = True

def timedOut(self):
    return self.done

def going_UP(self, mid_start, mid_end):
    if len(self.tracks) >= 2:
        if self.state == '0':
            if self.tracks[-1][1] < mid_end and self.tracks[-2][1] >= mid_end:
                state = '1'
                self.dir = 'up'
                return True
            else:
                return False
        else:
            return False

def going_DOWN(self, mid_start, mid_end):
    if len(self.tracks) >= 2:
        if self.state == '0':
            if self.tracks[-1][1] > mid_start and self.tracks[-2][1] <= mid_start:
                state = '1'
                self.dir = 'down'
                return True
            else:
                return False
        else:
            return False

def age_one(self):
    self.age += 1
    if self.age > self.max_age:
        self.done = True
    return True

class MultiPerson:
    def __init__(self, persons, xi, yi):
        self.persons = persons
        self.x = xi
        self.y = yi
        self.tracks = []
        self.R = randint(0, 255)

```

```
self.G = randint(0, 255)
self.B = randint(0, 255)
self.done = False
```

---

Código I.3: Classe para representar as pessoas na contagem.

## I.3 Banco de Dados

---

```
##Controle Antecipativo por Estimativa de Carga Termica em Video
##Biblioteca para Comunicacao com a Base de Dados
##Trabalho de Graduacao
##Autores:
## Alexandre Saran
## Mariana Pimentel

# coding=utf-8

import sqlite3
import serial.tools.list_ports
import time
import settings

# Inicialize database
def InicializeDatabase(name):
    database = sqlite3.connect('../..//Dados/'+name)
    return database

# Create table "informacaoSala"
def CreateTable(database):
    cursor = database.cursor()
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS informacaoSala(dataHora TEXT PRIMARY KEY, tempSala
            REAL,
            humSala REAL, tempVizinha REAL, tempExterna REAL,
            sinalPorta INTEGER, numPessoas INTEGER, sinalCompressor INTEGER,
            estadoLigado INTEGER, cicloDeTrabalho INTEGER)
    ''')
    database.commit()
    return database

# Insert data in table
def InsertData(database, dateTime, tempMeetingRoom, humMeetingRoom, tempLara,
    tempExternal, doorSignal, numPeople, compressorSignal, isOn, dutyCycle):
    cursor = database.cursor()
    TAG = '(sqlite) '
```

```

try:
    cursor.execute('''INSERT INTO
        informacaoSala(dataHora,tempSala,humSala,tempVizinha,tempExterna,sinalPorta,numPessoas,sinalCompressor,estadoLigado,cicloDeTrabalho)
        VALUES(?,?,?,?,?,?,?,?,?)''', (dateTime, tempMeetingRoom,
            humMeetingRoom, tempLara, tempExternal, doorSignal, numPeople,
            compressorSignal, isOn, dutyCycle))
    print(TAG+'dataHora:{0} | numPessoas:{1}'.format(dateTime, numPeople))
except Exception as e:
    print(TAG+'Falha ao inserir dados')
database.commit()
return database

# Read all data from the table
def ReadTable(database):
    TAG = '(sqlite) '
    cursor = database.cursor()
    try:
        cursor.execute('''SELECT
            dataHora,tempSala,humSala,tempVizinha,tempExterna,sinalPorta,numPessoas,sinalCompressor,
            estadoLigado, cicloDeTrabalho FROM informacaoSala''')
    except Exception as e:
        print(TAG+'Falha ao ler os dados')
    # Prints each line information
    data = cursor.fetchall()
    cont = 0
    for line in data:
        print(TAG+'Coluna{0} -> dataHora:{1} | tempSala:{2} | humSala:{3} |
            tempVizinha:{4} | tempExterna:{5} | sinalPorta:{6} | numPessoas:{7} |
            sinalCompressor:{8} | estadoLigado:{9} | cicloDeTrabalho:{10}'.format(cont,
            line[0], line[1], line[2], line[3], line[4], line[5], line[6], line[7],
            line[8], line[9]))
        cont = cont + 1
    return data

# Read all data from the table
def ReadTableLd(database):
    TAG = '(sqlite) '
    cursor = database.cursor()
    try:
        cursor.execute('''SELECT
            dataHora,tempSala,humSala,tempVizinha,tempExterna,sinalPorta,numPessoas,sinalCompressor,
            estadoLigado FROM informacaoSala''')
    except Exception as e:
        print(TAG+'Falha ao ler os dados')
    # Prints each line information
    data = cursor.fetchall()
    cont = 0

```

```

for line in data:
    print(TAG+'Coluna[{0}] -> dataHora:{1} | tempSala:{2} | humSala:{3} |
          tempVizinha:{4} | tempExterna:{5} | sinalPorta:{6} | numPessoas:{7} |
          sinalCompressor:{8} | estadoLigado:{9}'.format(cont, line[0], line[1],
          line[2], line[3], line[4], line[5], line[6], line[7], line[8]))
    cont = cont + 1
return data

# Read all data from the table
def ReadTableIdent(database):
    TAG = '(sqlite) '
    cursor = database.cursor()
    try:
        cursor.execute('''SELECT
            dataHora,tempSala,humSala,tempVizinha,tempExterna,sinalCompressor FROM
            informacaoSala''')
    except Exception as e:
        print(TAG+'Falha ao ler os dados')
    # Prints each line information
    data = cursor.fetchall()
    cont = 0
    for line in data:
        print(TAG+'Coluna[{0}] -> dataHora:{1} | tempSala:{2} | humSala:{3} |
              tempVizinha:{4} | tempExterna:{5} | sinalCompressor:{6}'.format(cont,
              line[0], line[1], line[2], line[3], line[4], line[5]))
        cont = cont + 1
    return data

# Close the table
def CloseTable(database):
    database.close()

def readTempHumDoor(tempMeetingRoom, humMeetingRoom, tempLara, tempExternal,
doorSignal):
    TAG = '(sqlite) '
    ports = list(serial.tools.list_ports.comports())
    for p in ports:
        if "Arduino" in p[1]:
            print(TAG+"Arduino Port: "+str(p[0]))
            ser=serial.Serial(p[0], 9600, timeout=1)
            break
    try:
        ser.flush()
        read = ser.readline()
        print(TAG + "read: " + str(read))

```

