



**UNIVERSIDADE FEDERAL DA FRONTEIRA SUL
CAMPUS CHAPECÓ
CURSO DE CIÊNCIA DA COMPUTAÇÃO**

ÁLESSON SCAPINELLO SELHORST

**UTILIZAÇÃO DE VISÃO COMPUTACIONAL E DETECÇÃO DE
CARACTERÍSTICAS PARA AUXILIAR NA NAVEGAÇÃO DE
PESSOAS COM DEFICIÊNCIA VISUAL EM AMBIENTES INTERNOS**

**CHAPECÓ
2014**

ÁLESSON SCAPINELLO SELHORST

**UTILIZAÇÃO DE VISÃO COMPUTACIONAL E DETECÇÃO DE
CARACTERÍSTICAS PARA AUXILIAR NA NAVEGAÇÃO DE
PESSOAS COM DEFICIÊNCIA VISUAL EM AMBIENTES INTERNOS**

Trabalho de conclusão de curso de graduação apresentado como requisito para obtenção do grau de Bacharel em Ciência da Computação da Universidade Federal da Fronteira Sul.

Orientador: Prof. Me. Fernando Bevilacqua

CHAPECÓ
2014

Scapinello Selhorst, Álesson

Utilização de Visão Computacional e Detecção de Características para auxiliar na navegação de pessoas com deficiência visual em ambientes internos / por Álesson Scapinello Selhorst. – 2014.

81 f.: il.

Orientador: Fernando Bevilacqua

Trabalho de conclusão de curso (graduação) - Universidade Federal da Fronteira Sul, Ciência da Computação, Curso de Ciência da Computação, Chapecó, SC, 2014.

- 1. Visão Computacional. 2. Processamento de Imagens.
- 3. Extração de Características. 4. Reconhecimento de Objetos.
- I. Bevilacqua, Fernando, orient. II. Universidade Federal da Fronteira Sul. III. Título.

© 2014

Todos os direitos autorais reservados a Álesson Scapinello Selhorst. A reprodução de partes ou do todo deste trabalho só poderá ser feita mediante a citação da fonte.

E-mail: alesson12382@gmail.com

ÁLESSON SCAPINELLO SELHORST

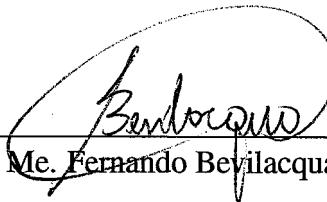
**UTILIZAÇÃO DE VISÃO COMPUTACIONAL E DETECÇÃO DE
CARACTERÍSTICAS PARA AUXILIAR NA NAVEGAÇÃO DE PESSOAS
COM DEFICIÊNCIA VISUAL EM AMBIENTES INTERNOS**

Trabalho de conclusão de curso de graduação apresentado como requisito para obtenção do grau de Bacharel em Ciência da Computação da Universidade Federal da Fronteira Sul.

Orientador: Prof. Me. Fernando Bevilacqua

Este trabalho de conclusão de curso foi defendido e aprovado pela banca em: 10/12/14

BANCA EXAMINADORA:



Me. Fernando Bevilacqua - UFFS



Dr. Rafael Piccin Torchelsen - UFFS



Me. Adriano Sanick Padilha - UFFS

AGRADECIMENTOS

Agradeço aos meus pais, que estavam presentes em todas as dificuldades encontradas me guiando para as decisões corretas, se doaram e renunciaram aos seus sonhos para que eu possa alcançar os meus.

Agradeço a todo o corpo docente da Universidade Federal da Fronteira Sul por transmitirem seus conhecimentos de tal modo que eu pudesse estar apto para concluir este trabalho. Aos professores Rafael Torchelsen e Adriano Padilha por participarem da banca e em especial ao professor Fernando Bevilacqua pela confiança, pelas correções, pelos incentivos e por ter dedicado seu tempo na orientação deste trabalho, exigindo de mim muito mais que eu supunha ser capaz de fazer.

Agradeço a todos os colegas e amigos que me acompanharam durante esta caminhada, que ajudaram sempre que necessário e que estavam sempre acreditando no sucesso do trabalho, em especial a sala 306. Meus sinceros agradecimentos também a todos aqueles que contribuíram diretamente ou indiretamente para a realização deste trabalho.

RESUMO

Os portadores de deficiência visual enfrentam inúmeros obstáculos em seu processo de inclusão na sociedade. A visão computacional pode ser usada para uma maior qualidade de vida aos portadores de deficiência visual, contribuindo com a acessibilidade dos locais onde eles frequentam, além de auxiliar na resolução de dificuldades encontradas em seu cotidiano. Este trabalho propõe a utilização de visão computacional e reconhecimento de imagens para auxiliar a navegação de pessoas com deficiência visual em ambientes fechados utilizando metadados previamente coletados. Criou-se um sistema-guia que possa auxiliar caso a pessoa esteja com dificuldades durante sua navegação e obteve-se um resultado satisfatório no reconhecimento destes locais. O trabalho possui como contribuição uma revisão bibliográfica acerca das técnicas de manipulação de imagens, extração de características e reconhecimento de imagens, além de uma aplicação capaz de reconhecer cenários/objetos e informá-los ao usuário através de um retorno auditivo. A aplicação foi validada através de testes.

Palavras-chave: Visão Computacional. Processamento de Imagens. Extração de Características. Reconhecimento de Objetos.

ABSTRACT

Visually impaired people face many obstacles in a daily basis. Computer vision can be used to improve their quality of life, contributing to enhance the accessibility of places. This work proposes the use of computer vision and image recognition to aid indoors navigation of visually impaired people. A prof-of-concept system was developed using a mobile application and a web server containing a set images and metadata describing the environment. The mobile application in possession of the user captures images and gives audio feedback based on the recognition of that image by the server. The image recognition and audio feedback are based on images and their metadata previously added to the server. The main contribution of this work is the implementation of a system able to help visually impaired people, along with a literature review about image manipulation, feature extraction and image recognition. A set of tests were performed to evaluate the system accuracy, aiming to simulate real use cases such as when an user taking a picture stands in front of or besides an object.

Keywords: Computer Vision. Image Processing. Feature Extraction. Object Recognition.

LISTA DE FIGURAS

Figura 1.1 – Estrutura e visão geral do projeto. 1- Captura da imagem, 2- Envio para o servidor, 3- Comparação de imagens usando visão computacional, 4- Acesso ao BD, 5- Retorno dos metadados ao servidor, 6- Retorno dos metadados ao usuário.....	17
Figura 2.1 – Estrutura de um sistema de visão computacional e suas etapas [25]	19
Figura 2.2 – Aplicação da técnica de filtro de média. À esquerda a imagem original, à direita resultado da aplicação do filtro de média [33]	20
Figura 2.3 – Aplicação de limiarização. À esquerda a imagem original, à direita o resultado da aplicação da limiarização sobre a imagem original [11]	22
Figura 2.4 – Aplicação da técnica de crescimento de região. a) Início da técnica com aplicação da semente. b) Início do crescimento da região. c) Nível intermediário da técnica. d) Região completa. [11].....	23
Figura 2.5 – Cálculo do gradiente de <i>Sobel</i> . Gx: Calcula máscara verticalmente - Gy: Calcula máscara horizontalmente [20].....	25
Figura 2.6 – Cálculo do gradiente de <i>Prewitt</i> . Gx: Calcula máscara verticalmente - Gy: Calcula máscara horizontalmente [20]	25
Figura 2.7 – a) Imagem original. b) Aplicação da técnica de <i>Sobel</i> sobre a imagem original. c) Aplicação da técnica de <i>Prewitt</i> sobre a imagem original [11] ...	26
Figura 2.8 – Processo de reconhecimento de padrões entre duas imagens através da técnica SIFT [22]	27
Figura 2.9 – Correlações encontradas em duas imagens utilizando o método SURF [28]..	28
Figura 3.1 – Estrutura e funcionamento do sistema <i>VizWiz</i> [4].....	29
Figura 3.2 – Protótipo do sistema <i>Drishti</i> [16].....	31
Figura 3.3 – Estrutura e componentes funcionais do sistema de navegação para cegos [21]	31
Figura 4.1 – Estrutura geral da aplicação. (1) Aplicação móvel. (2) Fluxo de dados no servidor	33
Figura 4.2 – Sequência de passos para uma foto capturada	34
Figura 4.3 – Sequência de passos quando chega uma resposta do servidor	35
Figura 4.4 – Sequência de passos para cada requisição.....	36
Figura 4.5 – Sequência de passos após a resposta do módulo de reconhecimento de imagens	36
Figura 4.6 – Sequência de passos que ocorrem na aplicação de reconhecimento de imagens	37
Figura 4.7 – Lista de imagens carregadas pela aplicação	38
Figura 4.8 – Exemplo de descritores em uma imagem.	38
Figura 4.9 – Exemplo de comparação entre duas imagens iguais. (A) Imagem original.(B) Aplicação das técnicas.....	39
Figura 4.10 – Exemplo de comparação entre duas imagens semelhantes	39
Figura 4.11 – Exemplo de comparação entre duas imagens diferentes, sem a criação de <i>matches</i>	40
Figura 4.12 – Envio dos metadados correspondente a imagem escolhida ao servidor.....	41
Figura 4.13 – Sequência de passos para carregar as imagens de treinamento. (1) Imagens de treinamento. (2) Imagens de treinamento com destaque para os pontos chaves	42
Figura 4.14 – Sequência de passos para carregar a imagem <i>query</i> . (1) Imagem <i>query</i> . (2) Imagem <i>query</i> com destaque para os pontos chaves	42

Figura 4.15 – Comparação das imagens de treinamento com a imagem <i>query</i>	43
Figura 5.1 – Imagens utilizadas para o teste de acurácia	45
Figura 5.2 – Caso de teste 1 com elevador. Imagem reconhecida corretamente com 100% de <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.....	46
Figura 5.3 – Caso de teste 2 com elevador. Imagem reconhecida corretamente com 38,65% dos <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.	47
Figura 5.4 – Caso de teste 3 com elevador. Imagem reconhecida corretamente com 26,97% dos <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.	47
Figura 5.5 – Caso de teste 4 com elevador. Imagem reconhecida corretamente com 15,67% dos <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados	48
Figura 5.6 – Caso de teste 5 com elevador. Nenhuma imagem reconhecida. À esquerda está a imagem da aplicação e à direita a imagem que deveria ser selecionada no banco de dados, com 8,74% de <i>matches</i>	48
Figura 5.7 – Caso de teste 6 com elevador. Nenhuma imagem reconhecida. À esquerda está a imagem da aplicação e à direita a imagem que deveria ser selecionada no banco de dados, com 7,54% de <i>matches</i>	49
Figura 5.8 – Caso de teste 7 com elevador. Imagem reconhecida corretamente com 41,92% dos <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.	50
Figura 5.9 – Caso de teste 8 com elevador. Imagem reconhecida corretamente com 31,28% dos <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.	50
Figura 5.10 – Caso de teste 9 com elevador. Imagem reconhecida corretamente com 37,83% dos <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.	51
Figura 5.11 – Porcentagem de reconhecimento dos testes realizados, utilizando o elevador como alvo.....	51
Figura 5.12 – Resultados obtidos nos testes realizados com o elevador	52
Figura 5.13 – Caso de teste 1 com extintor. Imagem reconhecida corretamente com 100% de <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.	53
Figura 5.14 – Caso de teste 2 com extintor. Imagem reconhecida corretamente com 26,61% de <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.	53
Figura 5.15 – Caso de teste 3 com extintor. Imagem reconhecida corretamente com 17,59% de <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.	54
Figura 5.16 – Caso de teste 4 com extintor. Imagem reconhecida corretamente com 23,31% de <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.	54
Figura 5.17 – Caso de teste 5 com extintor. Nenhuma imagem reconhecida. À esquerda está a imagem da aplicação e à direita a imagem que deveria ser selecionada no banco de dados, com 5,63% de <i>matches</i>	55

Figura 5.18 – Caso de teste 6 com extintor. Nenhuma imagem reconhecida. À esquerda está a imagem da aplicação e à direita a imagem que deveria ser selecionada no banco de dados, com 5,15% de <i>matches</i>	56
Figura 5.19 – Caso de teste 7 com extintor. Imagem reconhecida corretamente com 20,67% de <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.	56
Figura 5.20 – Caso de teste 8 com extintor. Nenhuma imagem reconhecida. À esquerda está a imagem da aplicação e à direita a imagem que deveria ser selecionada no banco de dados, com 9,31% de <i>matches</i>	57
Figura 5.21 – Caso de teste 9 com extintor. Nenhuma imagem reconhecida. À esquerda está a imagem da aplicação e à direita a imagem que deveria ser selecionada no banco de dados, com 6,22% de <i>matches</i>	57
Figura 5.22 – Porcentagem de reconhecimento dos testes realizados, utilizando o extintor como alvo	58
Figura 5.23 – Resultados obtidos nos testes realizados com o extintor	58
Figura 5.24 – Caso de teste 1 com porta de saída. Imagem reconhecida corretamente com 100% de <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.....	59
Figura 5.25 – Caso de teste 2 com porta de saída. Imagem reconhecida corretamente com 42,45% de <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.....	60
Figura 5.26 – Caso de teste 3 com porta de saída. Imagem reconhecida corretamente com 26,26% de <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.....	60
Figura 5.27 – Caso de teste 4 com porta de saída, imagem reconhecida incorretamente. (1) Relação entre a imagem capturada pela aplicação e a do banco de dados que deveria ter sido selecionada, com 28,72% dos <i>matches</i> . (2) Relação entre a imagem capturada pela aplicação e a imagem escolhida incorretamente com 46,62% dos <i>matches</i>	61
Figura 5.28 – Caso de teste 5 com porta de saída, imagem reconhecida incorretamente. (1) Relação entre a imagem capturada pela aplicação e a do banco de dados que deveria ter sido selecionada, com 11,08% dos <i>matches</i> . (2) Relação entre a imagem capturada pela aplicação e a imagem escolhida incorretamente com 27,81% dos <i>matches</i>	61
Figura 5.29 – Caso de teste 6 com porta de saída, imagem reconhecida incorretamente. (1) Relação entre a imagem capturada pela aplicação e a do banco de dados que deveria ter sido selecionada, com 8,64% dos <i>matches</i> . (2) Relação entre a imagem capturada pela aplicação e a imagem escolhida incorretamente com 24,94% dos <i>matches</i>	62
Figura 5.30 – Caso de teste 7 com porta de saída. Imagem reconhecida corretamente com 28,81% de <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.....	63
Figura 5.31 – Caso de teste 8 com porta de saída. Imagem reconhecida corretamente com 22,84% de <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.....	63
Figura 5.32 – Caso de teste 9 com porta de saída. Imagem reconhecida corretamente com 16,72% de <i>matches</i> . À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.....	64

Figura 5.33 – Porcentagem de reconhecimento dos testes realizados, utilizando a porta de saída como alvo	64
Figura 5.34 – Resultados obtidos nos testes realizados com a porta de saída	65
Figura 5.35 – Porcentagem de reconhecimento dos testes realizados com a foto capturada na distância ideal	65
Figura 5.36 – Porcentagem de reconhecimento dos testes realizados com a foto capturada um metro atrás da distância ideal	66
Figura 5.37 – Porcentagem de reconhecimento dos testes realizados com a foto capturada dois metros atrás da distância ideal	66
Figura 5.38 – Tempo total de execução com imagens de treinamento de 1280x720 <i>pixels</i> ..	69
Figura 5.39 – Tempo de reconhecimento e comunicação com imagens de treinamento de 1280x720 <i>pixels</i>	69
Figura 5.40 – Tempo total de execução com imagens de treinamento de 640x360 <i>pixels</i> ...	70
Figura 5.41 – Tempo de reconhecimento e comunicação para imagens de treinamento de 640x360 <i>pixels</i>	71
Figura 5.42 – Imagens de treinamento que descrevem a porta de saída. (1) Imagem de treinamento 5. (2) Imagem de treinamento 43.	72
Figura 5.43 – Relação entre tempo de execução e número de imagens de treinamento	73
Figura 5.44 – Tempo de reconhecimento e comunicação alterando o número de imagens de treinamento e utilizando a sacada como alvo	74
Figura 5.45 – Tempo de reconhecimento e comunicação alterando o número de imagens de treinamento e utilizando o elevador como alvo	75
Figura 5.46 – Tempo de reconhecimento e comunicação alterando o número de imagens de treinamento e utilizando o extintor como alvo.....	75

LISTA DE TABELAS

Tabela 5.1 – Descrição dos casos de teste realizados no teste de acurácia	45
Tabela 5.2 – Resultados do teste de acurácia	67
Tabela 5.3 – Configuração da rede durante a realização dos testes	68
Tabela 5.4 – Configuração da rede durante a realização dos testes	70
Tabela 5.5 – Configuração da rede durante a realização dos testes	71
Tabela 5.6 – Resultado dos testes realizados alterando o número de imagens do banco de dados	73

LISTA DE ABREVIATURAS E SIGLAS

API	<i>Application Programming Interface</i>
BD	Banco de Dados
HTTP	<i>Hypertext Transfer Protocol</i>
OpenCV	<i>Open Source Computer Vision Library</i>
PHP	<i>Personal Home Page</i>
PNG	<i>Portable Network Graphics</i>
SIFT	<i>Scale-Invariant Feature Transform</i>
SURF	<i>Speed Up Robust Features</i>
URL	<i>Uniform Resource Locator</i>

SUMÁRIO

1 INTRODUÇÃO.....	15
2 VISÃO COMPUTACIONAL	18
2.1 Pré-processamento de imagens	19
2.2 Segmentação	21
2.2.1 Segmentação por Limiarização	21
2.2.2 Segmentação por Regiões	22
2.2.2.1 Técnica baseada em Crescimento de Regiões	22
2.2.2.2 Algorítmo <i>K-means</i>	23
2.3 Extração de características	24
2.3.1 <i>Sobel</i>	25
2.3.2 <i>Prewitt</i>	25
2.4 Reconhecimento de padrões	26
2.4.1 Descritor <i>Scale Invariant Feature Transform</i>	26
2.4.2 Descritor <i>Speeded Up Robust Features</i>	27
2.4.3 Método Bayesiano	28
3 TRABALHOS RELACIONADOS	29
4 IMPLEMENTAÇÃO	33
4.1 Aplicação móvel.....	34
4.2 Servidor.....	35
4.3 Reconhecimento de imagens	36
4.3.1 Leitura de imagens	37
4.3.2 Detectar pontos chaves e descritores	37
4.3.3 Escolha da melhor imagem.....	39
4.3.4 Treinamento de imagens	41
4.3.5 Comparação de imagens	41
5 RESULTADOS	44
5.1 Teste de acurácia	44
5.1.1 Teste para o reconhecimento do elevador.....	45
5.1.2 Teste para o reconhecimento do extintor	52
5.1.3 Teste para o reconhecimento da porta de saída	59
5.1.4 Considerações finais	64
5.2 Teste de tempo de execução	68
5.2.1 Testes realizados com imagens de treinamento com 1280x720 <i>pixels</i>	68
5.2.2 Testes realizados com imagens de treinamento com 640x360 <i>pixels</i>	70
5.3 Teste com a alteração do número de imagens de treinamento	71
6 CONSIDERAÇÕES FINAIS.....	77
6.1 Trabalhos futuros	77
REFERÊNCIAS	79

1 INTRODUÇÃO

O sistema visual do corpo humano é bastante complexo, veloz e adaptativo, pois depende diretamente do cérebro e de outros órgãos do corpo. O olho é capaz de perceber e interpretar objetos em uma imagem de forma muito rápida. Isso acontece no cortex visual do cérebro, uma das partes mais complexas no sistema de processamento cerebral. Entretanto, alguns cientistas concentram seus estudos com intuito de compreender o funcionamento da parte do cérebro que comanda todas estas funções, para então trazer ideias para a computação [11].

Para as pessoas com deficiência no sistema visual, transitar de forma segura, eficiente e autônoma é um de seus desafios. Ao mesmo tempo em que estas pessoas enfrentam dificuldades, ocorre o crescimento e evolução de áreas tecnológicas como a visão computacional. Este avanço proporciona o surgimento de uma nova gama de aplicações que podem ser utilizadas para reduzir tais dificuldades. Essas aplicações podem conceder uma melhoria na qualidade de vida de deficientes visuais, ampliando seus horizontes, criando novas possibilidades e oferecendo uma maior independência, comprovando que elas são capazes de desenvolver atividades desafiadoras que até então eram complexas [17, 31].