```

except:
    print(TAG + "Sem dados")
    readOk = False
    return (readOk, tempMeetingRoom, humMeetingRoom, tempLara, tempExternal,
            doorSignal)

if(read == "" or len(read) < 4):
    readOk = False
    return (readOk, tempMeetingRoom, humMeetingRoom, tempLara, tempExternal,
            doorSignal)
time.sleep(1)
read = read.decode("utf-8")
#print(read)
read = read[: -2]
#print(read)
data = read.split("|")
print(TAG + "Data: " + str(data))

for values in data:
    print(TAG + "Values: " + str(values))
    try:
        key,value = values.split(":")
    except:
        key,value = "Not found", "Not found"

    #print(key)
    #print(value)

    if(key == 'TM' and value != "Not found"):
        # print("Temp Meeting Room")
        tempMeetingRoom = value[:]
    elif(key == 'HM' and value != "Not found"):
        # print("Hum Meeting Room")
        humMeetingRoom = value
    elif(key == 'TL' and value != "Not found"):
        # print("Lara")
        tempLara = value
    elif(key == 'TE' and value != "Not found"):
        # print("External")
        tempExternal = value
    elif(key == 'DS' and value != "Not found"):
        # print("External")
        doorSignal = value

print(TAG + "Temperatura Sala de Reuniao: " + str(tempMeetingRoom))
print(TAG + "Humidade Sala de Reuniao: " + str(humMeetingRoom))
print(TAG + "Temperatura Lara: " + str(tempLara))

```

```

print(TAG + "Temperatura Externa: " + str(tempExternal))
print(TAG + "Sinal da Porta: " + str(doorSignal))

settings.tempMeetingRoom = tempMeetingRoom
settings.tempLara = tempLara
settings.tempExternal = tempExternal

readOk = True

return (readOk, tempMeetingRoom, humMeetingRoom, tempLara, tempExternal, doorSignal)

```

---

Código I.4: Biblioteca para leitura e escrita no banco de dados.

## I.4 Controle

---

```

##Controle Antecipativo por Estimativa de Carga Termica em Video
##Biblioteca para o Controle do Ar-Condicionado
##Trabalho de Graduacao
##Autores:
## Alexandre Saran
## Mariana Pimentel

import serial
import time
import struct
import serial.tools.list_ports
import settings
from scipy import signal
from simple_pid import PID

def Controller(lastOutput, cont):
    TAG = '(controller) '

    if (settings.controllerType == 0):
        # Identificacao do Modelo

        print(TAG + "Identificacao do Modelo")

        PERIOD = 60 # 4 horas

        print(TAG + "Contador: " + str(cont))

        if (cont >= 3 * 60):
            settings.isOn = 0
            output = 0
            if (cont == 4 * 60):

```

```

        cont = 1
    else:
        cont = cont + 1
else:
    settings.isOn = 1
    output = 100
    cont = cont + 1

elif(settings.controllerType == 1):
    # Liga-Desliga

    print(TAG + "Controlador: Liga-Desliga")

    PERIOD = 15 # 10 segundos

    MIN_TEMP = 22.7
    MAX_TEMP = 23.3

    print(TAG + "Contador: " + str(cont))
    print(TAG + "Temperatuda da Sala: "+ str(settings.tempMeetingRoom))

    allTimeOn = True
    if ((not allTimeOn) and cont >= 3 * 60 * 4):
        settings.isOn = 0
        output = 0
        if (cont == 4 * 60 * 4):
            cont = 1
        else:
            cont = cont + 1
    else:
        settings.isOn = 1
        if(float(settings.tempMeetingRoom) > 23.30):
            print(TAG + "Liga-Desliga: LIGA")
            output = 100
        elif(float(settings.tempMeetingRoom) < 22.70):
            print(TAG + "Liga-Desliga: DESLIGA")
            output = 0
        else:
            print(TAG + "Liga-Desliga: MANTEM")
            output = lastOutput
        cont = cont + 1

elif(settings.controllerType == 2):
    # PI
    print(TAG + "Controlador: PI")

```

```

PERIOD = 90

TEMP = 23.0

Kp = 0.12
Ki = 13 * Kp

allTimeOn = True
if((not allTimeOn) and cont > 3 * 15):
    settings.isOn = 0
    output = 0
    if (cont == 4 * 15):
        cont = 1
    else:
        cont = cont + 1
else:
    settings.isOn = 1

    cont = cont + 1

# PI
piController = PID (-Kp, -Ki, 0, setpoint = TEMP)

# Erro: diferenca entre temperatura desejada e medida
error = float(settings.tempMeetingRoom) - TEMP

if((error < 0.3 and lastOutput != 100) or error <= -0.2):
    # Sinal de controle
    controllerSignal = piController(float(settings.tempMeetingRoom))
    print(TAG + "Controller Signal: " + str(controllerSignal))

    # Saturacao
    if (controllerSignal>0.1):
        controllerSignal = 0.1;
    elif (controllerSignal < 0):
        controllerSignal = 0;

    output = controllerSignal * 100.0 / 0.1;

    print(TAG + "Controller Signal: " + str(output))
else:
    PERIOD = 15
    output = 100.0
    print(TAG + "***** ATENCAO *****")
    print(TAG + "****CORRIGINDO O PI****")
    print(TAG + "Controller Signal: " + str(output))

```

```

elif(settings.controllerType == 3):
    #Antecipativo
    print(TAG + "Controlador: Antecipativo")

    PERIOD = 90

    TEMP = 23.0
    TEMP_EMPTY_ROOM = 30.0

    Kp = 0.12
    Ki = 13 * Kp

    SAT = 0.1

    allTimeOn = False
    if((not allTimeOn) and cont > 120):
        settings.isOn = 0
        output = 0
        if (cont == 160):
            cont = 1
        else:
            cont = cont + 1
    else:
        settings.isOn = 1

    cont = cont + 1

    # Sinal de controle
    nPeople = settings.cntUp-settings.cntDown + settings.inicialNumPeople
    print(TAG + "People: " + str(nPeople))
    dontTurnOff = False
    if(dontTurnOff or nPeople != 0):
        print(TAG + "TEMP: " + str(TEMP))

    # PI
    piController = PID (-Kp, -Ki, 0, setpoint = TEMP)

    # Erro: diferenca entre temperatura desejada e medida
    error = float(settings.tempMeetingRoom) - TEMP

    if((error < 0.3 and lastOutput != 100) or error <= -0.2):
        # Sinal de controle
        controllerSignal = piController(float(settings.tempMeetingRoom))
        print(TAG + "Controller Signal: " + str(controllerSignal))

        controllerSignal = controllerSignal + nPeople*SAT*2/30

```

```

print(TAG + "Controller Signal Feedforward: " + str(controllerSignal))

# Saturacao
if (controllerSignal>0.1):
    controllerSignal = 0.1;
elif (controllerSignal < 0):
    controllerSignal = 0;

output = controllerSignal * 100.0 / 0.1;

print(TAG + "Controller Signal: " + str(output))

else:
    PERIOD = 15
    controllerSignal = SAT
    output = 100.0
    print(TAG + "***** ATENCAO *****")
    print(TAG + "****CORRIGINDO O PI****")
    print(TAG + "Controller Signal: " + str(output))

else:
    print(TAG + "TEMP: " + str(TEMP_EMPTY_ROOM))

# PI
piController = PID (-Kp, -Ki, 0, setpoint = TEMP_EMPTY_ROOM)

# Erro: diferenca entre temperatura desejada e medida
error = float(settings.tempMeetingRoom) - TEMP_EMPTY_ROOM

controllerSignal = piController(float(settings.tempMeetingRoom))
print(TAG + "Controller Signal Original: " + str(controllerSignal))

# Saturacao
if (controllerSignal>SAT):
    controllerSignal = SAT;
elif (controllerSignal < 0):
    controllerSignal = 0;

output = controllerSignal * 100.0 / SAT;

print(TAG + "Controller Signal: " + str(output))

(onTime, period) = PWM(output, PERIOD, TAG)

writeRele(onTime, period, TAG)

```

```

return (output, cont)

def PWM(output, PERIOD, TAG):
    OUTPUT_MAX = 100

    period = float(PERIOD)

    percentage = float(output)/float(OUTPUT_MAX)

    if percentage > 0.7:
        percentage = 1
    elif percentage < 0.15:
        percentage = 0
    elif percentage < 0.3:
        percentage = 0.3

    settings.dutyCycle = percentage

    onTime = period * percentage

    print(TAG+'Period: '+str(period))
    print(TAG+'Tempo Ligado: '+str(onTime))

    return (onTime, period)

def writeRele(onTime, period, TAG):

    ports = list(serial.tools.list_ports.comports())

    for p in ports:
        if "Arduino" in p[1]:
            print(TAG+"Arduino Port: "+str(p[0]))
            ser=serial.Serial(p[0], 9600, timeout=1)

    if (onTime>0):
        settings.compressorSignal = 1
        data = b"1"
        print(TAG + 'Sinal para o compressor ligar por '+str(onTime)+' segundos')
        ser.write(data)
        time.sleep(1)
        ser.flush()
        time.sleep(onTime)

    if (onTime != period):
        settings.compressorSignal = 0

```

```

data = b"0"
print(TAG + 'Sinal para o compressor desligar por '+str(period - onTime)+'
      segundos')
ser.write(data)
time.sleep(1)
ser.flush()
time.sleep(period - onTime)

```

---

Código I.5: Biblioteca para o controle.

## I.5 Câmera

---

```

##Controle Antecipativo por Estimativa de Carga Termica em Video
##Biblioteca para Comunicacao com a Camera
##Trabalho de Graduacao
##Autores:
## Alexandre Saran
## Mariana Pimentel

# coding=utf-8

from picamera import PiCamera
from time import sleep
from picamera.array import PiRGBArray

# Inicialize camera object
def InicializeCamera():
    camera = PiCamera()
    camera.resolution = (320, 240) # 320 240
    camera.framerate = 30
    rawCapture = PiRGBArray(camera, size=(320, 240))
    return (camera,rawCapture)

# Start recording and setup the file name
def StartVideo(camera, name):
    camera.start_preview()
    camera.start_recording('../ ../ ../Videos/'+name+'.h264')
    return camera

# Stop recording
def StopVideo(camera):
    camera.stop_recording()
    camera.stop_preview()
    return camera

```

---

Código I.6: Biblioteca para utilização da câmera.

## I.6 Configurações

---

```
##Controle Antecipativo por Estimativa de Carga Termica em Video
##Dados Globais
##Trabalho de Graduacao
##Autores:
##   Alexandre Saran
##   Mariana Pimentel

def init():
    global cntUp
    global cntDown
    global camera
    global controllerType
    global tempMeetingRoom
    global tempLara
    global tempExternal
    global dutyCycle
    global isOn
    global inicialNumPeople

    cntUp = 0
    cntDown = 0
    tempMeetingRoom = 0
    tempLara = 0
    tempExternal = 0
    compressorSignal = 0
    dutyCycle = 0
    isOn = 0
    inicialNumPeople = 2

    # Tipos de Controladores
    # 0 -> Identificacao do Modelo
    # 1 -> Liga-Desliga
    # 2 -> PI
    # 3 -> Antecipativo
    controllerType = 3
```

---

Código I.7: Dados e variáveis globais.

## I.7 Código do Arduino

---

```
//Controle Antecipativo por Estimativa de Carga Termica em Video
//Codigo Arduino
//Trabalho de Graduacao
```

```

//Autores:
// Alexandre Saran
// Mariana Pimentel

#include <Sensirion.h>

#include <DHT_U.h>
#include <DHT.h>

#include <Adafruit_Sensor.h>

#define DHTTYPE DHT22 // Sensor DHT 22 (AM2302)

// SHT71
const uint8_t pinMeetingRoom = 3; // SHT serial data
const uint8_t pinClkMeetingRoom = 2; // SHT serial clock
const uint32_t TRHSTEP = 3000UL; // Sensor query period
const uint32_t BLINKSTEP = 250UL; // LED blink period

Sensirion sht = Sensirion(pinMeetingRoom, pinClkMeetingRoom);

uint16_t rawData;
float tempMeetingRoom;
float humMeetingRoom;

byte shtState = 0;

// LM35
int pinLara = 40, pinExternal = 32; // Pino analogico para ligacao do LM35