É perceptível a desvantagem na navegação de pessoas com deficiência visual, incluindo a existência de uma alta dependência para que elas possam orientar-se em locais internos, pelo fato de não possuírem informações necessárias para direcionar-se. A informação de contexto é essencial para que os indivíduos naveguem através de ambientes familiares ou desconhecidos com base em mapas externos e orientações verbais [21]. Sendo assim, locais onde a acessibilidade é escassa prejudicam a navegação destas pessoas, fazendo com que um simples passeio em um prédio se torne uma atividade perigosa. Para estes casos, novas alternativas com o intuito de auxiliar e amenizar as adversidades tornam-se bem-vindas.

As pessoas utilizam sistemas-guia para orientar-se no espaço, por exemplo, no decorrer de um trajeto, alguns orientam-se pela rua, pelos prédios ou pelas placas no caminho. A visão é um dos melhores desses sistemas de navegação, portanto, os deficientes visuais precisam recorrer a outros tipos de sistema-guia. Por exemplo, o tipo da calçada, as curvas e esquinas do trajeto, ou sistemas-guia adaptados como os pisos táteis [31].

Neste contexto, percebe-se a dificuldade encontrada por essas pessoas para orientar-se, embora os locais contenham sistemas-guia, a pessoa pode ainda sentir-se desorientada por não ter noção do contexto de onde ela está. Por exemplo, mesmo com um piso-tátil alertando, a

pessoa com deficiência visual provavelmente não saberá que está na frente de uma escada.

Salienta-se o fato que ambientes despreparados dificultam a vida de pessoas com deficiência visual. Para uma maior inclusão social é importante que os locais sejam estruturados e totalmente adaptados para estas dificuldades. Este trabalho apresenta uma aplicação para auxiliar nestes casos, servir de sistema-guia caso a pessoa esteja com dificuldade ou o local não seja acessível e principalmente, dar mais independência na orientação durante a locomoção.

A visão também é considerada a grande promotora da integração do indivíduo em atividades motoras, perceptivas e mentais e a ausência da mesma pode provocar marcantes alterações, diminuindo sua capacidade de adaptação na sociedade [6, 31]. Deste modo, este trabalho auxilia na integração da pessoa no meio comum, possibilitando novas experiências e reduzindo as dificuldades.

Trata-se de uma alternativa para amenizar o transtorno na locomoção destas pessoas com uma significativa redução nos custos. Isso porque a utilização da visão computacional para orientar as pessoas com deficiência visual no interior de prédios torna-se um meio barato comparado com a colocação de pisos-táteis e placas em braile em todo o local. Em determinados casos, a instalação de pisos-táteis ou outras marcações é inviável, é o caso de um prédio muito antigo ou histórico [6].

Este trabalho tem como objetivo a criação de uma aplicação para auxiliar os portadores de deficiência visual a se orientarem contextualmente em ambientes internos. Para isso o usuário utiliza uma câmera para registrar, através de uma foto, o que tem na sua frente. A foto é comparada com imagens do local que foram previamente armazenadas em um banco de dados juntamente com sua descrição. Ao encontrar uma imagem do banco de dados semelhante à foto capturada pelo usuário, sabe-se sobre o local e o contexto onde a pessoa está através da descrição armazenada.

A Figura 1.1 apresenta uma visão geral do trabalho. O processo inicia-se com a captura de uma foto pelo usuário, conforme ilustra a Figura 1.1-1. Esta imagem representa o que está na frente do usuário e é capturada toda vez que o usuário deseja saber o contexto onde está. Após ser capturada, a imagem é enviada para um servidor que será responsável pela execução do sistema onde acontece o reconhecimento do local, como é mostrado na Figura 1.1-2. O envio para o servidor ocorre pela internet e todos os procedimentos de reconhecimento ocorrem no servidor.

Este reconhecimento é realizado através da utilização de técnicas de visão

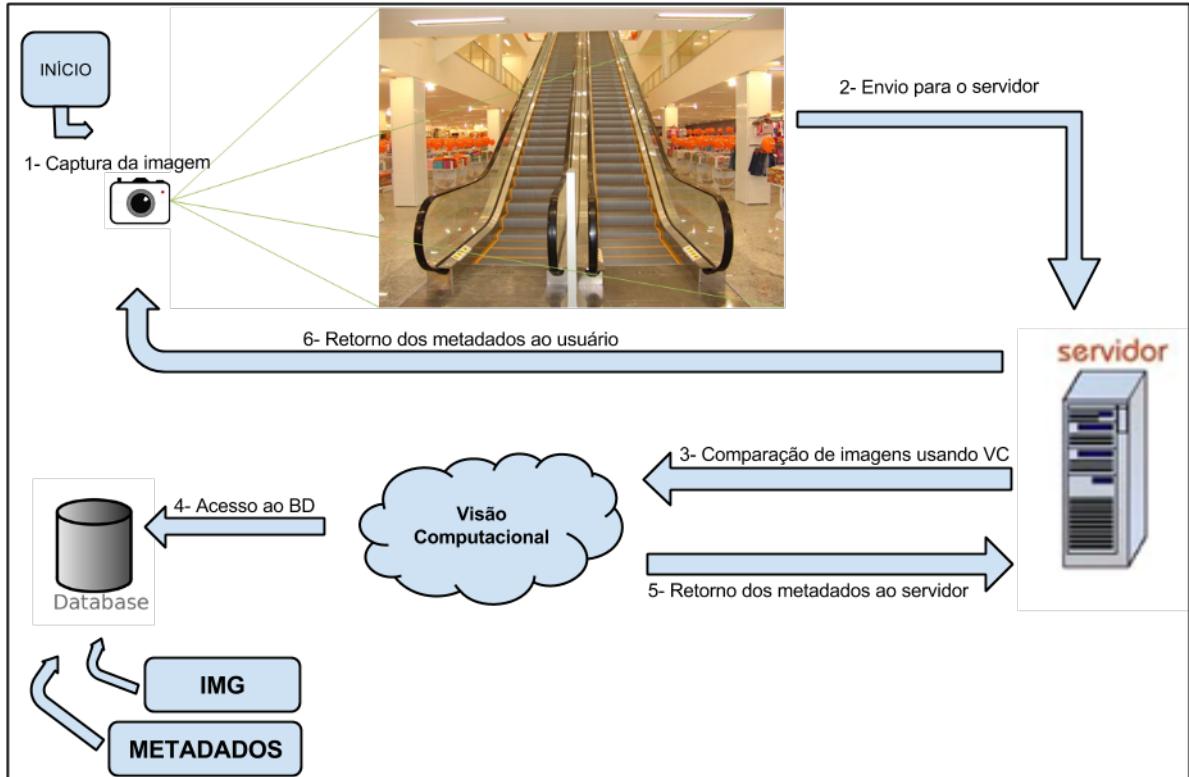


Figura 1.1: Estrutura e visão geral do projeto. 1- Captura da imagem, 2- Envio para o servidor, 3- Comparação de imagens usando visão computacional, 4- Acesso ao BD, 5- Retorno dos metadados ao servidor, 6- Retorno dos metadados ao usuário.

computacional. O servidor executa uma aplicação de reconhecimento de imagens, como ilustra a Figura 1.1-3. A aplicação que utiliza visão computacional para o reconhecimento de imagens realiza uma consulta a um banco de dados, mostrado na Figura 1.1-4.

O banco de dados contém imagens e metadados que descrevem o local onde o usuário estiver utilizando a aplicação, sendo que este banco de dados é populado antes do usuário utilizar a aplicação. Para todas as imagens do banco de dados, realiza-se a comparação delas com a imagem capturada pelo usuário, procurando-se semelhanças. A imagem com maior semelhança é escolhida e, em teoria, representa o mesmo local que o usuário está localizado no prédio.

Finalmente, a aplicação de visão computacional encaminha para o servidor os metadados correspondentes à imagem escolhida no banco de dados, como ilustra a Figura 1.1-5. A aplicação no dispositivo móvel, por fim, recebe os metadados e os informa ao usuário.

2 VISÃO COMPUTACIONAL

Este capítulo aborda os conhecimentos necessários para o entendimento da implementação e validação do projeto. Apresenta-se a visão computacional como uma subárea da Ciência da Computação, com destaque para diversos conceitos a respeito da manipulação de imagens e vídeos.

A visão computacional tem como objetivo treinar e ensinar computadores a enxergar como humanos, fornecendo ao computador informações precisas através de imagens ou vídeos, de forma que seja possível executar tarefas inteligentes, simulando a inteligência humana [19, 27, 32].

Os sistemas de visão computacional vêm se desenvolvendo muito nos últimos anos. Esta área trata da extração de informações das imagens e da identificação e classificação de objetos ou formas presentes na imagem. Estes sistemas envolvem diversas análises de imagens e técnicas de inteligência artificial ou de tomada de decisão, que permitem a identificação e classificação de objetos ou imagens [11, 25, 32].

O reconhecimento da importância da visão computacional se deu com a ideia de que os computadores poderiam ser usados para simular processamentos complexos de percepção visual. Até então, acreditava-se que a visão e outras formas de percepção e cognição eram exclusivas de seres vivos, ou seja, não desenvolvíveis através de máquinas. Entretanto, com o avanço das pesquisas, percebe-se que esta ideia não está mais correta [11].

Seguindo o objetivo de identificar objetos em uma imagem, baseado em análise e processamento de Imagens, a Visão Computacional pode ser estruturada com seis principais etapas, que seguem uma ordem, como mostra a Figura 2.1 [11, 25] .

Supondo que o *Problema*, destacado no topo da imagem, são todas as informações de entrada do sistema e os *Resultados*, destacado na base da imagem, como sendo o resultado esperado pelo sistema, por exemplo a identificação de um objeto em uma imagem, as demais etapas podem ser definidas a seguir [11, 25]:

- Aquisição: É a etapa na qual a imagem é adquirida para ser utilizada no sistema. Pode ser realizada por sensores, ou de forma digital por uma câmera.
- Pré-Processamento: Utiliza-se a imagem capturada no passo anterior e aplica-se operações com o intuito de remover ou reduzir falhas e imperfeições da imagem. Outra

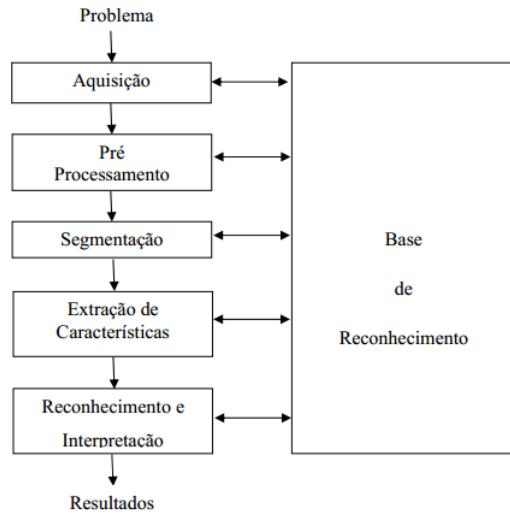


Figura 2.1: Estrutura de um sistema de visão computacional e suas etapas [25]

função é aplicar técnicas para, se necessário, transformar a imagem para facilitar na Segmentação e na Extração de Características, mais detalhes são apresentados na seção 2.1

- Segmentação: Responsável por dividir sub-imagens da imagem inicial de acordo com sua significância, ou seja, divide os objetos de interesse da imagem, apresentado na seção 2.2.
- Extração de características: Nesta etapa são extraídas as características que são classificadas como importantes para o reconhecimento da imagem. A extração é feita através de descritores e podem ocorrer de diversos modos. Ao final da extração, é possível caracterizar com precisão cada objeto de interesse.
- Reconhecimento e interpretação: Por fim, aplicam-se algoritmos para reconhecer e atribuir rótulos nas imagens de acordo com as características extraídas. Além disso, atribui-se significado às características e faz-se um pós processamento, dando sentido à extração e reconhecimento das características.

2.1 Pré-processamento de imagens

Uma imagem pode ser considerada como uma representação visual de objetos, podendo ser adquirida através de fotos, filmes, cenas, entre outras representações, ou geradas através de pinturas ou desenhos. Atualmente, as imagens digitais encontram-se difundidas em diversas aplicações [11, 27].

Até a imagem capturada realmente ser utilizada, ela pode passar por uma sequência de algoritmos para aperfeiçoá-la, para melhor extrair as informações desejadas. Isso facilita a realização das etapas posteriores de reconhecimento de padrões ou extração de conhecimento das imagens. O que se busca é a remoção de ruídos para que, com isso, a imagem fique clara para ser descrita facilmente [11, 33].

As técnicas de processamento de imagens devem ser escolhidas de acordo com sua importância para o resultado esperado. Uma técnica importante é chamada de Realce da Imagem, cujo principal objetivo é processar uma imagem de modo que a imagem resultante seja mais adequada que a imagem original, para uma aplicação específica. Esta técnica pode ser aplicada diretamente sobre a matriz de *pixels* que a imagem é digitalizada, ou utilizando uma base matemática no domínio da frequência, obtida com base no teorema da convolução [33, 27].

Um exemplo de técnica de Processamento de Imagens que utiliza apenas somas e divisões, é chamada de Filtro da Média, cujo objetivo é reduzir o ruído em uma imagem. Ela é aplicada diretamente nos *pixels* da imagem, sendo que cada pixel recebe uma média aritmética do nível de cinza dos *pixels* vizinhos [11, 33, 32]. Esta técnica faz com que detalhes da imagem sejam perdidos, como mostra a Figura 2.2: na imagem à direita, após a aplicação da técnica de Filtro de Média, detalhes como as penas do chapéu são abstraídos da imagem original à esquerda .



Figura 2.2: Aplicação da técnica de filtro de média. À esquerda a imagem original, à direita resultado da aplicação do filtro de média [33]

Em imagens coloridas, percebe-se facilmente a importância do processamento de imagens, pois em um reconhecimento de padrões em análise automática de imagens, a cor pode ser um significativo descritor das propriedades de um objeto, podendo assim simplificar

sua identificação e segmentação [25].

2.2 Segmentação

Para facilitar a identificação de padrões ou objetos presentes em uma imagem, é necessário “separar” este objeto dos demais elementos contidos na imagem. Esta etapa denominada segmentação, é fundamental para o processo de reconhecimento [11, 33, 30].

A segmentação consiste em particionar a imagem de modo que seja possível explicitar regiões representativas da imagem. O objetivo é fazer com que os objetos e as áreas de interesse em uma imagem tenham os seus *pixels* agrupados e destacados dos demais [11, 15]. Estas partições são obtidas a partir dos valores de tom de cinza (ou de cores) de cada pixel que forma a imagem, de texturas semelhantes, de formas predefinidas, ou de outros padrões [15].

Em geral, os algoritmos de Segmentação de Imagens são baseados nas descontinuidades ou nas similaridades. Nas descontinuidades, a segmentação pode ser representada particionando a imagem de acordo com mudanças bruscas nos tons de cinza dos *pixels*. Essa abordagem é utilizada principalmente para detecção de pontos isolados, de linhas ou de bordas na imagem. Na segmentação por similaridades, as principais abordagens baseiam-se em limiarização e em regiões. Ambos os métodos são melhor explicados nas seções 2.2.1 e 2.2.2 [11, 33].

2.2.1 Segmentação por Limiarização

A limiarização consiste em separar as regiões do fundo da imagem. Normalmente aplica-se sobre objetos escuros sobre fundos claros, visto que esta técnica é eficiente quando aplicada a objetos com tons de cinza muito distintos [30].

Na limiarização analisa-se a similaridade dos níveis de cinza da imagem extraíndo os objetos de interesse através da seleção de um limiar L que separa os agrupamentos de níveis de cinza. Uma imagem limiarizada $g(x, y)$ é definida como mostra a equação :

$$g(x, y) = 1 \text{ se } f(x, y) \geq L \quad (2.1)$$

$$g(x, y) = 0 \text{ se } f(x, y) < L \quad (2.2)$$

Onde $f(x, y)$ corresponde ao nível de cinza do ponto, os *pixels* rotulados com 1 correspondem aos objetos, os *pixels* rotulados com 0 correspondem ao fundo e L é um valor de tom de cinza predefinido denominado limiar [33]. A Figura 2.3 apresenta a aplicação da técnica

de limiarização: à esquerda está a imagem original e, à direita a imagem após a técnica aplicada. Observa-se que a imagem à direita teve como resultado um objeto (fotógrafo) destacado do fundo, através da aplicação da limiarização.



Figura 2.3: Aplicação de limiarização. À esquerda a imagem original, à direita o resultado da aplicação da limiarização sobre a imagem original [11]

2.2.2 Segmentação por Regiões

As técnicas que são baseadas em regiões geralmente são associadas à identificação de grupos de *pixels* ou áreas da imagem. Uma região em uma imagem é um grupo de *pixels* conectados que possuem propriedades similares. Uma imagem pode conter vários objetos, e cada objeto pode conter várias regiões correspondentes a partes diferentes [11, 15].

2.2.2.1 Técnica baseada em Crescimento de Regiões

Na segmentação por crescimento de regiões, inicia-se com um conjunto de *pixels* que formam uma região, chamada “semente”. Esta região cresce através de um processo interativo de agrupamento de *pixels* ou regiões com propriedades similares [25]. Para cada semente, avalia-se a satisfação da propriedade nos *pixels* vizinhos (ou região). Como exemplo de propriedade pode-se ter: cor em RGB com menos de 5% de variação. Conforme a propriedade for satisfeita, os *pixels* (ou regiões) são acrescentados à semente, formando uma região [11]. Em aplicações totalmente automáticas, no caso de não ser possível a definição de um ponto semente inicial, todas as regiões da imagem podem ser consideradas sementes em relação à propriedade [24].

A Figura 2.4 representa um exemplo de crescimento de Região. A Figura 2.4a contém a imagem $[I(x, y)]$ com semente $I(s)$ e critério: $|I(x, y) - I(s)| \leq 10\%$, ou seja, um pixel que possui diferença maior que 10% não é inserido na mesma região da semente. Na Figura

2.4b inicia-se o crescimento de *pixels* que satisfazem a propriedade inicial. Neste momento, os *pixels* agrupam-se com *pixels* vizinhos similares aumentando a região. Na Figura 2.4c é mostrado um estado intermediário de crescimento, na qual expandem-se os *pixels* e inicia-se a formação de uma região similar. Por fim na Figura 2.4d a região é crescida por completo, já que cresceu até o limiar de onde foi inserida, dando a tonalidade das bordas, definindo e separando-a das regiões vizinhas [11].

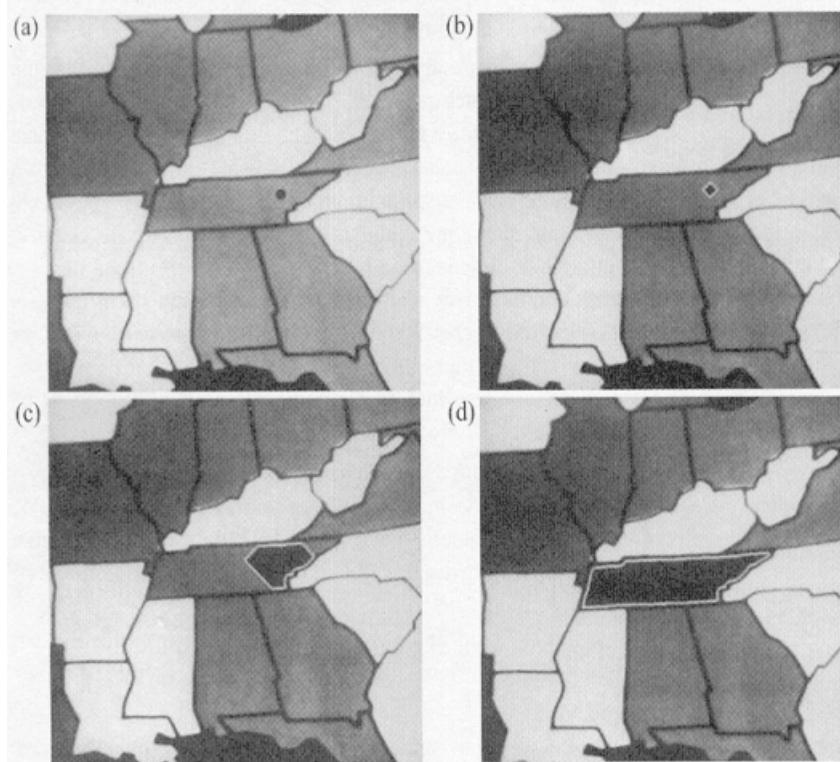


Figura 2.4: Aplicação da técnica de crescimento de região. a) Início da técnica com aplicação da semente. b) Início do crescimento da região. c) Nível intermediário da técnica. d) Região completa. [11]

2.2.2.2 Algoritmo *K-means*

O algoritmo *K-means* pertence a um grupo de técnicas baseadas em clusterização ou agrupamentos. Estas técnicas tem como objetivo agrupar elementos de um conjunto de tal forma que os grupos gerados sejam constituídos de elementos que possuam maior similaridade. Por exemplo, dado um conjunto X , uma técnica de clusterização consiste em agrupar os elementos de X , de modo que objetos mais similares fiquem no mesmo grupo ou *cluster* e os objetos menos similares sejam alocados em grupos distintos [1].