// Definicoes do sensor : pino, tipo
DHT dhtLara(pinLara, DHTTYPE);
DHT dhtExternal(pinExternal, DHTTYPE);

float tempLara = 0, tempExternal = 0; // Variaveis que armazenam a temperatura em
Celsius

// Rele
float f;
char compSignal;
char lastCompSignal;
int rele = A0;
const uint32_t TRHCOM = 100UL;

// Time
unsigned long curMillis; // Time interval tracking
unsigned long trhMillis = 0;

```

```

unsigned long curMillisCom;           // Time interval tracking
unsigned long trhMillisCom = 0;

// Magnetic Sensor
boolean doorSignal;
int pinReedSwitch = 22;

void setup()
{
  Serial.begin(9600); // Wait at least 11 ms before first cmd

  //VCCs e GNDs
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(24, OUTPUT);
  pinMode(26, OUTPUT);
  pinMode(30, OUTPUT);
  pinMode(34, OUTPUT);
  pinMode(38, OUTPUT);
  pinMode(42, OUTPUT);
  digitalWrite(4, HIGH);
  digitalWrite(5, LOW);
  digitalWrite(24, HIGH);
  digitalWrite(26, LOW);
  digitalWrite(30, HIGH);
  digitalWrite(34, LOW);
  digitalWrite(38, HIGH);
  digitalWrite(42, LOW);

  sht.measTemp(&rawData);           // Maps to: sht.meas(TEMP, &rawData, BLOCK)
  tempMeetingRoom = sht.calcTemp(rawData);
  sht.measHumi(&rawData);           // Maps to: sht.meas(HUMI, &rawData, BLOCK)
  humMeetingRoom = sht.calcHumi(rawData, tempMeetingRoom);
  pinMode(rele, OUTPUT);
  dhtLara.begin();
  pinMode(pinReedSwitch, INPUT_PULLUP);
  logData();
}

void loop()
{
  curMillis = millis();
  curMillisCom = millis();

  if (curMillisCom - trhMillisCom >= TRHCOM) {
    compSignal = recvInfo();
    if(lastCompSignal != compSignal) {

```

```

    if(compSignal == '1') {
        digitalWrite(rele, LOW);
    }
    else if(compSignal == '0') {
        digitalWrite(rele,HIGH);
    }
    else if (compSignal == '2') {
        compSignal = lastCompSignal;
    }
}
lastCompSignal = compSignal;
trhMillisCom = curMillisCom;
}

switch (shtState) {
case 0:
    if (curMillis - trhMillis >= TRHSTEP) { // Start new temp/humi measurement?
        sht.meas(TEMP, &rawData, NONBLOCK);

        // Leitura da umidade
        // float humLara = dhtLara.readHumidity();
        // Leitura da temperatura (Celsius)
        tempLara = dhtLara.readTemperature();
        tempExternal = dhtExternal.readTemperature();
        doorSignal = digitalRead(pinReedSwitch);

        shtState++;
        trhMillis = curMillis;
    }
    break;
case 1:
    if (sht.measRdy()) { // Process temperature measurement?
        tempMeetingRoom = sht.calcTemp(rawData);
        sht.meas(HUMI, &rawData, NONBLOCK);
        shtState++;
    }
    break;
case 2:
    if (sht.measRdy()) { // Process humidity measurement?
        humMeetingRoom = sht.calcHumi(rawData, tempMeetingRoom);
        shtState = 0;
        logData();
    }
    break;
default:
    //Serial.println("How did I get here?");
    break;
}

```

```

    }
}

void logData() {

    Serial.print("TM:"); Serial.print(tempMeetingRoom);
    Serial.print("|HM:"); Serial.print(humMeetingRoom);
    Serial.print("|TL:"); Serial.print(tempLara,2);
    Serial.print("|TE:"); Serial.print(tempExternal,2);
    Serial.print("|DS:"); Serial.print(doorSignal);
    Serial.println();

    delay(100);
    tempLara = 0;
    tempExternal = 0;
}

char recvInfo() {
    char received;
    if (Serial.available() > 0) {
        received = Serial.read();
        // Serial.print("Signal Rele: ");
        // Serial.println(received);
    } else {
        received = '2';
    }
    return received;
}

```

---

Código I.8: Código Arduino para leitura e escrita dos dados.

## I.8 Simulação *Matlab* e *Simulink*

---

```

%Codigo para Carregar Dados para as Simulacoes
%Programa Principal
%Autores:
% Alexandre Saran
% Mariana Pimentel

%Carregando banco de dados
database = '2018-11-23_09-51-38.db'; %Substituir pelo desejado

conn = sqlite(database);

sqlquery = 'SELECT * from informacaoSala';

```

```

result = fetch(conn, sqlquery);

%Lendo informacoes
time = result(:,1);
tempMeetingRoom = result(:,2);
tempLara = result(:,4);
tempExternal = result(:,5);
numPeople = result(:,7);
compressorSignal = result(:,8);

%Mudando a formatacao para ser possivel trabalhar no Matlab
tempExternal = cell2mat(tempExternal);
tempExternal = double(tempExternal);

tempLara = cell2mat(tempLara);
tempLara = double(tempLara);

numPeople = cell2mat(numPeople);
numPeople = double(numPeople);

tempMeetingRoom = cell2mat(tempMeetingRoom);
tempMeetingRoom = double(tempMeetingRoom);

dateTime = cell2mat(time);
dateTime = datetime(dateTime, 'InputFormat', 'yyyy-MM-dd HH:mm:ss');

compressorSignal = cell2mat(compressorSignal);
compressorSignal = double(compressorSignal);

%Funcao de transferencia da planta identificada
K = 0.36783;
td = 5;
tau = 13.3140;

Gar = tf([K], [tau 1]);

%Ganhos do controlador PI projetado
Kp = 0.12;
Ki = 13 * Kp;

%Dados para a simulacao
%Ganhos
K = 0.01; %Ganho geral no modelo de carga termica
Kc = 2.5; %%Ganho da planta no modelo de carga termica
Ke = 1.4; %Influencia da carga termica externa
Kv = 3.3; %Influencia da carga termica das salas vizinhas
Knp = 0.1; %Influencia da carga termica de uma pessoa

```

```

Ktp = 36.5; %Temperatura media de uma pessoa
Kpwm = 20; %Ganho para normalizacao do PWM
Qp = (Ktp-23)*Knp; %Carga termica estimada