Aplicado em imagens, os *pixels* são chamados de elementos, os grupos de elementos

significam as regiões que possuem maior similaridade na imagem e o elemento que representa cada grupo é chamado de centro ou centróide. O número de agrupamentos a serem formados k deve ser definido para cada caso [1, 11].

O algoritmo é dividido em quatro passos, primeiramente são determinadas as posições iniciais dos k centróides dos grupos, que pode ser feito de forma aleatória. Após isso, aloca-se cada elemento ao grupo com centróide mais próximo de acordo com uma estimativa inicial das coordenadas, geralmente calculada com distância euclidiana [1, 11].

A seguir, o algoritmo recalcula os centros dos grupos a partir dos elementos alocados. A nova estimativa das coordenadas dos centróides é calculada pela média aritmética das coordenadas dos pontos associados a cada grupo. Estes passos são repetidos segundo um critério de convergência. Usualmente, esse critério é a soma dos quadrados das distâncias de cada grupo, quanto menor for este valor, mais homogêneos serão os elementos dentro de cada grupo e melhor será a partição [11].

2.3 Extração de características

O próximo passo após a segmentação dos objetos é a escolha de características que deverão ser extraídas. Nesta etapa, é necessário estabelecer quais informações são relevantes para o reconhecimento do objeto e quais informações deverão ser extraídas a partir dos contornos da imagem. O resultado é usado para o treinamento do reconhecimento de padrões em outras imagens com outras perspectivas [8, 29].

Os métodos de extração de características são fundamentados no reconhecimento de bordas. Na maioria das vezes são baseados em contornos, linhas e curvas, mas também podem ser baseados em padrões das características ou na correspondência estrutural considerando a geometria dos elementos que foram segmentados [8].

Alguns métodos de reconhecimento de bordas representam a segmentação da imagem juntamente com a extração de características. Este é o caso dos algoritmos de Gradiente de *Sobel* citado na seção 2.3.1 e *Prewitt* citado na seção 2.3.2, que utilizam a primeira derivada de uma imagem para o reconhecimento das bordas [8, 29].

2.3.1 Sobel

O algoritmo Sobel calcula diferenças finitas e resulta em uma aproximação do gradiente da intensidade dos pixels de determinada imagem. Para fazer isso, para cada pixel da imagem calcula-se o gradiente da intensidade, resultando na variação de claro para escuro naquela posição e na variação de luminosidade em cada ponto. Com isso, é possível estimar uma transição de claro para escuro definindo fronteiras entre objetos [20].

Para encontrar o gradiente, multiplicam-se duas matrizes 3x3 com a imagem original. Estas matrizes são chamadas de máscaras e são multiplicadas com cada pixel da imagem para descobrir a intensidade verticalmente e horizontalmente [20]. A Figura 2.5 representa estas máscaras que multiplicam cada pixel. O termo central da matriz multiplica o pixel escolhido e o complemento da matriz multiplica os pixels vizinhos, resultando no gradiente de cada pixel.

$$\begin{array}{c} \begin{array}{|c|c|c|} \hline -1 & 0 & +1 \\ \hline -2 & 0 & +2 \\ \hline -1 & 0 & +1 \\ \hline \end{array} & \begin{array}{|c|c|c|} \hline +1 & +2 & +1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array} \\ \text{Gx} & \text{Gy} \end{array}$$

Figura 2.5: Cálculo do gradiente de *Sobel*. Gx: Calcula máscara verticalmente - Gy: Calcula máscara horizontalmente [20]

2.3.2 Prewitt

O algoritmo de *Prewitt* é semelhante ao operador de *Sobel*, diferindo na utilização de um valor ao invés de dois que multiplicam o termo central nas linhas da máscara. A Figura 2.6 representa as máscaras utilizadas por este algoritmo para multiplicar os *pixels* e gerar seus gradientes verticalmente e horizontalmente [20].

$$\begin{array}{c} \begin{array}{|c|c|c|} \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline \end{array} & \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -1 & -1 \\ \hline \end{array} \\ \text{Gx} & \text{Gy} \end{array}$$

Figura 2.6: Cálculo do gradiente de *Prewitt*. Gx: Calcula máscara verticalmente - Gy: Calcula máscara horizontalmente [20]

Ao utilizar um valor ao invés de dois como ocorre no algoritmo de *Sobel*, o algoritmo de *Prewitt* tende a deixar o resultado do operador menos suave e menos sensível a ruídos [20, 11]. Esta diferença é pequena e pode ser visualizada na Figura 2.7. A Figura 2.7a apresenta a imagem

original, a Figura 2.7b apresenta a imagem após a aplicação de uma abordagem da técnica de *Sobel* e a Figura 2.7c apresenta a imagem após a aplicação de uma abordagem da técnica de *Prewitt*. Nota-se a pequena diferença na suavidade das técnicas no resultado encontrado no cabelo.

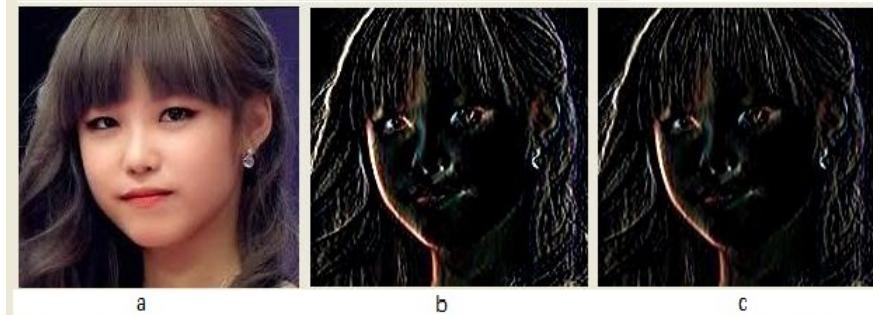


Figura 2.7: a) Imagem original. b) Aplicação da técnica de *Sobel* sobre a imagem original. c) Aplicação da técnica de *Prewitt* sobre a imagem original [11]

2.4 Reconhecimento de padrões

Em geral, os algoritmos para reconhecimento de padrões são criados utilizando-se conceitos de Inteligência Artificial. Além disso, operam tanto no domínio espaço quanto no domínio frequência. Os de domínio espaço são baseados no processamento das coordenadas (x, y) da imagem alvo enquanto os algoritmos do domínio frequência utilizam o espectro da imagem obtido através da Transformada Rápida de Fourier [9].

Estes algoritmos, também chamados de categorizadores, em sua maioria executam duas etapas distintas, que são treinamento e teste. Durante a etapa de treinamento, os dados são enviados para o classificador e serão usados para adaptar um processo de decisão de análise estatística para a distinção das categorias. Dentre os diversos métodos para classificação de padrões, destacam-se os classificadores apresentados a seguir [12].

2.4.1 Descritor *Scale Invariant Feature Transform*

O método SIFT (*Scale Invariant Feature Transform*) permite a detecção de descritores locais a partir de características extraídas. Estes descritores são utilizados para fazer correspondência entre diferentes visões de um objeto ou de uma cena [22, 23].

O método espera uma imagem e características filtradas desta imagem para fazer um mapeamento, então estipula pontos chaves nestas figuras baseando-se geralmente em medidas

de estabilidade. Após isso, orientam-se os pontos chaves em relação à orientação, escala e localização de cada ponto chave na imagem [22, 23].

A Figura 2.8 representa o processo no qual ocorre o reconhecimento entre duas imagens. Ela é dividida em duas imagens que contém o mesmo desenho, porém em contextos diferentes. Na figura da esquerda, o desenho está centralizado, aproximado da câmera enquanto na figura à direita ele está rotacionado, longe da câmera e em um contexto diferente.

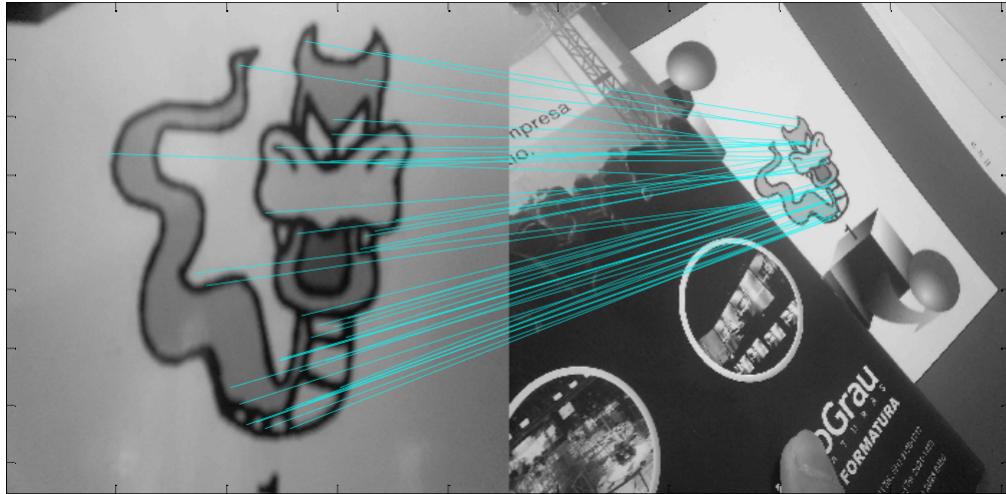


Figura 2.8: Processo de reconhecimento de padrões entre duas imagens através da técnica SIFT [22]

Após a separação e identificação dos pontos-chave, realiza-se a construção de descritores locais para uma região a cada ponto-chave. Esta operação é realizada para duas imagens na qual deseja-se encontrar as semelhanças, procurando-se os pontos correspondentes em cada uma, como mostra a Figura 2.8 [22, 23].

Esta operação é chamada de *matching*. Trata-se da comparação entre pontos-chaves baseados na similaridade de seus descritores correspondentes. Os descritores geralmente são comparados utilizando distância Euclidiana [23].

2.4.2 Descritor *Speeded Up Robust Features*

O método SURF (*Speeded Up Robust Features*) é um descritor inspirado no método SIFT, pois trata-se de técnicas de correspondência baseadas em características locais, entretanto, este método possui particularidades e seu processo de detecção baseia-se na matriz Hessiana [3, 5].

Após a imagem possuir pontos chaves, os descritores SURF são computados através da construção de uma janela quadrada com orientação e centrada em cada ponto chave. Esta

janela é dividida em uma área de dimensões 4x4 e aplica-se a técnica de matriz Hessiana novamente para cada sub-região [2, 3, 34]. O resultado da aplicação do determinante na matriz Hessiana, em duas imagens, geram pontos de máximo, na qual são utilizados como descritores e representam a correlação de cantos e regiões entre as imagens. Esta correlação é ilustrada na Figura 2.9, na qual através de duas imagens iguais são encontrados relações com os mesmos pontos chaves.[7, 13].

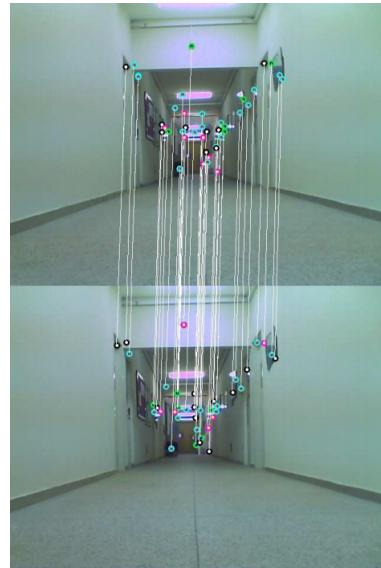


Figura 2.9: Correlações encontradas em duas imagens utilizando o método SURF [28]

2.4.3 Método Bayesiano

O método *Naïve Bayes* é utilizado geralmente em reconhecimento de textos. Pode ser considerado como um classificador probabilístico, pois cada segmento da imagem é escolhida de forma independente a partir de uma distribuição e após isso cada categoria é selecionada de acordo com as probabilidades anteriores [12].

Para categorizar uma imagem, o método assume um conjunto de segmentos da imagem rotuladas $I\{\}$ e um vocabulário $V\{\}$ de pontos-chave a serem reconhecidos, ou seja, características da segmentação [12].

Aplica-se o método em novas imagens e para cada descritor extraído e reconhecido na imagem, rotula-se um ponto chave. Após isso conta-se N vezes a ocorrência de pontos chaves na imagem, para confirmar o reconhecimento [12].

3 TRABALHOS RELACIONADOS

O projeto *VizWiz*, apresentado por [4], é destinado para que as pessoas com deficiência visual recrutem auxiliares com visão remota para ajudá-las em quase tempo real. As pessoas cegas utilizam *VizWiz* em seus celulares. Com suas câmeras os usuários tiram uma foto, fazem uma pergunta e em seguida recebem respostas faladas. Atualmente, estas respostas são fornecidas por trabalhadores de sites de *Crowdsourcing*, onde permite-se que indivíduos utilizem sua inteligência humana para realizar tarefas que atualmente são complicadas para os computadores realizarem, por exemplo, criar um comentário para explicar o contexto de uma foto.

O sistema, apresentado na Figura 3.1 divide-se em três etapas e é utilizado como uma aplicação para celulares iPhones. Na primeira etapa o usuário tira uma foto, após isso o usuário fala uma pergunta e o celular utiliza leitor de voz para interpretá-la. Na terceira etapa o usuário fica à espera de uma resposta.

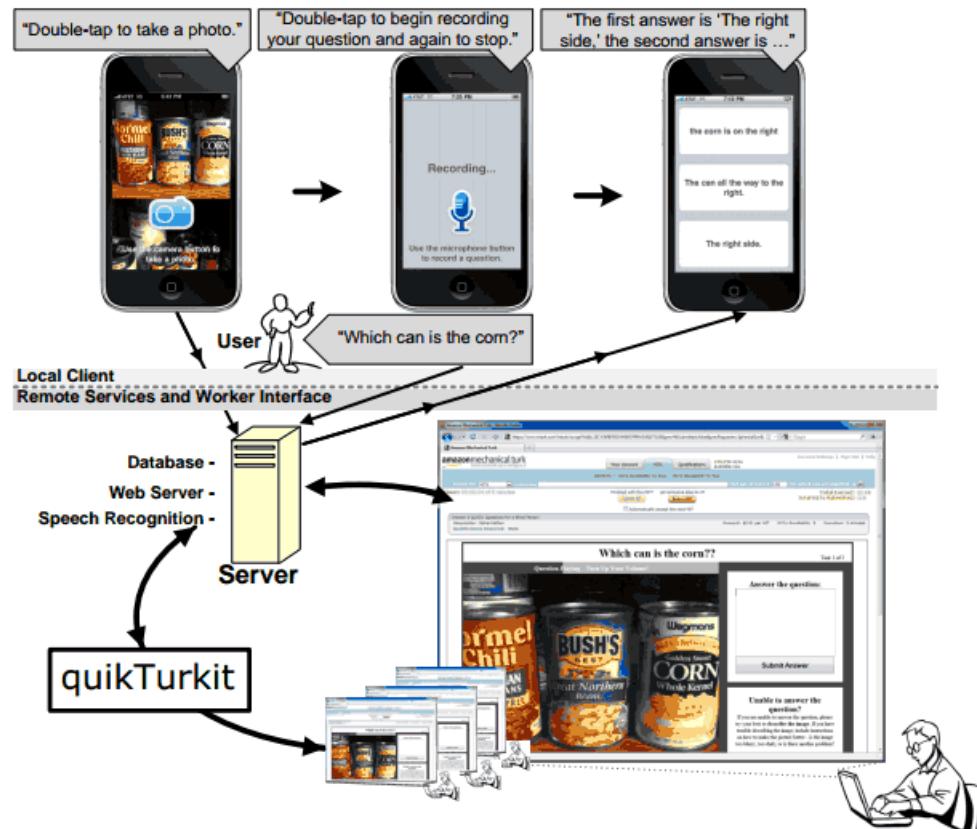


Figura 3.1: Estrutura e funcionamento do sistema *VizWiz* [4]

De acordo com a Figura 3.1, os componentes do sistema incluem um servidor web para encaminhar a pergunta aos trabalhadores, um serviço de reconhecimento de voz que converte

as perguntas faladas em texto e um banco de dados que contém perguntas e respostas. Também utiliza o serviço *quikTurkit* que é responsável por manter configurações específicas como o número mínimo de respostas por pergunta ou um conjunto de trabalhadores para determinado tempo de espera.

Este projeto serviu como base para impulsionar pesquisas que utilizam o *Google Glass*. Os autores [35], utilizam o *Google Glass* para desenvolver aplicações que podem auxiliar usuários com deficiência visual a identificar objetos e ambientes. A primeira aplicação desenvolvida, permite que os usuários deficientes visuais tirem uma foto e façam uma pergunta. Então envia-se para a rede estes dados e espera-se respostas de usuários via *Twitter* ou da *Amazon Mechanical Turk Platform* [35].

A segunda aplicação, chamada de *Memento*, utiliza a visão computacional e o reconhecimento de imagens. Primeiramente, usuários gravam imagens e comentários dos objetos ou locais. Quando uma pessoa com deficiência visual utiliza o *Google Glass* e se aproxima do mesmo local, a aplicação reconhece o objeto ou a cena e lê o comentário pré-armazenado [35].

Outro trabalho que utiliza técnicas para reconhecimento de contexto é o projeto *Drishti*, desenvolvido por [16], é um complexo sistema de navegação para pessoas com deficiência visual. Ele inclui várias tecnologias como computadores portáteis, reconhecimento de voz, banco de dados estáticos e dinâmicos, redes sem fio, Sistemas de Informação Geográfica e GPS. O sistema tem como objetivo aumentar a informação contextual para os deficientes visuais e calcular rotas otimizadas com base na preferência do usuário, incluindo restrições e obstáculos como congestionamentos e bloqueios no caminho. Além de ter o foco voltado para auxiliar na navegação de pessoas com deficiência visual, ele pode ser facilmente estendido para suportar múltiplas aplicações, tais como guias turísticos.

Este trabalho descreve uma ideia para orientação de pessoas com deficiência visual, pois ele leva em conta o caminho a ser tomado pelo usuário, voltado especialmente a ambientes externos, porém não concentra atenção a utilizar visão computacional para identificar objetos ou locais a sua frente. A Figura 3.2 mostra um protótipo do hardware utilizado no sistema.

Outro sistema de navegação que tem como objetivo orientar uma pessoa com deficiência visual é o projeto *Navigation System for the Blind*. Neste projeto, [21], cria-se um sistema portátil e independente, que permite com que pessoas com deficiência visual naveguem em ambientes familiares e não familiares, sem o auxílio de outras guias.



Figura 3.2: Protótipo do sistema *Drishti* [16]

O projeto foi dividido em três módulos principais. O módulo inicial está relacionado com a posição e a orientação do usuário, baseando-se num GPS e numa bússola eletrônica. O segundo módulo é representado por um computador contendo o sistema. Este sistema contém o banco de dados com informações espaciais como prédios, calçadas, árvores e outros detalhes. Por fim, o terceiro módulo é a interface do utilizador, na qual o sistema utiliza as informações gravadas no banco de dados, juntamente com informações da localização atual para orientar, direcionando e transmitindo informações do seu redor para o usuário. A Figura 3.3 representa a estrutura deste sistema, separando os três módulos principais.