%Constantes de tempo das perturbacoes
ae = 5.4867; %Atraso da influencia da temperatura externa
av = 6.9028; %Atraso da influencia da temperatura das salas vizinhas
anp = 16.5; %Atraso da influencia do n de pessoas

%Montando estruturas para a simulacao
rangeData = size(tempMeetingRoom);

time = 1:rangeData;

tempExternalStruct.time = time;
tempExternalStruct.signals.values = tempExternal;
tempExternalStruct.signals.dimensions = 1;
temperaturaExterna = tempExternalStruct;

tempLaraStruct.time = time;
tempLaraStruct.signals.values = tempLara;
tempLaraStruct.signals.dimensions = 1;
temperaturaSalaVizinha = tempLaraStruct;

compressorSignalStruct.time = time;
compressorSignalStruct.signals.values = compressorSignal;
compressorSignalStruct.signals.dimensions = 1;
sinalCompressor = compressorSignalStruct;

tempMeetingRoomStruct.time = time;
tempMeetingRoomStruct.signals.values = tempMeetingRoom;
tempMeetingRoomStruct.signals.dimensions = 1;
temperaturaSalaReuniao = tempMeetingRoomStruct;

numPeopleStruct.time = time;
numPeopleStruct.signals.values = numPeople;
numPeopleStruct.signals.dimensions = 1;
numeroPessoas = numPeopleStruct;
numeroPessoasReal = numeroPessoas;
numeroPessoasDetectadas = numeroPessoas;

%Simulacoes (Descomentar o desejado)
%sim('simulink_ma') %Malha Aberta
%sim('simulink_ld') %Liga-Desliga
%sim('simulink_pi') %PI
%sim('simulink_antecipativo') %Antecipativo

```

```

%Plots

%Referencia
ref = zeros(rangeData,1);
for i = 1:rangeData
    ref(i) = 23;
end
ref = double(ref);

subplot(211);
plot(ref, 'r')
grid on;
hold on;
plot(ScopeData.time,ScopeData.signals.values, 'b');

%Descomentar o desejado
%title('Malha Aberta');
%title('Controlador Liga-Desliga');
%title('Controlador PI');
%title('Controlador Antecipativo');

%Media movel da temperatura da sala de reuniao
mean = movmean(ScopeData.signals.values, 50);

plot(mean, 'k')
plot(tempLara, 'm');
plot(tempExternal, 'g');
axis([0 rangeData 19 32]); %Ajustar para ficar visivel
xlabel('Tempo (min)');
ylabel('Temperatura (C)');
legend('Referencia', 'Temperatura da Sala de Reunio Simulada', 'Media Movel da
    Temperatura', 'Temperatura da Sala Vizinha', 'Temperatura Externa');

%No caso de Malha Aberta, comentar esta parte abaixo e a linha 117
subplot(212);
plot(numeroPessoas.time, numeroPessoas.signals.values);
grid on;
title('Contagem de Pessoas');
axis([0 rangeData 0 9]); %Ajustar para ficar visivel
xlabel('Tempo (min)');
ylabel('N de Pessoas');

```

---

Código I.9: Código para simulação dos controladores.

# ANEXOS

# I. DIAGRAMAS ELÉTRICOS

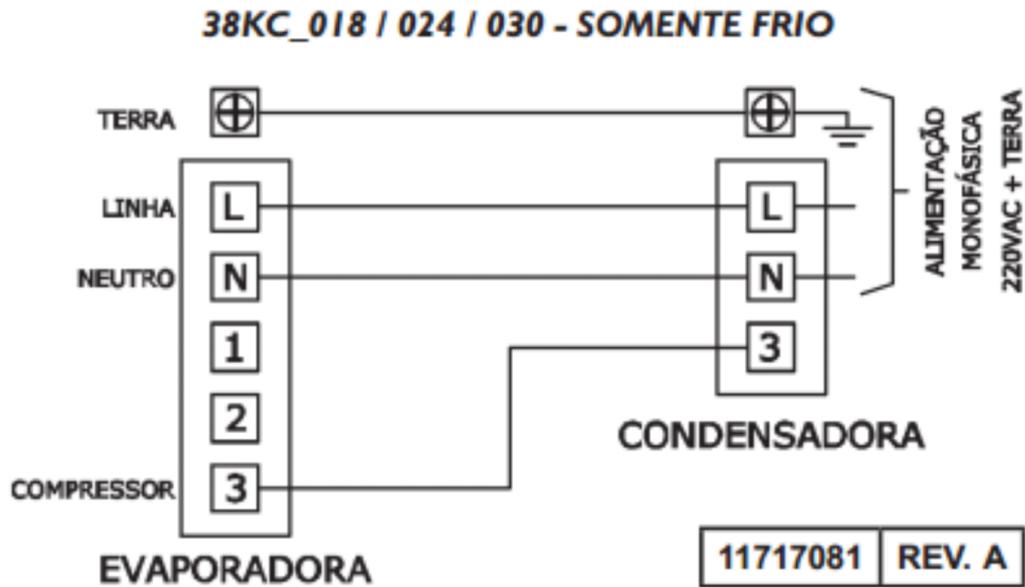


Figura I.1: Interligação elétrica entre a evaporadora e a condensadora [16].

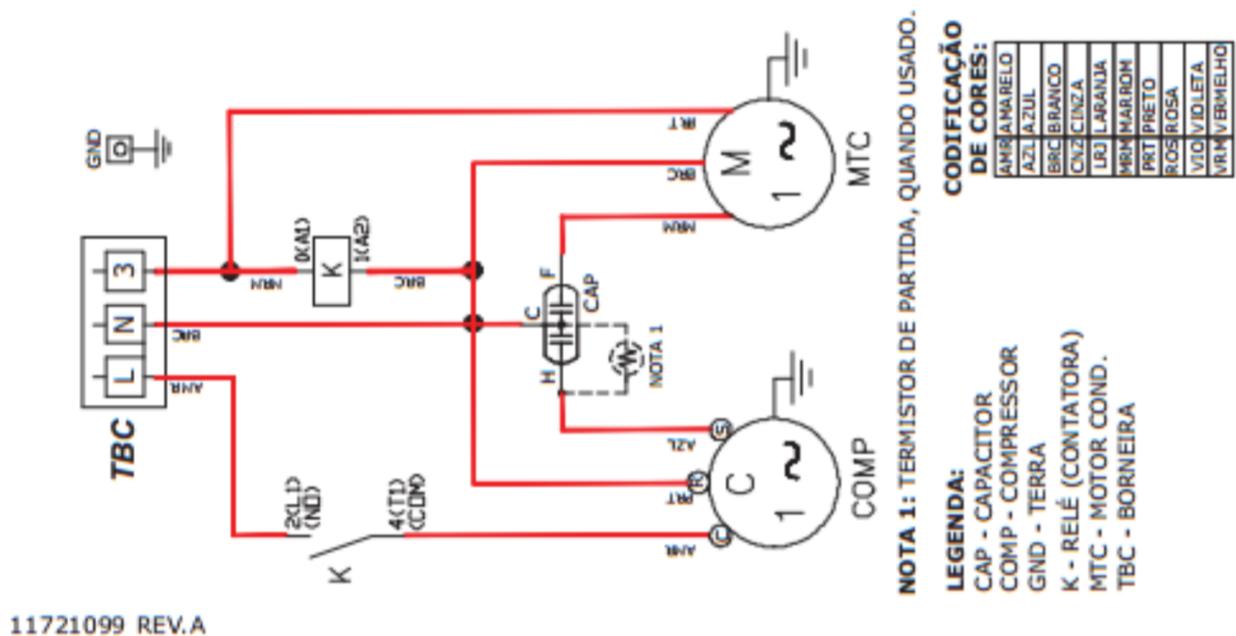


Figura I.2: Diagrama elétrico da condensadora [16].



## II. MANUAL DE INSTALAÇÃO E CONFIGURAÇÃO

### II.1 Sistema Operacional *Raspbian*

No presente trabalho foi instalado o sistema operacional *Raspbian* no *Raspberry Pi* através do *NOOBS*. Para isso utilizou-se como base o tutorial disponível em [49]. Os passos seguidos estão descritos abaixo.

Primeiramente, baixou-se a versão 2.8.1 do *NOOBS* no site [50]. Além disso, também foi necessário baixar o *software freeware* no site [51] responsável por formatar o cartão *SD* em um formato compatível com o *Raspberry Pi*.

Em seguida, o arquivo do *NOOBS* já baixado foi descompactado no cartão *micro-SD* de 32 GB destinado ao projeto. Com o *Raspberry Pi* desligado e nenhum dispositivo de entrada e saída conectado nele, colocou-se o *micro-SD* em sua entrada dedicada.

Então, ligou-se o *Raspberry Pi* e conectou-se os outros dispositivos de entrada e saída como o monitor, o teclado e o *mouse*. O *boot* do sistema ocorre de forma automática. Ao aparecer uma mensagem solicitando a instalação do *NOOBS*, deve-se confirmar. Assim que finalizar a instalação do SO, o *Raspberry Pi* irá reiniciar. Em seguida, basta finalizar as configurações básicas para uso.

Para definir a senha do usuário *textitroot*, deve-se digitar na linha de comando descrita no comando II.1.

---

```
$ sudo passwd
```

---

Comando II.1: Configurar a Senha

E então digitar a senha necessária.

Para finalizar o procedimento de configuração básica, pode-se atualizar o repositório para isso usa-se os comandos II.2.

---

```
$ sudo apt-get update  
$ sudo apt-get upgrade
```

---

Comando II.2: Atualizar o Repositório

### II.2 Câmera

Para utilizar a câmera, foi necessário configurar o *Raspberry Pi*. Para isso, foi utilizado o procedimento descrito em [17].

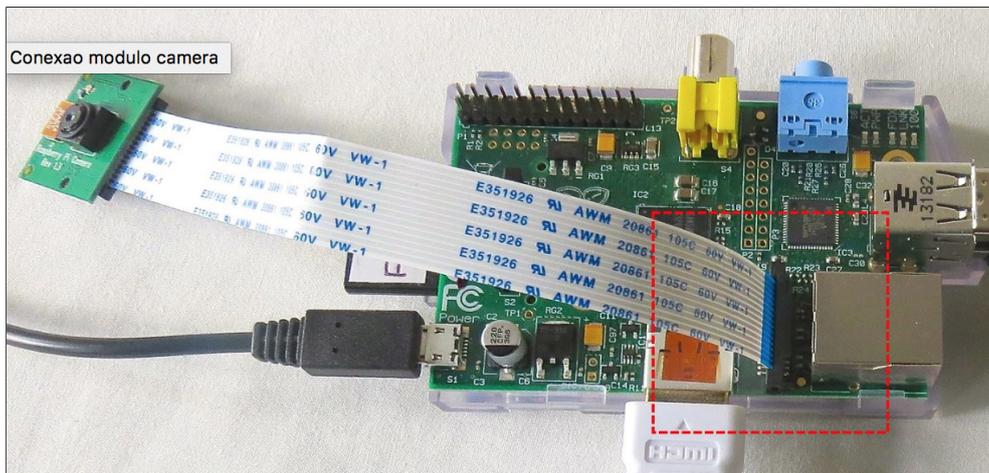


Figura II.1: Conexão do Módulo Câmera com o Raspberry Pi [17]

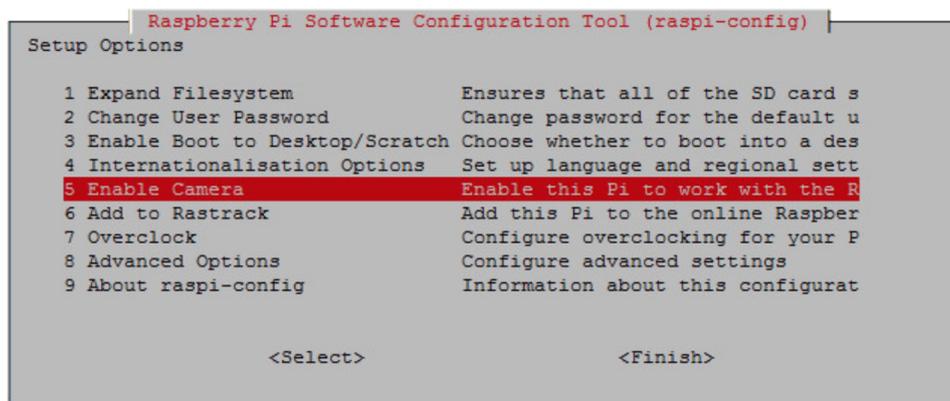


Figura II.2: Tela de Configurações do *Raspberry Pi* com opção *Enable Camera* Selecionada [17]

Primeiramente, deve-se conectar a câmera cuidadosamente no *Raspberry Pi*. Já existe um conector no *hardware* do *Raspberry Pi* exclusivo para a conexão com a câmera utilizada no projeto como pode ser visto na figura II.1. Para fazer a conexão, deve-se desligar o *Raspberry Pi* e inserir o cabo da câmera com o lado azul virado em direção ao conector *Ethernet*. Em seguida, deve-se apertar as laterais do conector e checar se o cabo esta devidamente preso.

Assim que a câmera e o *Raspberry Pi* já estejam conectados, deve-se abrir o terminal e entrar nas configurações utilizando o comando II.3.

---