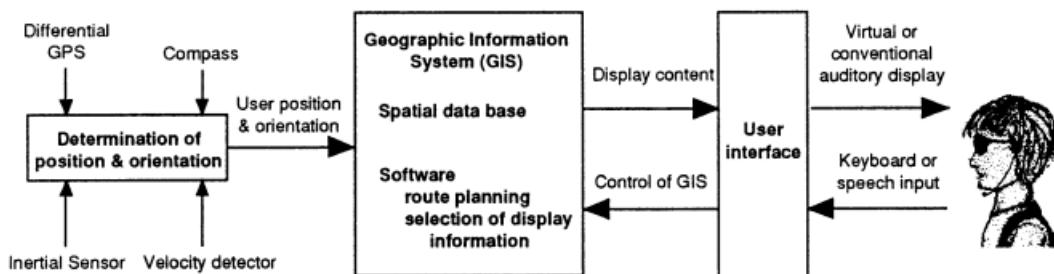


Figura 3.3: Estrutura e componentes funcionais do sistema de navegação para cegos [21]

Este sistema visa orientar e direcionar a pessoa com deficiência visual em trajetos externos, mostrando obstáculos e caminhos a se tomar. As limitações das configurações do

GPS e do custo de instalação e de manutenção da rede em relação a cobertura alcançada são dificuldades apontadas pelo autor.

Por fim, um trabalho de processamento de imagens, capaz de detectar textos em imagens de cenários naturais, apresentado por [14], propõe um sistema que lê o texto encontrado em cenas naturais e tem objetivo de prestar assistência às pessoas com deficiência visual. Quando uma pessoa com deficiência visual estiver caminhando na rua, pode ser essencial saber que existe uma placa escrito "Pare" a sua frente.

Para realizar o projeto, foi utilizado o método de segmentação *k-means*. Após isso o sistema foi treinado e testado para identificar letras, através de duas abordagens: Classifier Naive Bayes e o Machine Vector Support, sendo que, os melhores resultados foram encontrados com a última.

4 IMPLEMENTAÇÃO

A implementação deste trabalho foi estruturada em três módulos, que são a aplicação do dispositivo móvel, o servidor responsável por integrar e gerenciar algumas funcionalidades do processo e o módulo de reconhecimento das imagens, utilizado pelo servidor.

O módulo inicial é uma aplicação executada no dispositivo móvel. Primeiramente, o usuário irá capturar uma foto, após isso a aplicação é responsável por empacotar e enviar esta foto para um servidor e aguardar por uma resposta, como mostra a Figura 4.1-1.

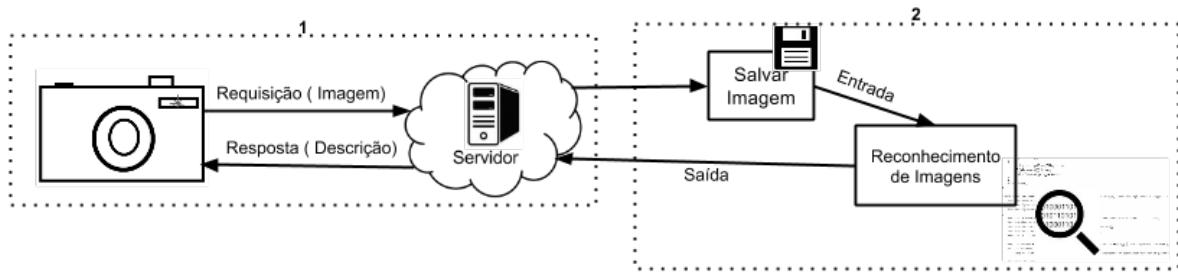


Figura 4.1: Estrutura geral da aplicação. (1) Aplicação móvel. (2) Fluxo de dados no servidor

A aplicação no dispositivo móvel é responsável por enviar ao servidor uma imagem representando o local onde o usuário está, além de esperar uma resposta do servidor, que representa uma possível descrição da imagem enviada. A cada foto capturada pelo usuário, o dispositivo envia uma nova requisição para o servidor e aguarda por uma nova resposta. Detalhes sobre o desenvolvimento móvel estão explicados na seção 4.1.

O segundo módulo é o servidor, que deve estar interligado com o aplicativo móvel para o funcionamento da solução. Ele é responsável por gerenciar as funcionalidades durante toda a execução do sistema. Primeiramente, o servidor aguarda por uma requisição, como mostra a Figura 4.1-1. Assim que o servidor recebe uma nova requisição, ele decodifica a mensagem e salva a imagem que foi encaminhada. Após isso, autoriza a execução da aplicação responsável pelo reconhecimento das imagens, visto na Figura 4.1-2.

Assim que a aplicação de reconhecimento de imagens gera uma resposta, o servidor é responsável por encaminhar esta resposta pela requisição enviada do dispositivo móvel. Detalhes sobre o desenvolvimento do servidor estão explicados na seção 4.2.

Por fim, o terceiro módulo representa o reconhecimento das imagens, na qual através de uma imagem recebida como entrada, aplicam-se técnicas de visão computacional para reconhecer uma semelhante dentre uma sequência de outras imagens. A saída deste módulo

é um metadado correspondente à imagem reconhecida, explicado na seção 4.3. As seções a seguir explicam em detalhes o desenvolvimento de cada módulo.

4.1 Aplicação móvel

O módulo da aplicação móvel foi desenvolvido utilizando-se a plataforma *Android*, através de recursos como o acesso à câmera do aparelho, configuração e envio de imagens para a rede via POST e leitura e reprodução de voz a partir de um texto.

Quando o usuário deseja saber o contexto do local onde ele está, é necessário pressionar a tela para que a aplicação tire uma foto. Assim que uma foto é capturada, a aplicação é responsável por ajustar a imagem, comprimindo-a e convertendo-a para o formato PNG, com o intuito de diminuir o atraso no envio à rede, para então enviar ao servidor. A Figura 4.2 representa a sequência de passos que ocorrem quando uma foto é capturada e empacotada para estar configurada de acordo com o servidor.

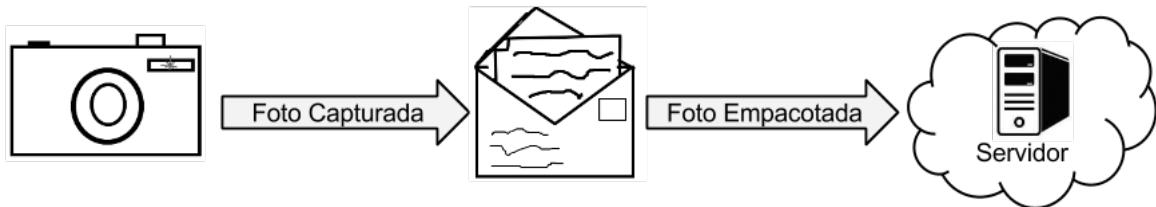


Figura 4.2: Sequência de passos para uma foto capturada

Para enviar ao servidor, utilizou-se o protocolo HTTP, através de uma requisição POST. Trata-se de um método de comunicação entre uma aplicação e uma URL [18]. Ou seja, para realizar o *upload* de informações, primeiramente cria-se uma conexão com o servidor através de uma URL fixa. A aplicação tenta se conectar com essa URL correspondente para se comunicar com o servidor.

Após a conexão ser estabelecida e a imagem estar em um tamanho reduzido e no formato PNG, a aplicação codifica a imagem para um vetor de bytes. Esse formato permite que o servidor consiga decodificar e obter a imagem enviada. Com a imagem codificada, a aplicação envia uma requisição e aguarda por uma resposta do servidor.

O retorno que o servidor encaminha para a aplicação móvel é uma resposta para a requisição encaminhada. Trata-se de um texto que descreve a respectiva imagem no banco de dados. Quando a resposta do servidor é lida, a aplicação é responsável por encaminhar esta

descrição ao usuário, conforme mostra a Figura 4.3. Pelo fato do público alvo ser pessoas com deficiência visual, utilizou-se uma API de leitura e reprodução de textos, para que o usuário possa ouvir o metadado encaminhado, conforme é mostrado na Figura 4.3. Após o usuário ouvir a descrição da última foto, ele pode tirar novas fotos para o reconhecimento.

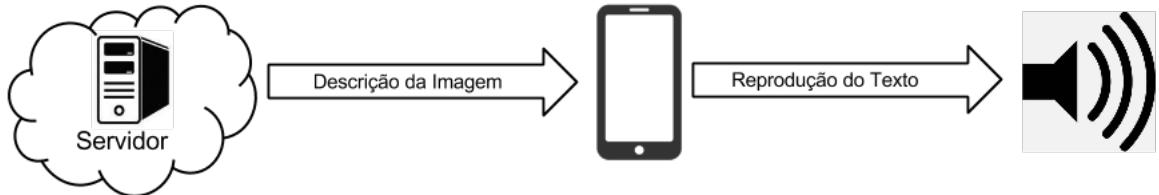


Figura 4.3: Sequência de passos quando chega uma resposta do servidor

4.2 Servidor

O módulo responsável pelo servidor foi desenvolvido utilizando-se a linguagem PHP. A aplicação faz o recebimento de informações via URL, leitura e escrita de arquivos. Conforme visto na seção 4.1, a aplicação móvel faz o *upload* da imagem através de uma requisição POST, codificando a imagem em um vetor de bytes. Portanto, o servidor foi configurado utilizando as mesmas técnicas, aguardando uma requisição POST, que ao chegar é decodificada para obter a imagem enviada.

O servidor possui uma URL, na qual a aplicação móvel irá se conectar e enviar as requisições. Assim que uma nova requisição é recebida, o servidor interpreta a mensagem enviada pela aplicação, identificando as informações recebidas, como mostra a Figura 4.4-1.

Após isso, o servidor salva em disco a imagem recebida para que o módulo de reconhecimento a utilize posteriormente, como é mostrado na Figura 4.4-2. Com a imagem salva, o servidor executa o módulo de reconhecimento de imagens. Portanto, assim que o servidor recebe uma requisição, ele atualiza os dados que serão utilizados no módulo de reconhecimento de imagens e autoriza sua execução, como mostra a Figura 4.4-3.

Em seguida, o servidor aguarda a aplicação de reconhecimento de imagens gerar uma saída, que correspondente à descrição da imagem recebida pela aplicação móvel. Assim que uma saída é gerada, o servidor é encarregado de encaminhar esta saída para a aplicação móvel através de uma resposta para a requisição recebida, como é apresentado na Figura 4.5.

Outra função atribuída para o servidor é armazenar os metadados e as imagens que são utilizadas para o treinamento. As imagens e metadados descrevem o local onde a aplicação

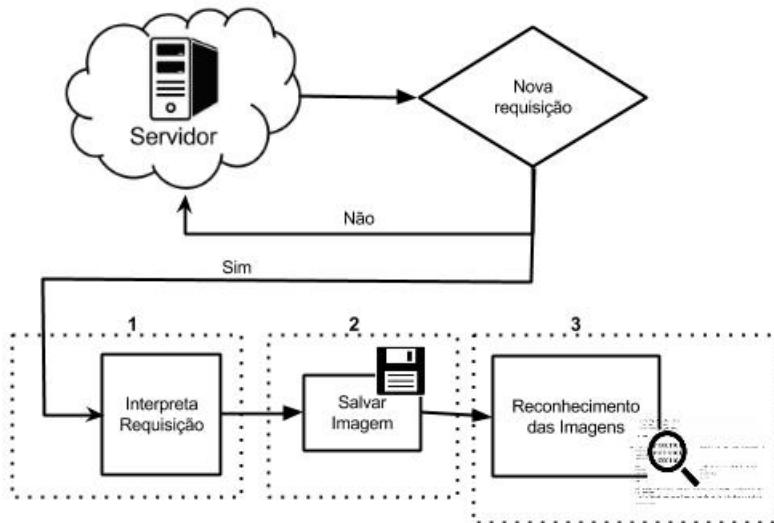


Figura 4.4: Sequência de passos para cada requisição

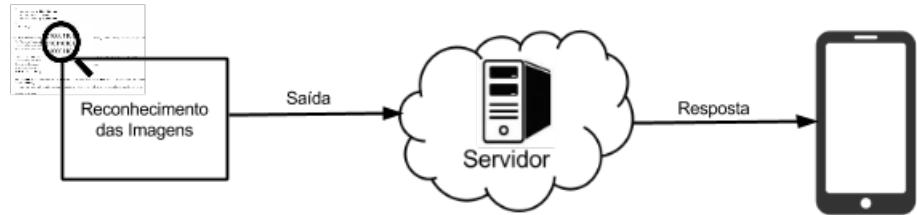


Figura 4.5: Sequência de passos após a resposta do módulo de reconhecimento de imagens

está sendo utilizada e são coletadas previamente pelo administrador do sistema. As imagens são utilizadas para comparação e busca por semelhanças, já os metadados são descrições das respectivas imagens que contextualizam o local. Portanto, essas informações são armazenadas em um diretório e são utilizados pelo módulo de reconhecimento de imagens.

4.3 Reconhecimento de imagens

O módulo responsável pelo reconhecimento das imagens foi desenvolvido utilizando-se a linguagem C++. Também utilizou-se a biblioteca de processamento de imagens e vídeos OpenCV. O reconhecimento de uma imagem semelhante pode ser visto através de uma sequência de funcionalidades, representada na Figura 4.6, onde primeiramente ocorre a leitura das imagens, explicada na seção 4.3.1. Após a leitura das imagens ocorre a detecção de pontos chaves e descritores das imagens, abordados na seção 4.3.2. Então é realizada a comparação entre todas as imagens com a imagem *query*, que é a imagem capturada pelo dispositivo móvel, para que seja possível escolher a imagem armazenada no servidor mais semelhante com a *query*.

Com a escolha realizada, os metadados correspondentes à imagem escolhida são retornados ao usuário, como é apresentado na seção 4.3.3.

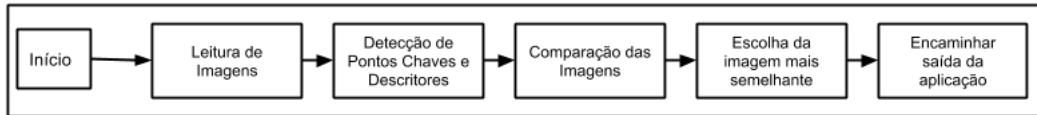


Figura 4.6: Sequência de passos que ocorrem na aplicação de reconhecimento de imagens

4.3.1 Leitura de imagens

As imagens utilizadas para o reconhecimento são divididas em imagens de treinamento, que são as imagens gravadas no banco de dados do servidor e a imagem *query*, capturada pelo usuário durante a utilização do sistema. As imagens de treinamento descrevem o local onde a aplicação está sendo utilizada e devem ser capturadas e armazenadas antes que o usuário utilize a aplicação. As imagens de treinamento são importantes para o sucesso do reconhecimento, pois a aplicação só reconhece o local e seu contexto, através de imagens previamente cadastradas. A imagem *query* é capturada pelo usuário, quando ele deseja saber o contexto ou o que tem em sua frente. Esta imagem é capturada e enviada através da aplicação móvel para o servidor, que a encaminha para a aplicação de reconhecimento de imagens.

Nesta etapa, as imagens são carregadas e armazenadas em uma lista, de acordo com um caminho fornecido para a aplicação. Criam-se neste momento as variáveis que serão utilizadas durante todo o processo de reconhecimento, conforme ilustra a Figura 4.7.

4.3.2 Detectar pontos chaves e descritores

É nesta etapa que são processados os pontos chaves e descritores de cada imagem, ou seja, são avaliadas as características e propriedades que descrevem cada imagem, como cores e formas. A Figura 4.8, à esquerda, contém uma placa com características, cores e formas que a definem unicamente. Após a aplicação das técnicas de detecção de pontos chaves e descritores, é possível reconhecer estas propriedades e categorizar unicamente uma imagem conforme suas características, conforme ilustra a Figura 4.8 à direita. Nela os pontos chaves são destacados em amarelo e são utilizados posteriormente para realizar a comparação com outras imagens.

Após cada imagem ser descrita como um vetor de descritores e pontos chaves é necessário comparar as imagens com a *query* para descobrir quais imagens possuem mais

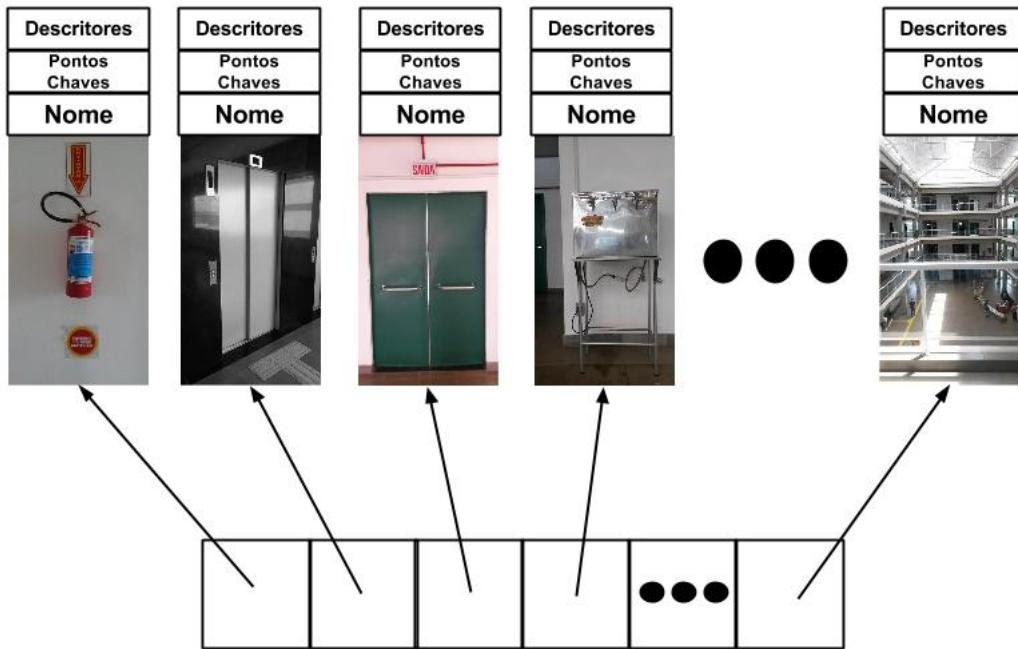


Figura 4.7: Lista de imagens carregadas pela aplicação

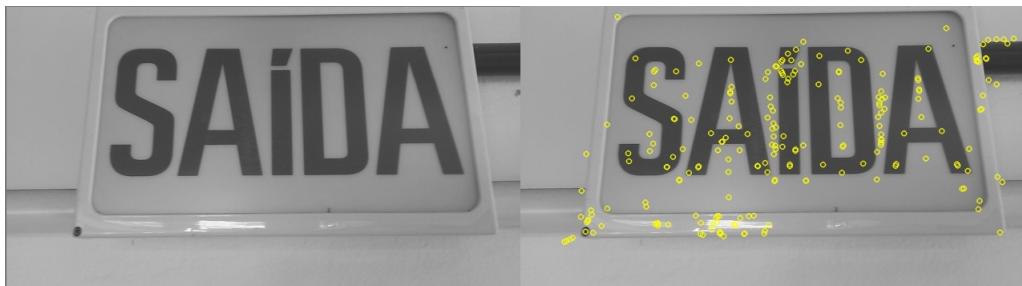


Figura 4.8: Exemplo de descritores em uma imagem.

pontos em comum, ou seja, qual imagem é mais semelhante com a capturada pelo usuário. Para isso, são criadas correlações entre pontos chaves. Essas relações são chamadas de *matches* e são utilizadas para definir qual imagem possui mais pontos chaves em comum com a *query*. As *matches* são criadas através de fórmulas, descritas na técnica SURF e explicadas na seção 2.4.

A Figura 4.9-A ilustra a imagem com seus pontos chaves destacados em círculos amarelo. A Figura 4.9-B apresenta uma visualização de como as *matches* são criadas, em azul a relação entre os pontos chaves. Conforme mostra a Figura 4.9, para duas imagens iguais todos os pontos chaves são relacionados, ou seja, tem-se 100% de relacionamento entre as imagens. Para o caso em que as imagens possuem alguns pontos em comum o número de *matches* diminui, conforme ilustra a Figura 4.10. Quando uma imagem não possui nenhuma semelhança para a técnica utilizada, não são criados *matches*, conforme ilustra a Figura 4.11.