```
$ sudo raspi-config
```

---

Comando II.3: Abrir as Configurações do *Raspberry Pi*

Ao executar esse comando, aparecerá a tela de configurações mostrada na figura II.2. Deve-se selecionar a opção *Enable Camera* que irá habilitar o uso da câmera. Em seguida, será necessário reiniciar o *Raspberry Pi* para que ele atualize as configurações feitas.

---

```
$ raspistill -o imagem.png
```

---

Comando II.4: Capturar uma Imagem

---

```
$ raspivid -o nome_do_video.h264
```

---

Comando II.5: Capturar uma Vídeo

Agora, já é possível utilizar a câmera para capturar imagens e vídeos. Se desejar uma foto, utilize o comando II.4. Já para obter um vídeo, utilize o comando II.5.

---

```
$ raspistill  
$ raspivid
```

---

Comando II.6: Mostrar Lista Completa de Comandos

Além disso, pode-se acessar um lista com todos os comandos possíveis de serem utilizados com as mais diversas funcionalidades, como mudar o tamanho, definir efeitos, entre outras. Para isso digite os comandos II.6.

## II.3 Banco de Dados SQLite

Para instalar e utilizar o banco de dados, os tutoriais utilizados como base foram [52] e [53]. Para instalar o *SQLite*, basta digitar o comando II.7. Já para criar uma base dados, pode-se utilizar o comando II.8.

---

```
$ sudo apt-get install sqlite3
```

---

Comando II.7: Instalar o Banco de Dados SQLite

---

```
$ sqlite3 basededados.db
```

---

Comando II.8: Criar uma Base de Dados

## II.4 OpenCV

O tutorial utilizado para fazer a instalação do *OpenCV* no *Raspberry Pi* está descrito em [54]. O primeiro passo é expandir o sistema de arquivos para incluir todo o espaço disponível no cartão *SD*. Para isso, utilize o comando II.9 para abrir as configurações do *Raspberry Pi* e selecione a opção *Expand Filesystem* como mostrado na figura II.4. Então, reinicie o *Raspberry Pi*.

---

```
$ sudo raspi-config
```

---

Comando II.9: Abrir a tela de Configurações do Raspberry Pi

Como o *OpenCV* em conjunto com as suas dependências necessita de bastante espaço em disco, um dica é deletar o programa *Wolfram Engine* do *Raspberry Pi*. Para fazer isso basta utilizar o comando II.10.

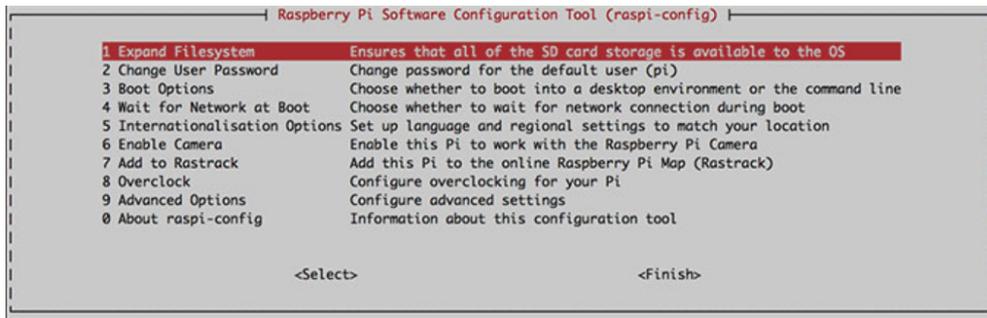


Figura II.3: Tela de Configurações do *Raspberry Pi* com opção *Expand Filesystem* Seleccionada [17]

---

```
$ sudo apt-get purge wolfram-engine
```

---

Comando II.10: Deletar o Programa Wolfram Engine do Raspberry Pi

Caso não tenha atualizado o repositório do *Raspberry* ainda, utilize o comando II.11. Em seguida, instale o *CMake* conforme o comando II.12. O *CMake* irá ajudar nas configurações do *OpenCV*.

---

```
$ sudo apt-get update
$ sudo apt-get upgrade
```

---

Comando II.11: Atualizar o Repositório

---

```
$ sudo apt-get install build-essential cmake pkg-config
```

---

Comando II.12: Instalar o CMake

Então, deve-se instalar algumas bibliotecas de entrada e saída tanto de imagens como de vídeos. Algumas outras bibliotecas também são importantes, como as bibliotecas utilizadas para mostrar imagens na tela ou criar uma interface gráfica com o usuário. Outras bibliotecas otimizam algumas operações internas do *OpenCV*. Para instalar todas essas bibliotecas citadas, basta utilizar os comandos II.13.

---

```
$ sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev
$ sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev
$ sudo apt-get install libxvidcore-dev libx264-dev
$ sudo apt-get install libgtk2.0-dev
$ sudo apt-get install libatlas-base-dev gfortran
```

---

Comando II.13: Baixar as Bibliotecas Necessárias

Por fim, deve-se instalar o *Python* versões 2.7 e 3 para conseguir compilar o *OpenCV* com *Python*. Para isso, utilize o comando II.14.

---

```
$ sudo apt-get install python2.7-dev python3-dev
```

---

Comando II.14: Instalar o Python

O próximo passo é baixar código fonte do *OpenCV* e instalá-lo. Neste trabalho, utilizamos a versão 3.1.0. Utilize os comandos descritos em II.15. Atente-se para as versões dos arquivos *opencv* e *opencv\_contrib*, ambos devem ser da mesma versão, nesse caso a versão 3.1.0.

---

```
$ cd ~
$ wget -O opencv.zip https://github.com/Itseez/opencv/archive/3.1.0.zip
$ unzip opencv.zip
$ wget -O opencv_contrib.zip
  https://github.com/Itseez/opencv_contrib/archive/3.1.0.zip
$ unzip opencv_contrib.zip
```

---

Comando II.15: Baixar e Instalar o OpenCV

Em seguida, instala-se o pacote de gerenciamento do *Python*, conhecido como *pip*. Além disso, é altamente recomendável a instalação dos pacotes de ambientes virtuais do *Python* como o *virtualenv* e o *virtualenvwrapper*. Para fazer a instalação, pode-se utilizar o comando II.16.

---

```
$ wget https://bootstrap.pypa.io/get-pip.py
$ sudo python get-pip.py
$ sudo pip install virtualenv virtualenvwrapper
$ sudo rm -rf ~/.cache/pip
```

---

Comando II.16: Instalar o *Pip*, o *Virtualenv* e o *Virtualenvwrapper*

Após a instalação do pacotes *virtualenv* e *virtualenvwrapper*, deve-se modificar o arquivo *~/.profile* de forma a incluir as linhas II.17 em seu final. Para isso, pode-se utilizar qualquer editor de texto no terminal, como o *Nano* ou o *Vim*. Para recarregar o arquivo, utilize o comando II.18.

---

```
# virtualenv and virtualenvwrapper
export WORKON_HOME=$HOME/.virtualenvs
source /usr/local/bin/virtualenvwrapper.sh
```

---

Texto II.17: Linhas a Serem Adicionadas no Final do Arquivo *~/.profile*

---

```
$ source ~/.profile
```

---

Comando II.18: Recarregar o arquivo *~/.profile*

Para criar o ambiente virtual do *Python* que será utilizado para o desenvolvimento dos códigos, utiliza-se o comando II.19. Para acessar esse mesmo ambiente virtual posteriormente, basta utilizar os comandos II.20.

---

```
$ mkvirtualenv cv -p python3
```

---

Comando II.19: Criar um Ambiente Virtual

---

```
$ source ~/.profile
```

---

```
$ workon cv
```

---

Comando II.20: Acessar o Ambiente Virtual

Dentro do ambiente virtual, deve-se então instalar o *Numpy* que é um pacote do *Python* utilizado para processamento numérico. O comando II.21 irá instalar o pacote.

---

```
$ pip install numpy
```

---

Comando II.21: Instalar o pacote Numpy

Agora, já pode-se compilar e instalar o *OpenCV* em si. Para configurar e construir o *OpenCV* utilizando o *CMake*, use os comando II.22. Então, para compilar o *OpenCV*, basta digitar o comando II.23. Finalmente, para instalar o *OpenCV*, use os comandos II.24.

Então, pode-se checar se o *OpenCV* e os pacotes necessários estão instalados utilizando o comando II.25 para verificar o conteúdo da pasta. Por fim, pode-se alterar o nome do arquivo do *OpenCV* por um nome mais fácil como *cv2.so* com os comandos II.26.

Para finalizar, pode-se fazer um último teste para garantir que o *OpenCV* foi corretamente instalado. Para fazer o teste, digite os comandos mostrados em II.27. Então remova os arquivos *opencv* e *opencv\_contrib* com o comando II.28.

---

```
$ cd ~/opencv-3.1.0/  
$ mkdir build  
$ cd build  
$ cmake -D CMAKE_BUILD_TYPE=RELEASE \  
  -D CMAKE_INSTALL_PREFIX=/usr/local \  
  -D INSTALL_PYTHON_EXAMPLES=ON \  
  -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-3.1.0/modules \  
  -D BUILD_EXAMPLES=ON ..
```

---

Comando II.22: Configurar e Construir o OpenCV

```
$ make
```

---

Comando II.23: Compilar o OpenCV

```
$ sudo make install  
$ sudo ldconfig
```

---

Comando II.24: Instalar o OpenCV

```
$ ls -l /usr/local/lib/python3.6/site-packages/
```

---

Comando II.25: Checar a Pasta Onde o OpenCV e os Pacotes Devem Estar Instalados

```
$ cd /usr/local/lib/python3.6/site-packages/  
$ sudo mv cv2.cpython-36m.so cv2.so
```

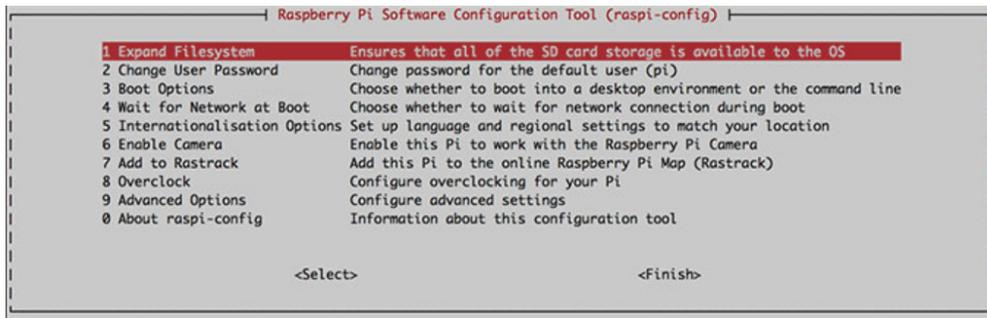


Figura II.4: Tela de Configurações do Raspberry Pi com opção *Expand Filesystem* Seleccionada [17]

```
$ cd ~/.virtualenvs/cv/lib/python3.6/site-packages/
$ ln -s /usr/local/lib/python3.6/site-packages/cv2.so cv2.so
```

Comando II.26: Checar a Pasta Onde o OpenCV e os Pacotes Devem Estar Instalados

```
$ source ~/.profile
$ workon cv
$ python
>>> import cv2
>>> cv2.__version__
'3.1.0'
```

Comando II.27: Testar se o OpenCV foi Instalado

```
$ rm -rf opencv-3.1.0 opencv_contrib-3.1.0
```

Comando II.28: Remover os Arquivos *Opencv* e *Opencv\_contrib*