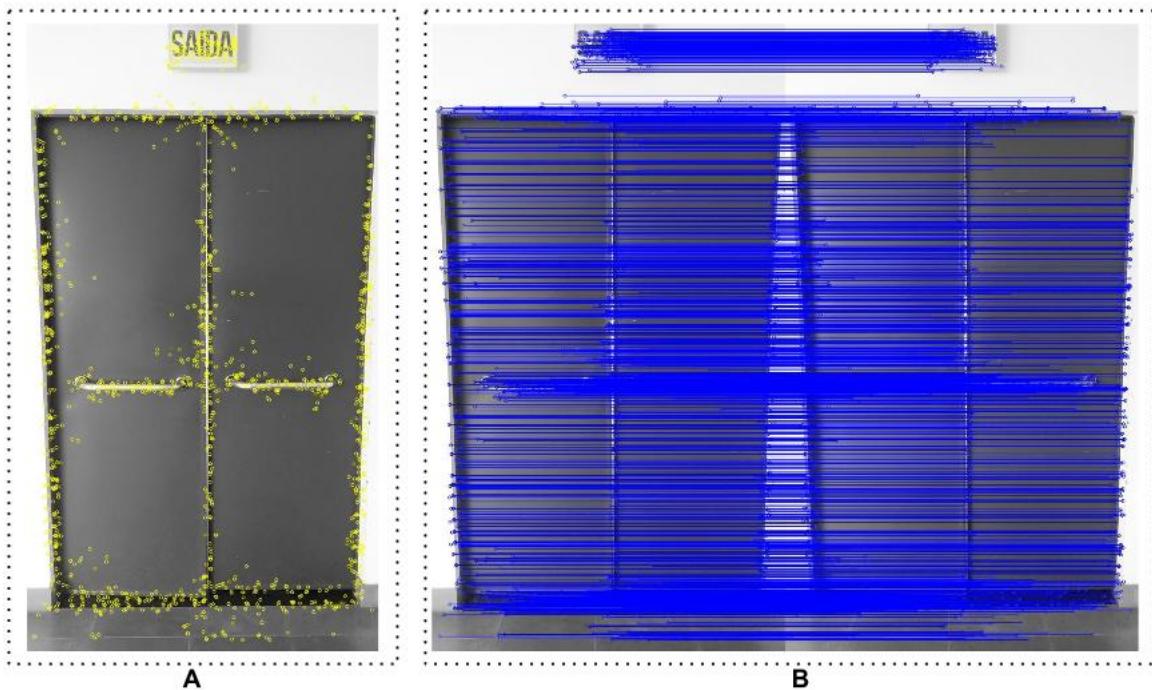


Figura 4.9: Exemplo de comparação entre duas imagens iguais. (A) Imagem original.(B) Aplicação das técnicas

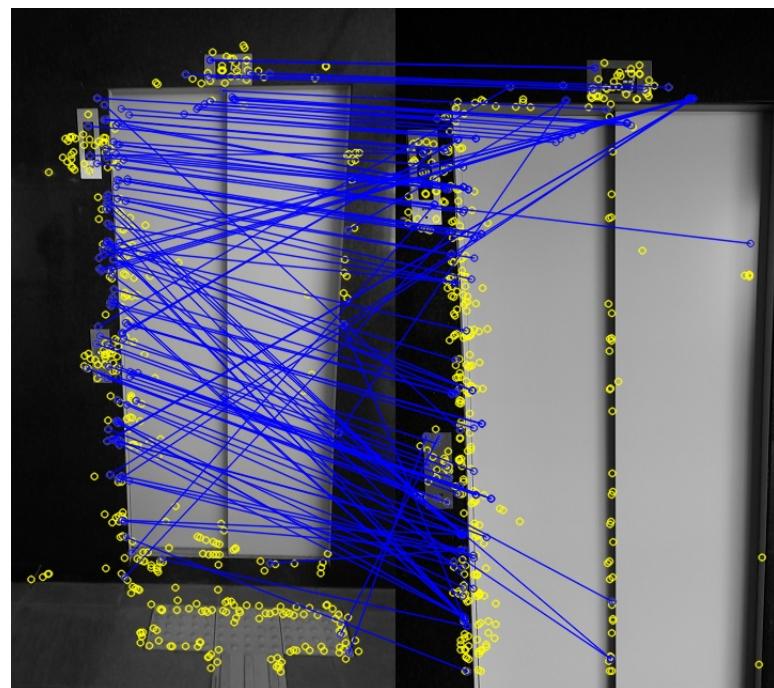


Figura 4.10: Exemplo de comparação entre duas imagens semelhantes

4.3.3 Escolha da melhor imagem

Com o término da comparação entre as imagens, tem-se como resultado o número de *matches* das imagens de treinamento com a imagem *query*. Então faz-se um cálculo de

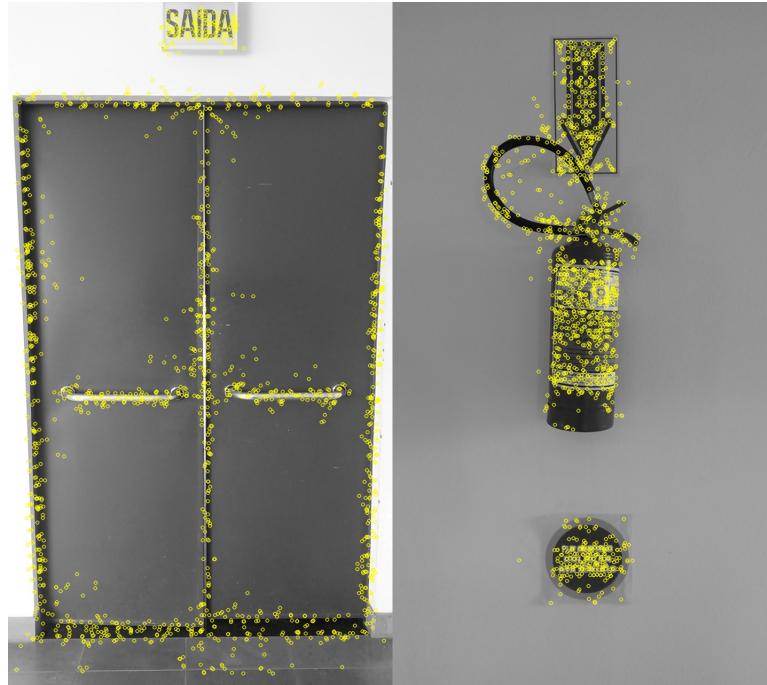


Figura 4.11: Exemplo de comparação entre duas imagens diferentes, sem a criação de *matches*

porcentagem para realizar a escolha da melhor imagem, utilizando o número de descritores e o número de *matches* da imagem de treinamento com a imagem *query*. A imagem que possuir a maioria de seus descritores formando *matches* com a imagem *query* terá uma porcentagem maior de semelhanças com a *query* do que uma que contenha a minoria de seus descritores formando *matches*. A imagem do banco de dados que possuir a maior porcentagem é considerada a mais semelhante em relação a imagem de *query*. Na hipótese de nenhuma imagem do banco de dados ter um percentual maior que 10% nenhuma imagem é selecionada.

Após uma imagem do banco de dados ser escolhida, a aplicação de reconhecimento encaminha para o servidor o metadado correspondente, que descreve a imagem escolhida. Conforme demonstra a Figura 4.12, a imagem escolhida possui como metadado "Você está na frente do elevador", que é enviado para o servidor. O servidor, por fim, é responsável por encaminhar para a aplicação móvel o metadado recebido.

As etapas descritas representam os passos realizados na aplicação de reconhecimento de imagens. A fim de otimizar a execução, o reconhecimento foi dividido em duas etapas: treinamento das imagens armazenadas no banco de dados, mostrado na seção 4.3.4 e comparação das imagens, explicado na seção 4.3.5. A etapa de treinamento ocorre apenas no primeiro carregamento da aplicação, enquanto a etapa de comparação, vista na seção 4.3.5, é realizada para toda requisição enviada pelo usuário ao servidor. Ou seja, assim que o usuário

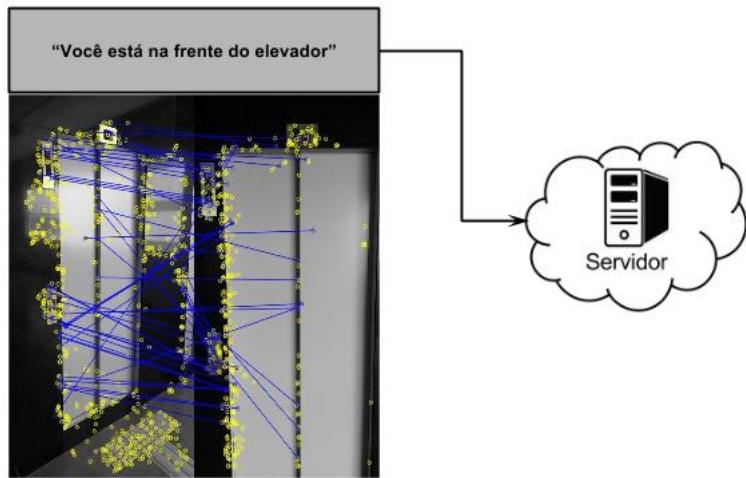


Figura 4.12: Envio dos metadados correspondente a imagem escolhida ao servidor

deseja saber o contexto onde está, a aplicação de reconhecimento de imagens executa a leitura e detecção de pontos chaves e descritores somente para esta nova imagem *query*, realiza a comparação com as imagens de treinamento e retorna a descrição do local, para o servidor encaminhar ao usuário. As seções a seguir detalham as etapas de treinamento e comparação de imagens.

4.3.4 Treinamento de imagens

Primeiramente a aplicação carrega as imagens que serão utilizadas durante sua execução, conforme ilustra a Figura 4.13-1. Estas imagens são chamadas de imagens de treinamento e ficam armazenadas no servidor. Elas são capturadas antecipadamente e contextualizam o local onde o usuário está utilizando a aplicação.

Após isso, ocorre nestas imagens a criação de pontos chaves e descritores. Os pontos chaves e descritores categorizam uma imagem conforme suas características, assunto explicado na seção 4.3.2 e mostrado nos pontos amarelo da Figura 4.13-2. Ao realizar este passo, todas as imagens podem ser reconhecidas por um vetor de descritores, que são utilizadas posteriormente para verificar a semelhança com outras imagens.

4.3.5 Comparação de imagens

Com o término da etapa de treinamento, todas as imagens já estão representadas através de descritores. Passa-se então para a etapa de comparação. Para isso, todas as imagens de treinamento são comparadas com uma imagem encaminhada pelo servidor, que representa o

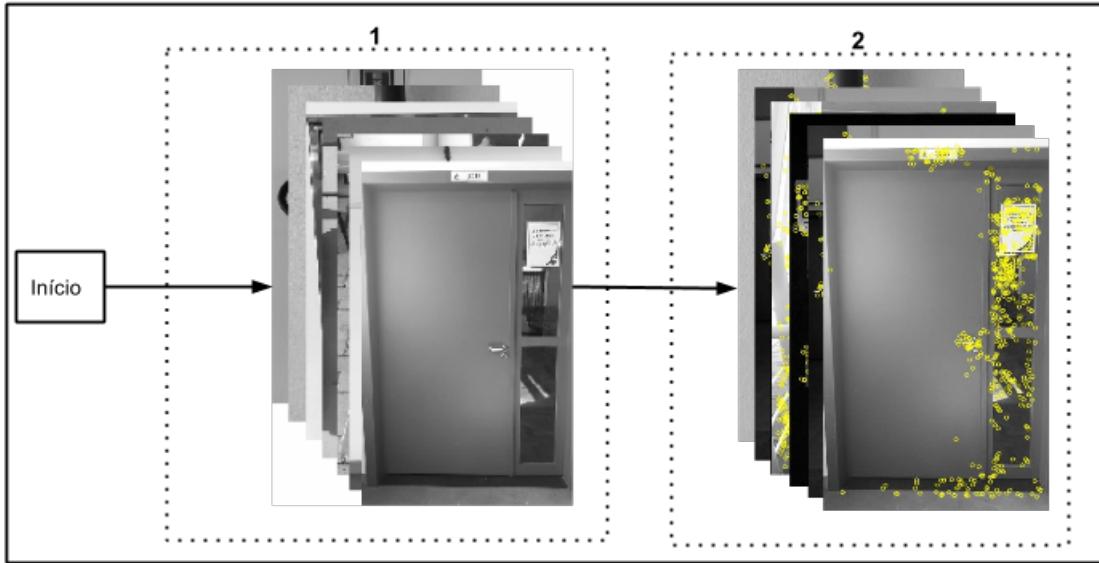


Figura 4.13: Sequência de passos para carregar as imagens de treinamento. (1) Imagens de treinamento. (2) Imagens de treinamento com destaque para os pontos chaves

local onde o usuário está. Esta imagem é chamada de *query* pela aplicação.

A imagem *query* é carregada pela aplicação de reconhecimento, conforme ilustra a Figura 4.14-1. Após isso, ocorre a criação de pontos chaves e descritores, do mesmo modo que é realizado nas imagens de treinamento, mostrado na Figura 4.14-2.

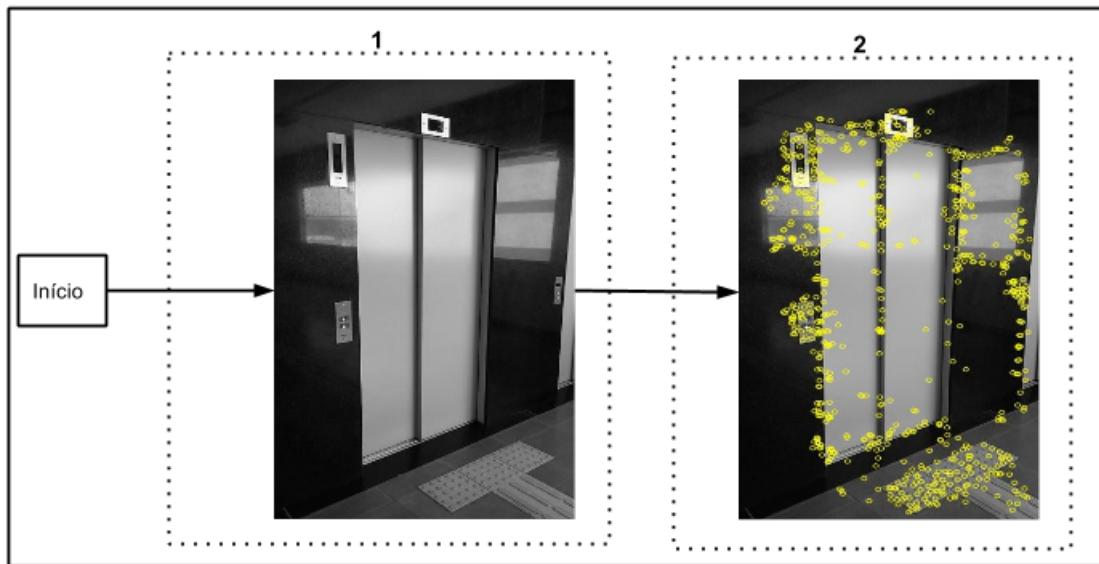


Figura 4.14: Sequência de passos para carregar a imagem *query*. (1) Imagem *query*. (2) Imagem *query* com destaque para os pontos chaves

Nesta etapa do processo de reconhecimento, todas as imagens do banco de dados já estão categorizadas em vetores de pontos chaves, conforme ilustra a Figura 4.15-1. Neste passo, a imagem *query* também está categorizada em vetores de pontos chaves, conforme ilustra a Figura

4.15-2. Então ocorre a comparação entre as imagens de treinamento com a imagem *query*, conforme ilustra a Figura 4.15-3.

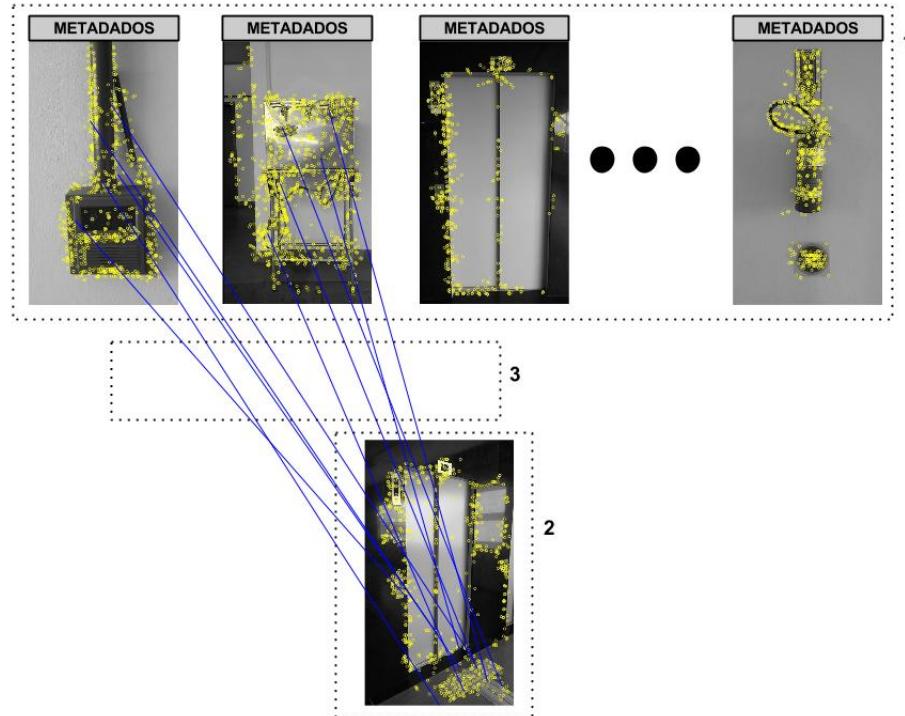


Figura 4.15: Comparação das imagens de treinamento com a imagem *query*

Com a comparação entre as imagens realizada, escolhe-se a imagem mais semelhante com a *query* e encaminha-se ao servidor o metadado correspondente a esta imagem, conforme demonstra a Figura 4.12. Após o término da implementação, a aplicação está pronta para o reconhecimento de diferentes locais. Para validar o seu funcionamento foram realizados diferentes testes, sendo que os resultados são apresentados na seção 5.

5 RESULTADOS

Para validar a aplicação foram realizados testes de acurácia, com o intuito de validar os casos em que o software funciona e os casos em que não alcança o objetivo. Com estes testes é possível identificar falsos positivos e negativos na utilização do sistema. Realizou-se um teste de tempo de execução, com o objetivo de avaliar se o software satisfaz o objetivo de proporcionar um melhor contexto na navegação dos deficientes visuais em tempo hábil. Também foi realizado um teste alternando o número de imagens de treinamento do banco de dados, para computar a acurácia e o tempo de execução da aplicação para diferentes casos. Os testes de acurácia são apresentados na seção 5.1, os testes de tempo de execução são demonstrados na seção 5.2 e os testes alterando o número de imagens de treinamento são mostrados na seção 5.3.

5.1 Teste de acurácia

O teste de acurácia tem como objetivo validar o funcionamento da aplicação proposta, simulando situações reais de utilização. Nesse contexto, a aplicação precisa reconhecer corretamente um cenário/objeto, informando-o ao usuário. Utilizou-se como base as dependências do prédio da Universidade Federal da Fronteira Sul (UFFS). No prédio da UFFS, foram capturadas vinte fotos para serem utilizadas no reconhecimento do local. Estas fotos foram ajustadas para 640x360 *pixels* e são mostradas na Figura 5.1. As fotos foram armazenadas no servidor e cada foto recebeu seu respectivo metadado, também armazenados no servidor, que descrevem os respectivos locais.

Após o armazenamento das fotos que descrevem o prédio da UFFS, foram escolhidas três elementos aleatoriamente para o reconhecimento, chamados de alvos:

- Elevador;
- Extintor;
- Porta de Saída.

Para validar o reconhecimento com estes três itens, cada um foi fotografado pela aplicação em ângulos e distâncias diferentes que correspondem a nove casos de teste para cada item, conforme ilustra a Tabela 5.1. As fotos diferem em ângulo, com fotos frontais e laterais à

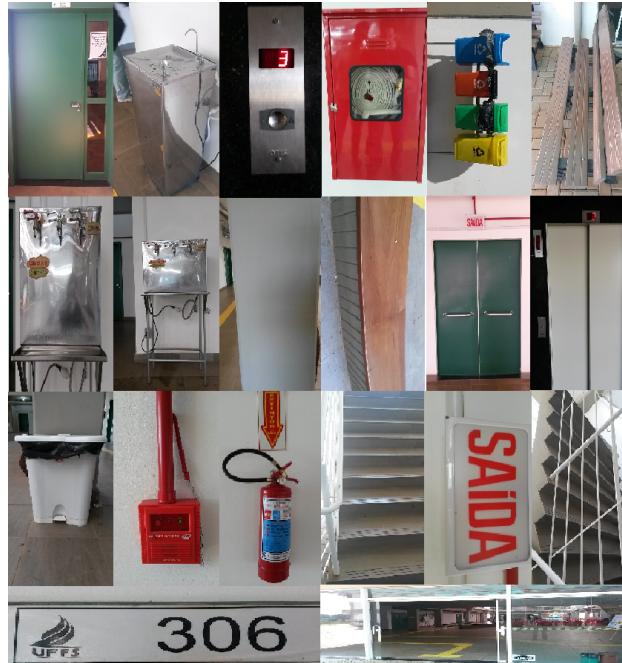


Figura 5.1: Imagens utilizadas para o teste de acurácia

esquerda e à direita. Há também três diferentes distâncias: a distância ideal, um metro atrás da distância ideal e dois metros atrás da distância ideal.

Caso de teste	Descrição
Caso 1	Foto frontal na distância ideal
Caso 2	Foto frontal um metro atrás da distância ideal
Caso 3	Foto frontal dois metros atrás da distância ideal
Caso 4	Foto lateral à esquerda na distância ideal
Caso 5	Foto lateral à esquerda um metro atrás da distância ideal
Caso 6	Foto lateral à esquerda dois metros atrás da distância ideal
Caso 7	Foto lateral à direita na distância ideal
Caso 8	Foto lateral à direita um metro atrás da distância ideal
Caso 9	Foto lateral à direita dois metros atrás da distância ideal

Tabela 5.1: Descrição dos casos de teste realizados no teste de acurácia

Os resultados encontrados nos testes com o elevador estão descritos na seção 5.1.1, os testes para o reconhecimento do extintor estão apresentados na seção 5.1.2 e os testes com a porta de saída são mostrados na seção 5.1.3.

5.1.1 Teste para o reconhecimento do elevador

No caso de teste 1, foto frontal na distância ideal, a imagem correta do banco de dados foi reconhecida com 100% dos *matches* criados entre a imagem capturada e a imagem encontrada no banco de dados, conforme ilustra a Figura 5.2. Neste caso, as demais imagens do banco

de dados que foram analisadas ficaram com 0% de *matches* e o resultado encaminhado para o usuário foi o esperado.

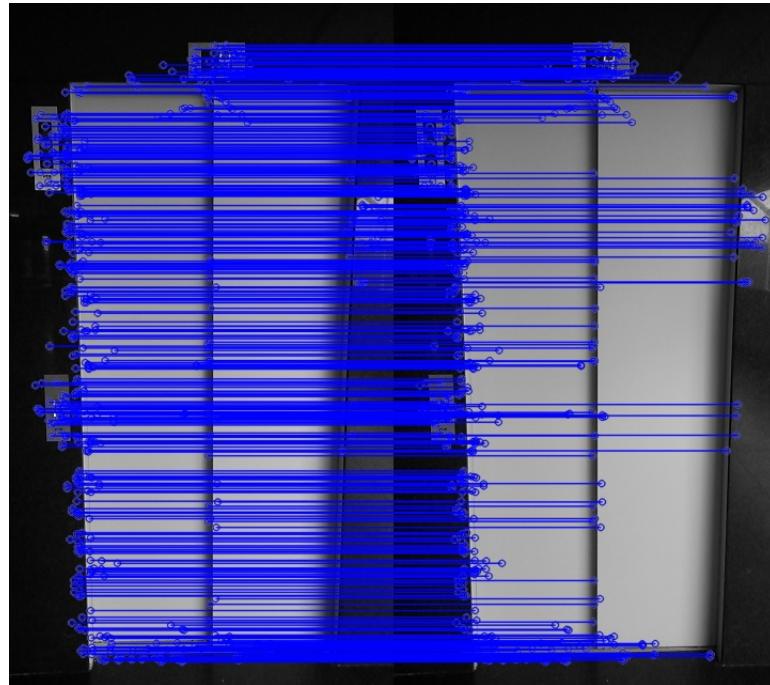


Figura 5.2: Caso de teste 1 com elevador. Imagem reconhecida corretamente com 100% de *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

Para o caso de teste 2, foto frontal um metro atrás da distância ideal, a imagem correta do banco de dados foi reconhecida com 38,65% dos *matches* criados entre as duas imagens, conforme ilustra a Figura 5.3. Na figura, à esquerda, está a imagem *query*, capturada pela aplicação móvel e à direita a imagem armazenada no banco de dados, que teve o maior número de *matches* (38,65%) e foi selecionada como sendo o cenário provável onde o usuário está.

Para uma foto frontal, com dois passos atrás da distância ideal, que é o caso de teste 3, a imagem correta do banco de dados foi reconhecida com 26,97% dos *matches* criados entre as duas imagens, conforme ilustra a Figura 5.4. Na figura, à esquerda está a imagem *query* e à direita a imagem armazenada no banco de dados.

Para o caso de teste 4, uma imagem à esquerda do local na distância ideal, a imagem correta do banco de dados foi reconhecida com 15,67% dos *matches* criados entre elas, conforme ilustra a Figura 5.5. Novamente, à esquerda está a imagem *query* e à direita a imagem armazenada no banco de dados que foi selecionada como a mais provável.

Para uma imagem à esquerda do local e com um passo atrás da distância ideal, que é o caso de teste 5, nenhuma imagem foi reconhecida. A imagem do banco de dados que

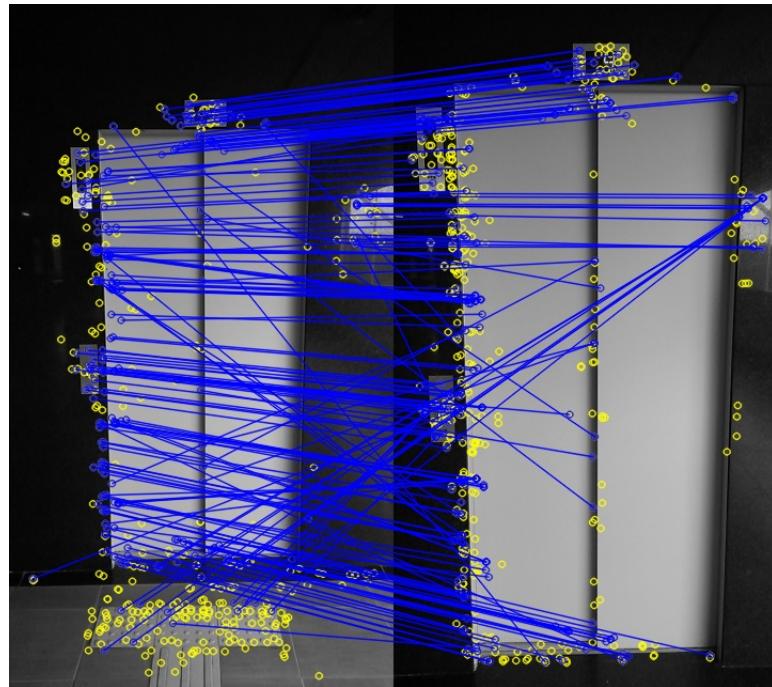


Figura 5.3: Caso de teste 2 com elevador. Imagem reconhecida corretamente com 38,65% dos *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

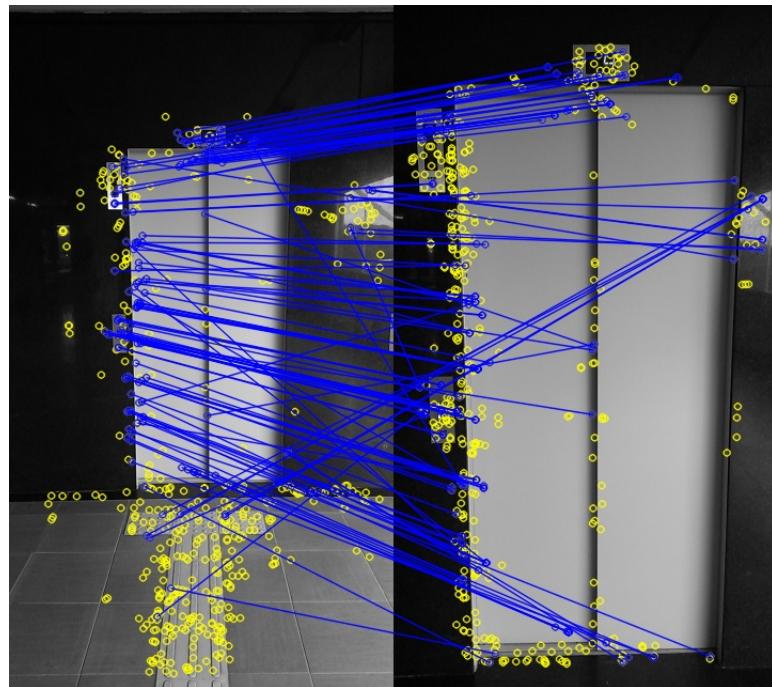


Figura 5.4: Caso de teste 3 com elevador. Imagem reconhecida corretamente com 26,97% dos *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

deveria ter sido selecionada obteve 8.74% dos matches. Esse valor está abaixo do valor mínimo de reconhecimento que é 10%, por isso não houve imagem selecionada do banco, conforme

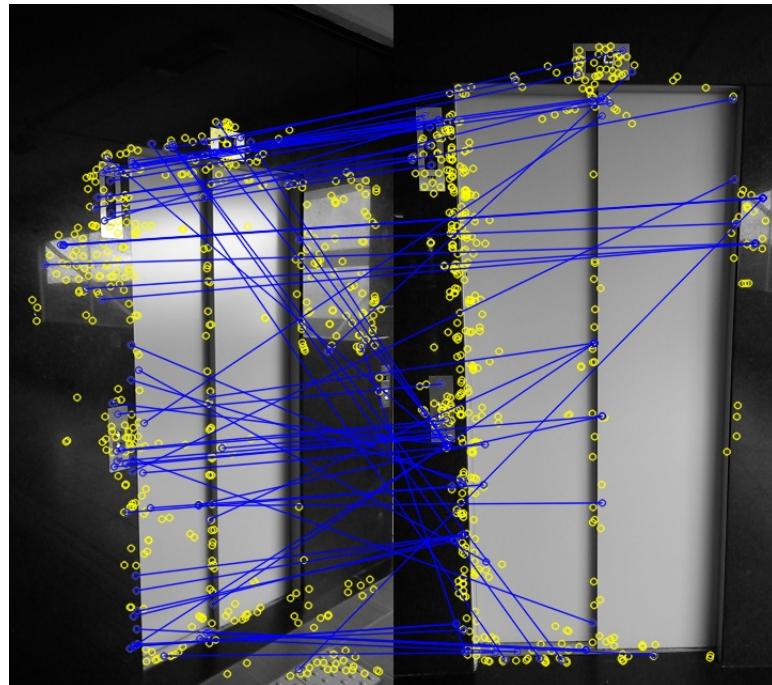


Figura 5.5: Caso de teste 4 com elevador. Imagem reconhecida corretamente com 15,67% dos *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados

explicado na Seção 4.3.3. A Figura 5.6 mostra a imagem que deveria ter sido escolhida pela aplicação.

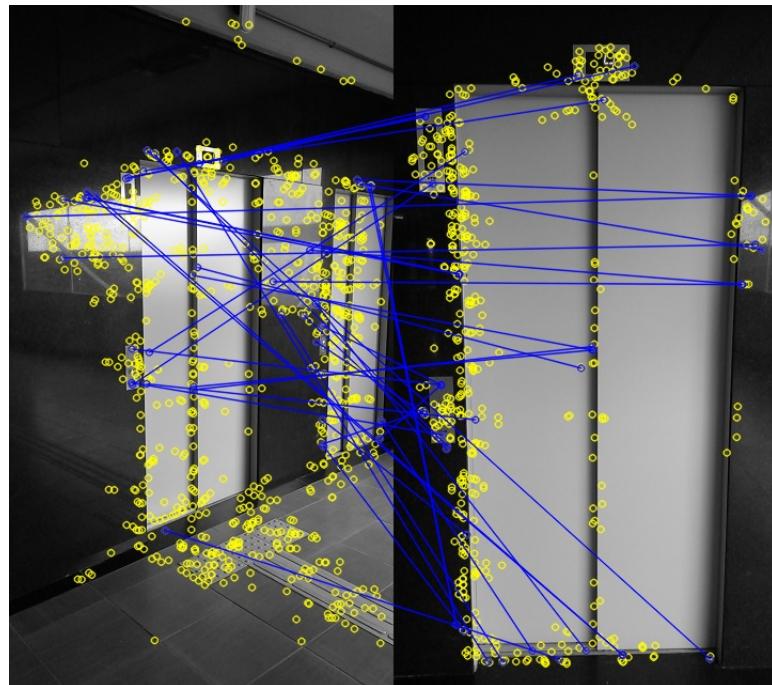


Figura 5.6: Caso de teste 5 com elevador. Nenhuma imagem reconhecida. À esquerda está a imagem da aplicação e à direita a imagem que deveria ser selecionada no banco de dados, com 8,74% de *matches*

Para uma imagem à esquerda do local ideal e com dois passos atrás da distância ideal, caso de teste 6, nenhuma imagem foi reconhecida. Novamente, a imagem do banco de dados que deveria ter sido selecionada obteve 7,54% dos *matches*, abaixo do valor mínimo, conforme ilustra a Figura 5.7.

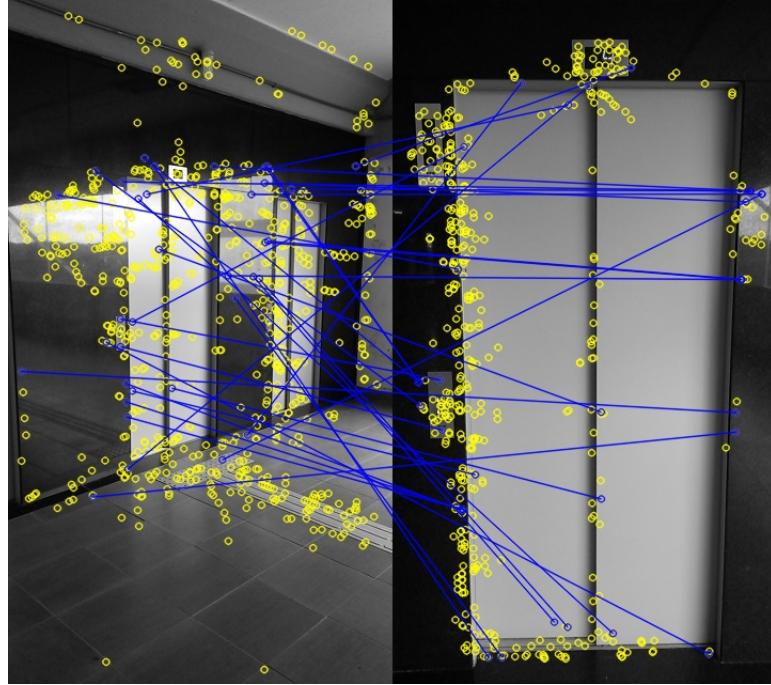


Figura 5.7: Caso de teste 6 com elevador. Nenhuma imagem reconhecida. À esquerda está a imagem da aplicação e à direita a imagem que deveria ser selecionada no banco de dados, com 7,54% de *matches*

Para uma imagem à direita do local ideal, caso de teste 7 com elevador, a imagem correta do banco de dados foi reconhecida com 41,92% dos *matches* criados entre as duas imagens, conforme ilustra a Figura 5.8. A figura à esquerda está a imagem *query* e à direita a imagem armazenada no banco de dados.

Para o caso de teste 8, foto capturada à direita do local e com um metro atrás da distância ideal, a imagem correta do banco de dados foi reconhecida com 31,28% dos *matches* criados entre as duas imagens, conforme ilustra a Figura 5.9. Para uma imagem à direita do local ideal e com dois passos atrás da distância ideal, a imagem correta do banco de dados foi reconhecida com 37,83% dos *matches* criados entre as duas imagens, conforme ilustra a Figura 5.10.

Através dos resultados encontrados no caso de teste do elevador, obtêm-se uma taxa de reconhecimento satisfatória. O resultado esperado pelo usuário foi encontrado em sete de nove diferentes testes realizados com o elevador como alvo, sendo que o usuário recebeu o retorno correto da aplicação em 78% dos casos, conforme ilustra o gráfico da Figura 5.11.

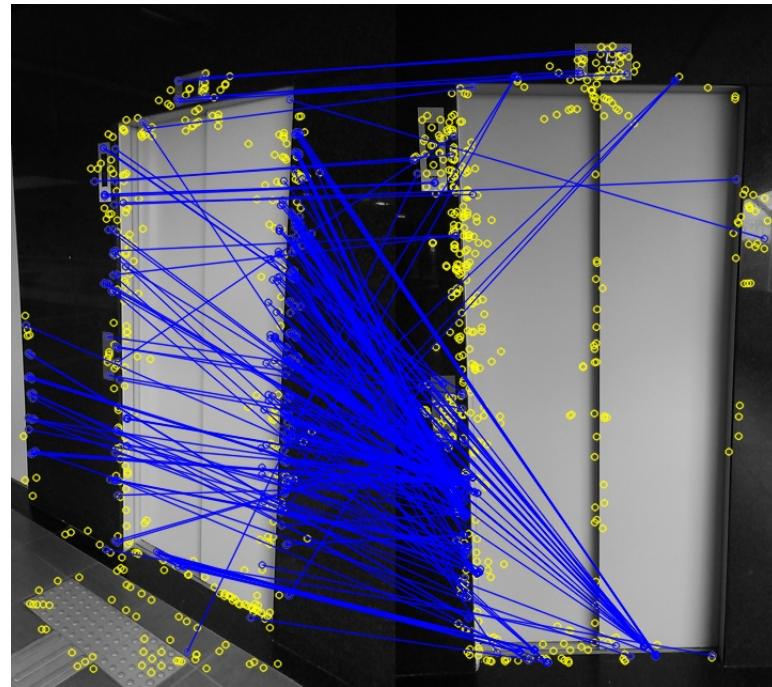


Figura 5.8: Caso de teste 7 com elevador. Imagem reconhecida corretamente com 41,92% dos *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

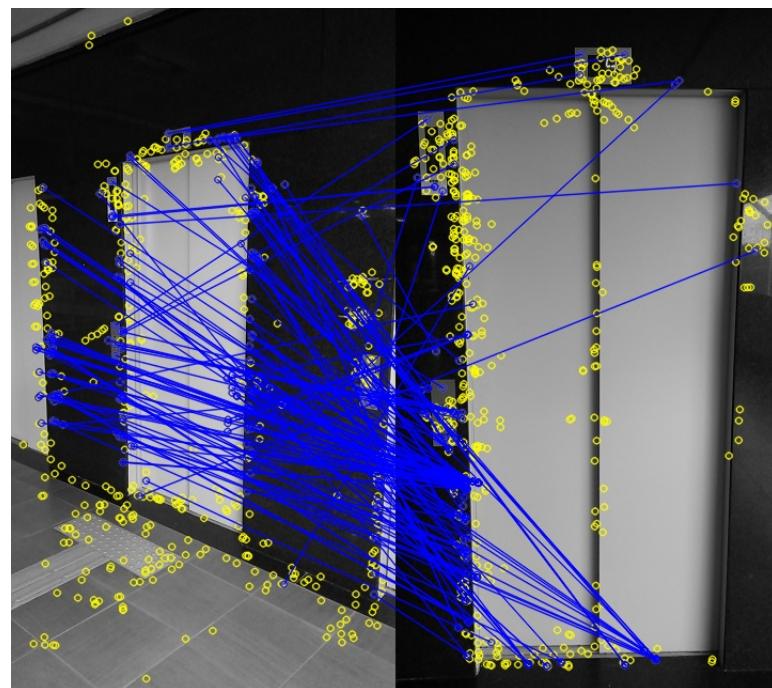


Figura 5.9: Caso de teste 8 com elevador. Imagem reconhecida corretamente com 31,28% dos *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

A Figura 5.12 apresenta um gráfico com todos os resultados obtidos nos testes realizados com o elevador. O eixo x representa os casos de teste. O eixo y a porcentagem de *matches* entre

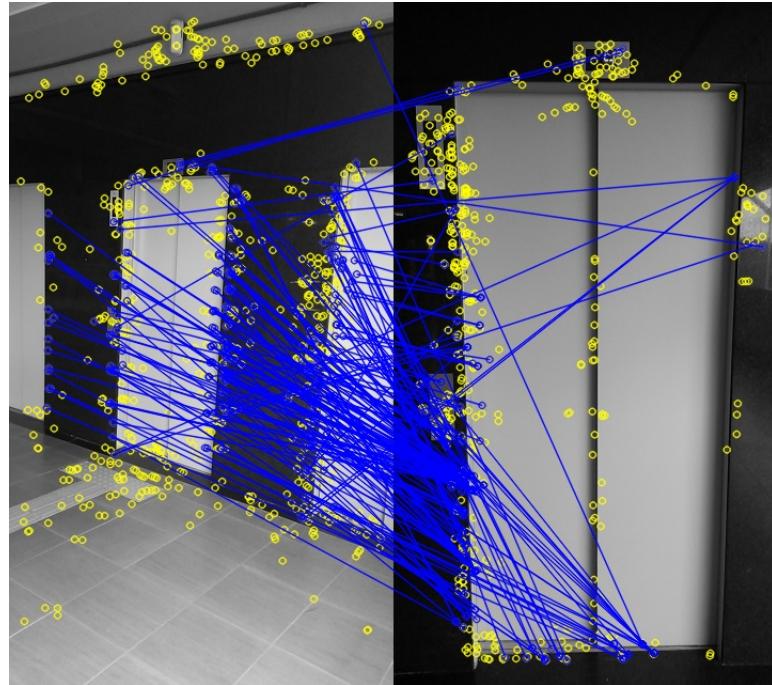


Figura 5.10: Caso de teste 9 com elevador. Imagem reconhecida corretamente com 37,83% dos *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

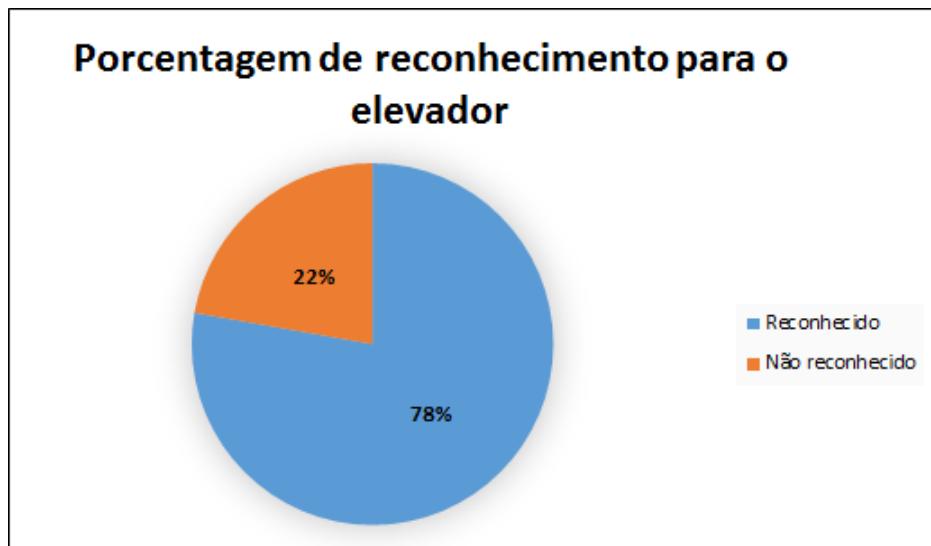


Figura 5.11: Porcentagem de reconhecimento dos testes realizados, utilizando o elevador como alvo

a imagem capturada e a imagem mais provável do banco de dados. A linha representa a relação entre estes dados.

De acordo com os dados apresentados no gráfico da Figura 5.12, percebe-se que com a imagem frontal na distância ideal o resultado é 100% de *matches* entre a imagem capturada e a escolhida no banco. Conforme a distância aumenta, o resultado de *matches* criados diminui. Do

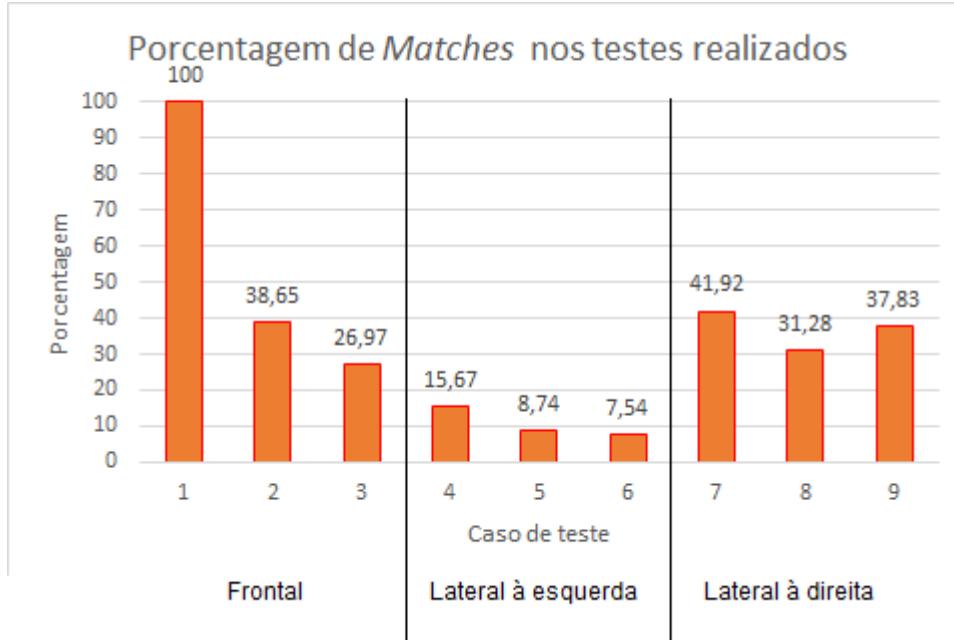


Figura 5.12: Resultados obtidos nos testes realizados com o elevador

mesmo modo ocorre com as fotos à esquerda e à direita do ângulo ideal. Nos testes realizados com o elevador, os casos de teste 3, 4 e 5, que representam fotos à esquerda do ângulo ideal, foram os casos em que obteve-se o menor número de *matches*.

5.1.2 Teste para o reconhecimento do extintor

No reconhecimento do extintor, no caso de teste 1, a imagem do banco de dados foi reconhecida corretamente com 100% dos *matches* criados entre elas, conforme ilustra a Figura 5.13. Neste caso, as demais imagens do banco de dados que foram analisadas ficaram com 0% de *matches* e o resultado encaminhado para o usuário foi o esperado.

Para uma imagem de frente, com um passo atrás da distância ideal, caso de teste 2 com o extintor, a imagem correta do banco de dados foi reconhecida com 26,61% dos *matches*, conforme ilustra a Figura 5.14. Na figura à esquerda está a imagem *query* e à direita a imagem armazenada no banco de dados que foi reconhecida pela aplicação.

Para uma imagem frontal, com dois passos atrás da distância ideal, caso de teste 3, a imagem correta do banco de dados foi reconhecida com 17,59% dos *matches*, conforme ilustra a Figura 5.15.

Para uma imagem à esquerda do local ideal, a imagem correta do banco de dados foi reconhecida com 23,31% dos *matches* criados entre as duas imagens, conforme ilustra a Figura 5.16. Na figura, à esquerda, está a imagem *query* e à direita a imagem armazenada no banco de

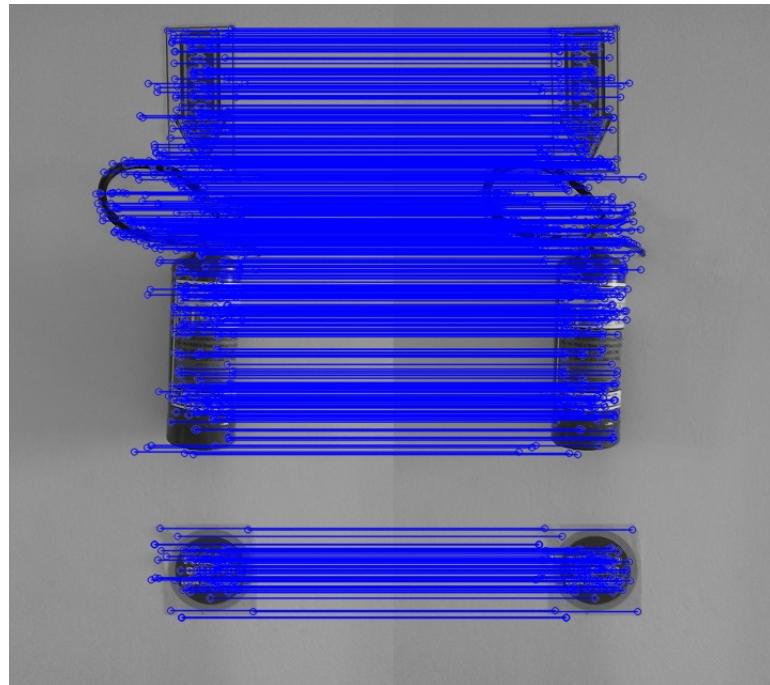


Figura 5.13: Caso de teste 1 com extintor. Imagem reconhecida corretamente com 100% de *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

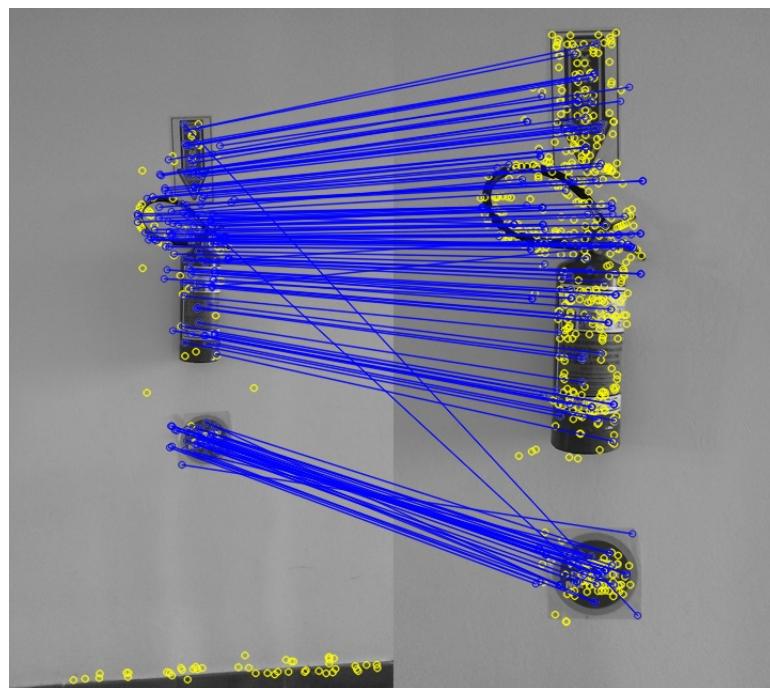


Figura 5.14: Caso de teste 2 com extintor. Imagem reconhecida corretamente com 26,61% de *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

dados.

Para o caso de teste 5 com o extintor, imagem à esquerda do local e com um passo atrás

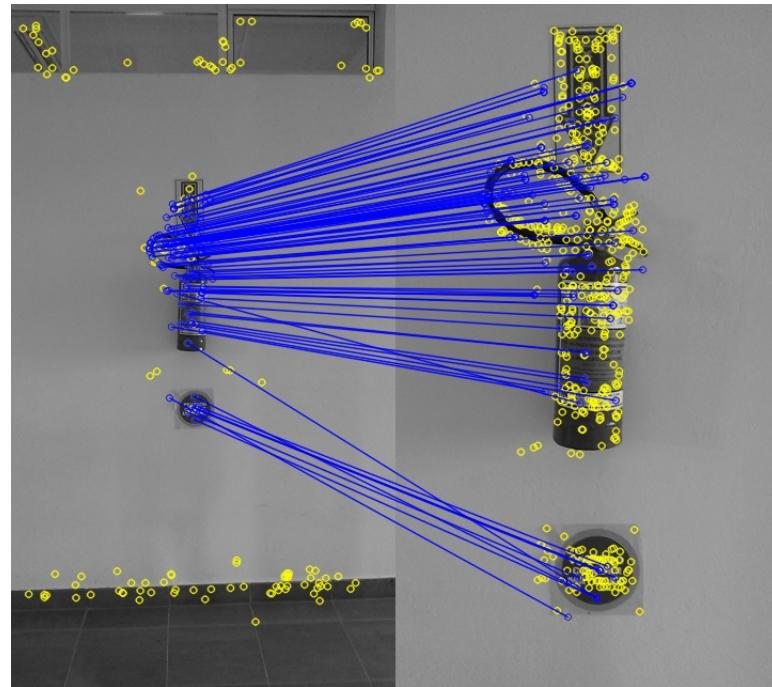


Figura 5.15: Caso de teste 3 com extintor. Imagem reconhecida corretamente com 17,59% de *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

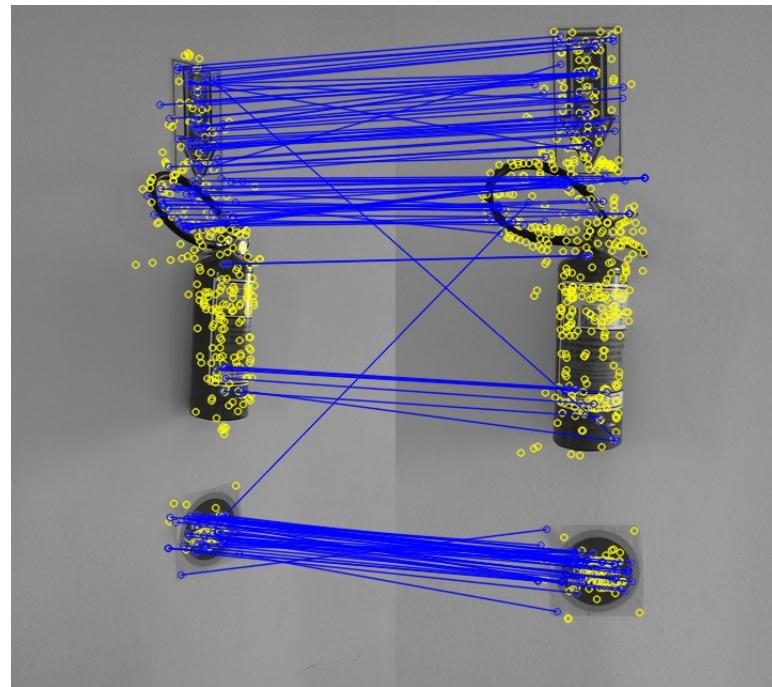


Figura 5.16: Caso de teste 4 com extintor. Imagem reconhecida corretamente com 23,31% de *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

da distância ideal, nenhuma imagem foi reconhecida. A imagem do banco de dados que deveria ter sido reconhecida obteve apenas 5,63% dos *matches*, conforme ilustra a Figura 5.17. Na

figura à esquerda está a imagem *query* e à direita a imagem armazenada no banco de dados que deveria ter sido reconhecida.

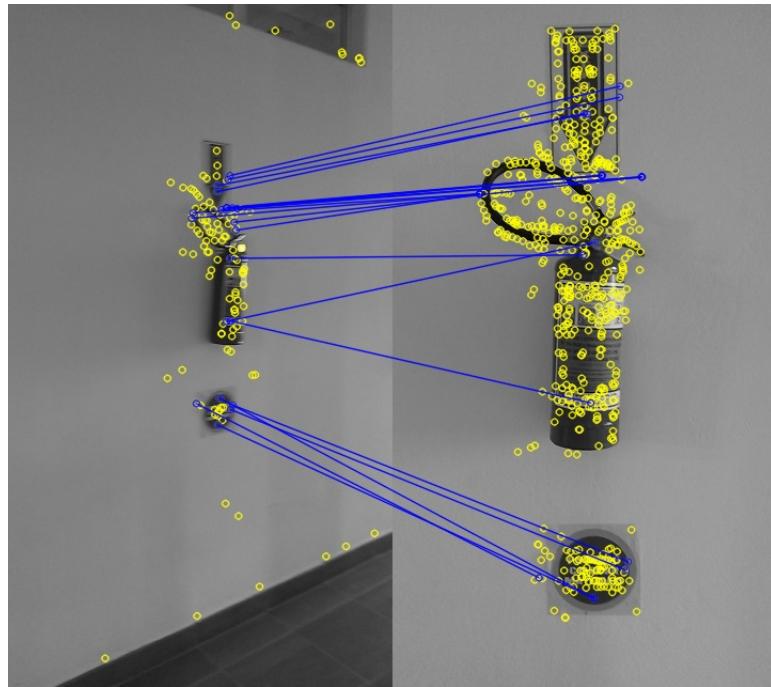


Figura 5.17: Caso de teste 5 com extintor. Nenhuma imagem reconhecida. À esquerda está a imagem da aplicação e à direita a imagem que deveria ser selecionada no banco de dados, com 5,63% de *matches*

Para uma imagem à esquerda do local ideal e com dois passos atrás da distância ideal, nenhuma imagem foi reconhecida. A imagem do banco de dados que deveria ter sido reconhecida obteve 5,15% dos *matches*, conforme ilustra a Figura 5.18.

Para o caso de teste 7 com o extintor, imagem à direita do local ideal, a imagem correta do banco de dados foi reconhecida com 20,67% dos *matches*, conforme ilustra a Figura 5.19. Na figura, à esquerda, está a imagem *query* e à direita a imagem armazenada no banco de dados que foi reconhecida. Para uma imagem à direita do local e com um passo atrás da distância ideal, nenhuma imagem foi reconhecida. A imagem do banco de dados que deveria ter sido reconhecida obteve apenas 9,31% dos *matches*, conforme ilustra a Figura 5.20. Para o caso de teste 9 com o extintor, nenhuma imagem foi reconhecida. Novamente, a imagem do banco de dados que deveria ter sido reconhecida obteve 6,22% dos *matches*, conforme ilustra a Figura 5.21. Na figura, à esquerda, está a imagem *query* e à direita a imagem armazenada no banco de dados que deveria ter sido reconhecida.

Através dos resultados obtidos nos casos de teste do extintor, obtém-se o resultado esperado pelo usuário em cinco de nove diferentes testes realizados, totalizando um retorno

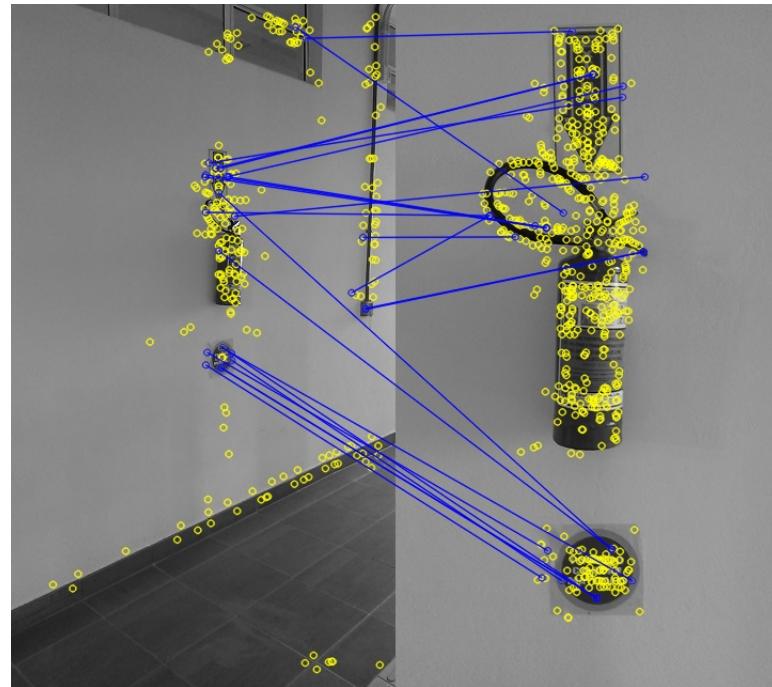


Figura 5.18: Caso de teste 6 com extintor. Nenhuma imagem reconhecida. À esquerda está a imagem da aplicação e à direita a imagem que deveria ser selecionada no banco de dados, com 5,15% de *matches*

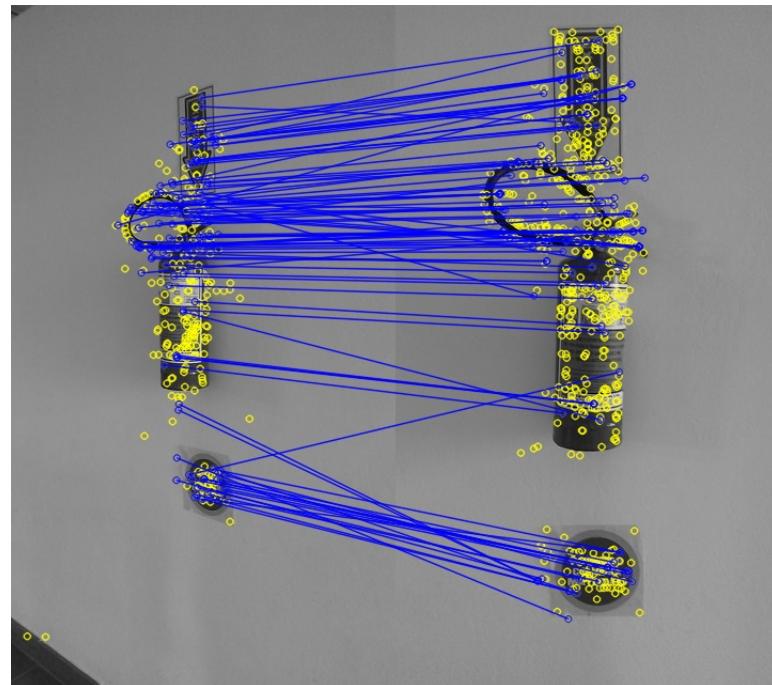


Figura 5.19: Caso de teste 7 com extintor. Imagem reconhecida corretamente com 20,67% de *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

correto da aplicação em 56% dos casos, conforme ilustra o gráfico da Figura 5.22.

A Figura 5.23 apresenta um gráfico com todos os resultados obtidos nos testes realizados

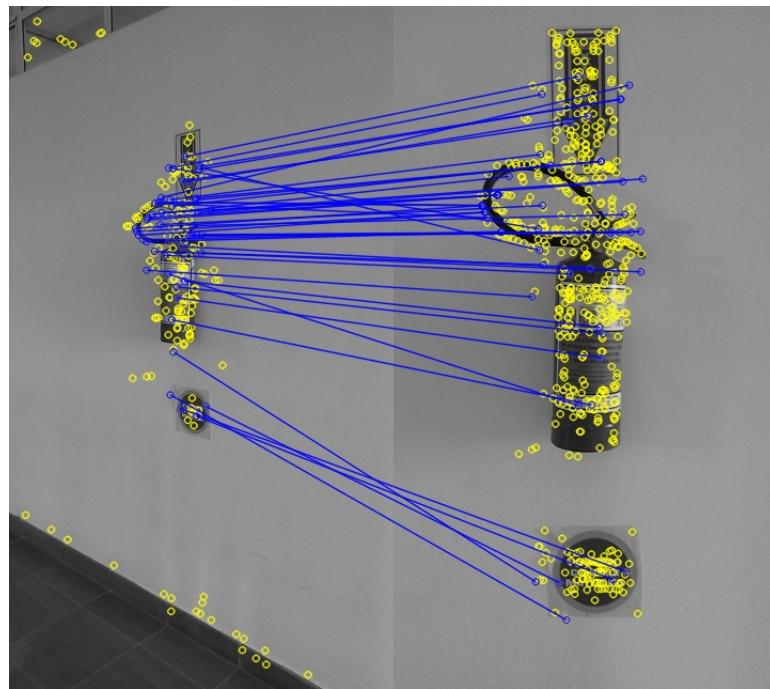


Figura 5.20: Caso de teste 8 com extintor. Nenhuma imagem reconhecida. À esquerda está a imagem da aplicação e à direita a imagem que deveria ser selecionada no banco de dados, com 9,31% de *matches*

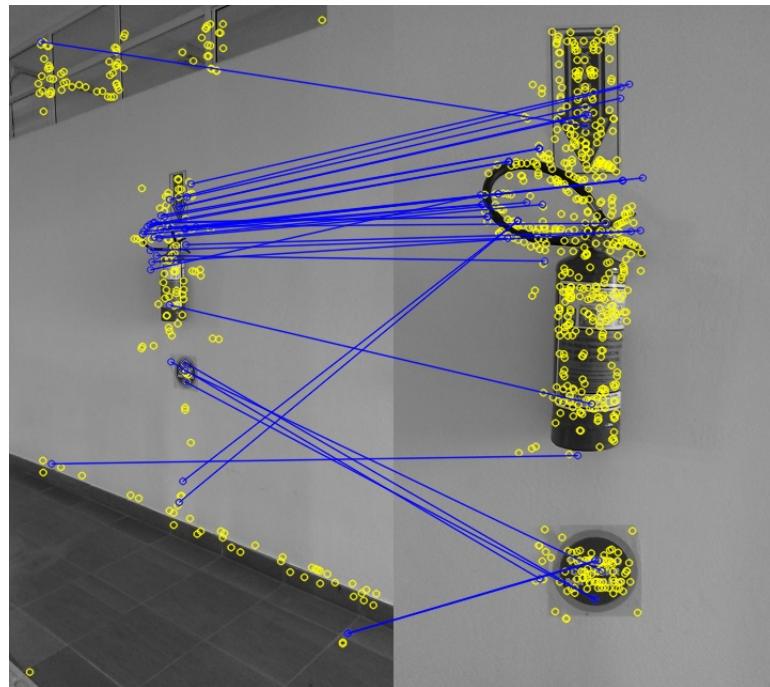


Figura 5.21: Caso de teste 9 com extintor. Nenhuma imagem reconhecida. À esquerda está a imagem da aplicação e à direita a imagem que deveria ser selecionada no banco de dados, com 6,22% de *matches*

com o extintor. O eixo x representa os casos de teste, o eixo y a porcentagem de *matches* entre a imagem capturada e a imagem mais provável do banco de dados e a linha representa a relação

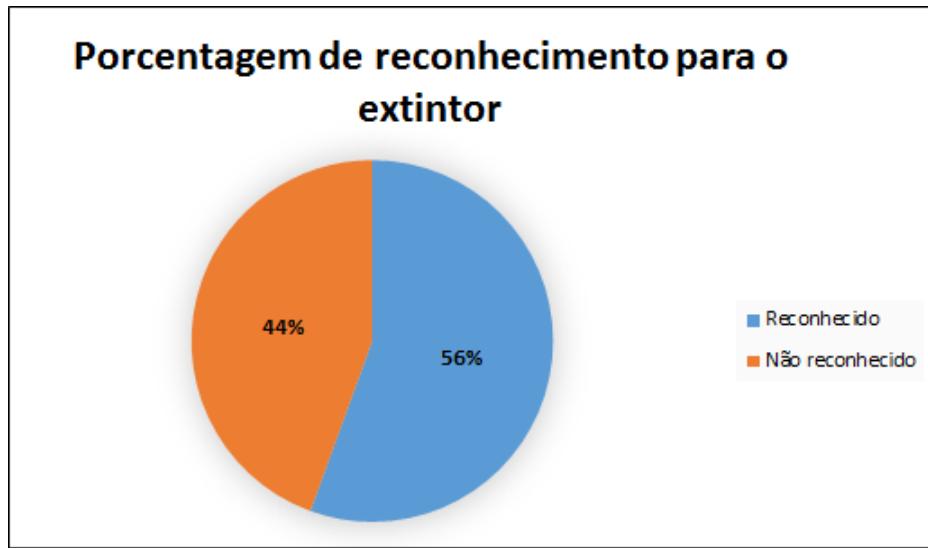


Figura 5.22: Porcentagem de reconhecimento dos testes realizados, utilizando o extintor como alvo

entre estes dados.

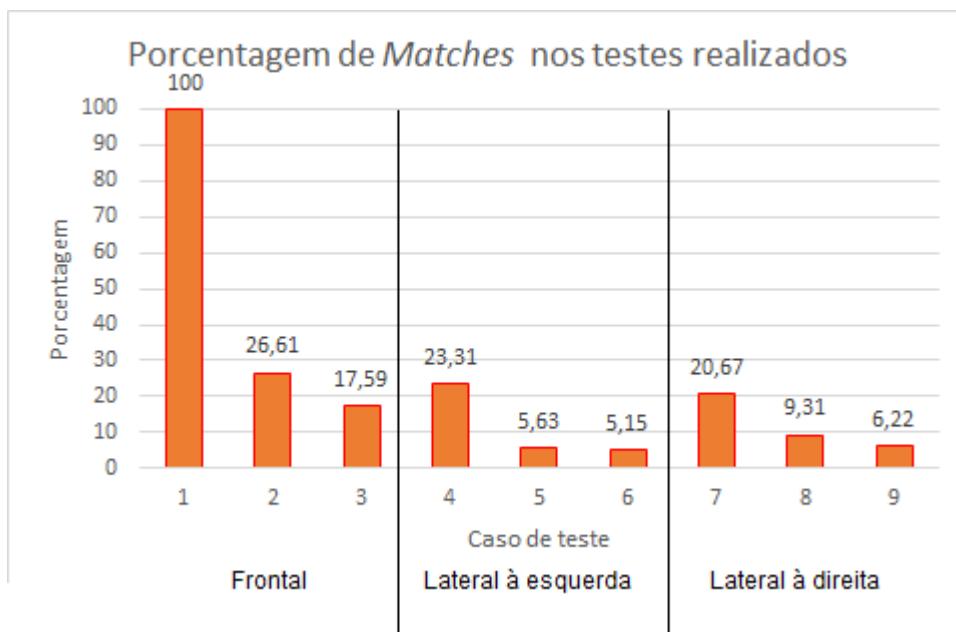


Figura 5.23: Resultados obtidos nos testes realizados com o extintor

Conforme os dados apresentados no gráfico da Figura 5.23, percebe-se novamente que com a imagem frontal na distância ideal o resultado é 100% de *matches* entre a imagem capturada e a escolhida corretamente no banco de dados. Nos casos de teste 5, 6, 8 e 9, que representam fotos à esquerda e à direita, com um e dois metros atrás da distância ideal, o resultado foi muito abaixo do esperado. Nestes quatro testes a aplicação não reconheceu o extintor.

5.1.3 Teste para o reconhecimento da porta de saída

No reconhecimento da porta de saída, no caso de teste 1, a imagem correta do banco de dados foi reconhecida com 100% dos *matches* criados entre as duas imagens, conforme ilustra a Figura 5.24. Neste caso, as demais imagens do banco de dados que foram analisadas, ficaram com 0% de *matches* e o resultado encaminhado para o usuário foi o esperado. Para uma imagem frontal, com um passo atrás da distância ideal, a imagem correta do banco de dados foi reconhecida com 42,45% de *matches*, conforme ilustra a Figura 5.25. Na figura, à esquerda, está a imagem *query* e à direita a imagem armazenada no banco de dados. Para o caso de teste 3 com a porta de saída, a imagem correta do banco de dados foi reconhecida com 26,26% dos *matches*, conforme ilustra a Figura 5.26. Na figura, à esquerda, está a imagem *query* e à direita a imagem armazenada no banco de dados.

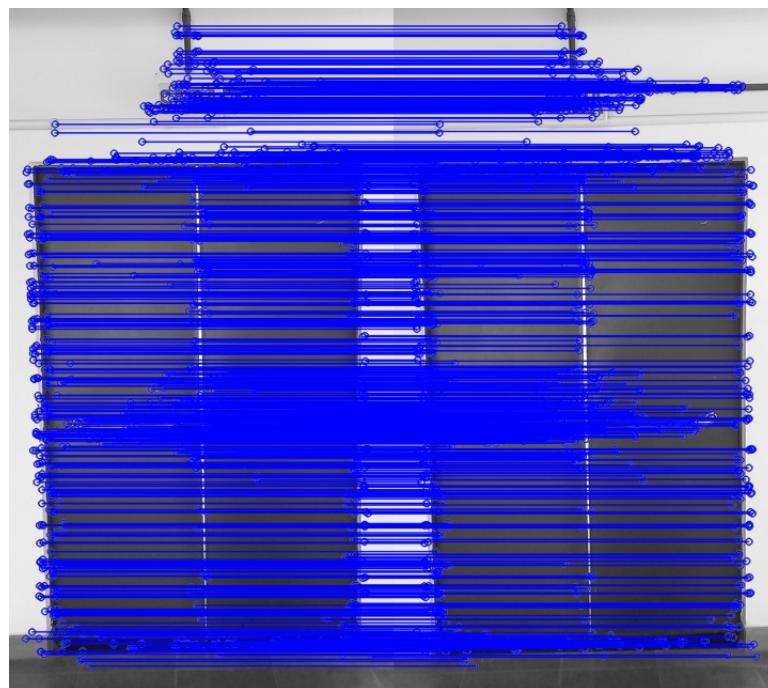


Figura 5.24: Caso de teste 1 com porta de saída. Imagem reconhecida corretamente com 100% de *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

Para uma imagem à esquerda do local ideal, a aplicação reconheceu incorretamente a imagem do banco de dados. A imagem correta ilustrada na Figura 5.27-1 possui 28,72% dos *matches* criados, enquanto a imagem incorreta, que a aplicação definiu ser a mais semelhante, possui 46,62% dos *matches* criados entre as duas imagens, conforme ilustra a Figura 5.27-2.

Para uma imagem à esquerda do local ideal e com um passo atrás da distância ideal, a

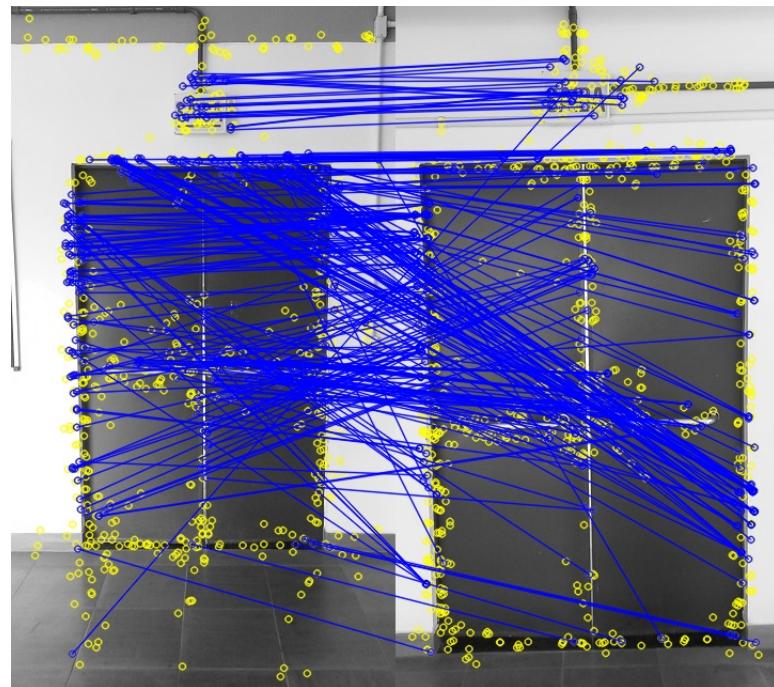


Figura 5.25: Caso de teste 2 com porta de saída. Imagem reconhecida corretamente com 42,45% de *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

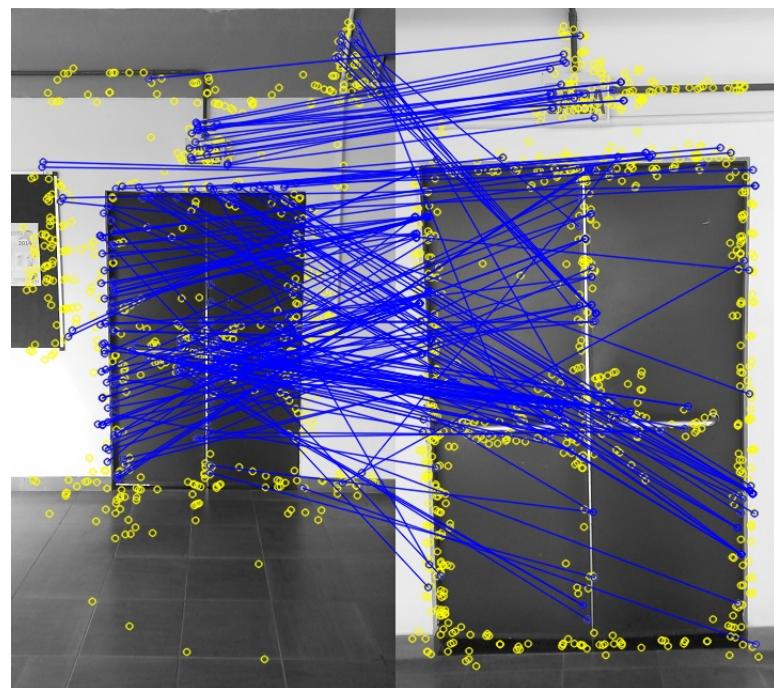


Figura 5.26: Caso de teste 3 com porta de saída. Imagem reconhecida corretamente com 26,26% de *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

aplicação reconheceu incorretamente a imagem do banco de dados. A imagem correta ilustrada na Figura 5.28-1 possui 11,08% dos *matches* criados, enquanto a imagem incorreta, que a

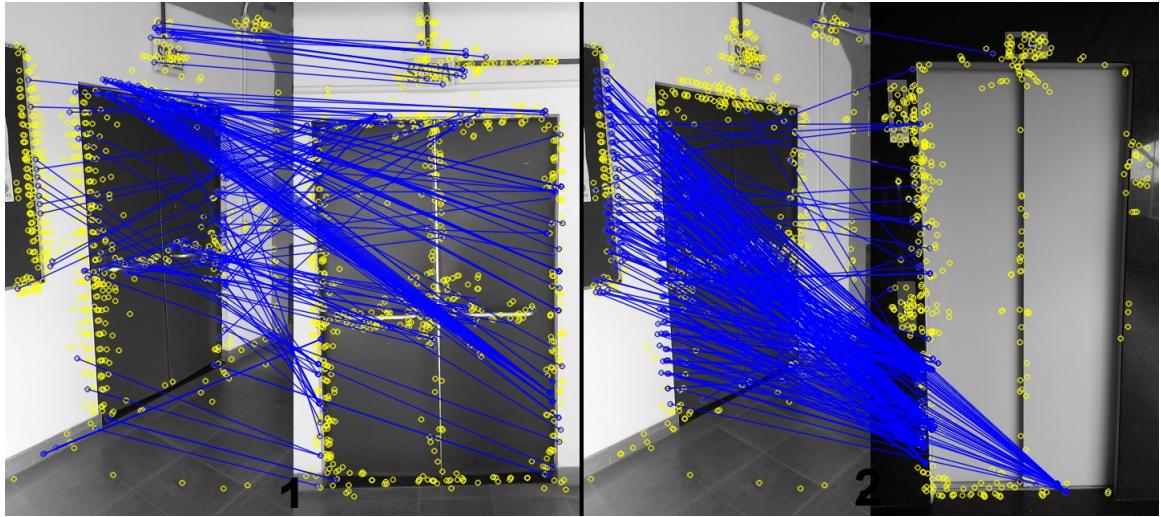


Figura 5.27: Caso de teste 4 com porta de saída, imagem reconhecida incorretamente. (1) Relação entre a imagem capturada pela aplicação e a do banco de dados que deveria ter sido selecionada, com 28,72% dos *matches*. (2) Relação entre a imagem capturada pela aplicação e a imagem escolhida incorretamente com 46,62% dos *matches*

aplicação definiu ser a mais semelhante, possui 27,81% dos *matches* criados, conforme ilustra a Figura 5.28-2.

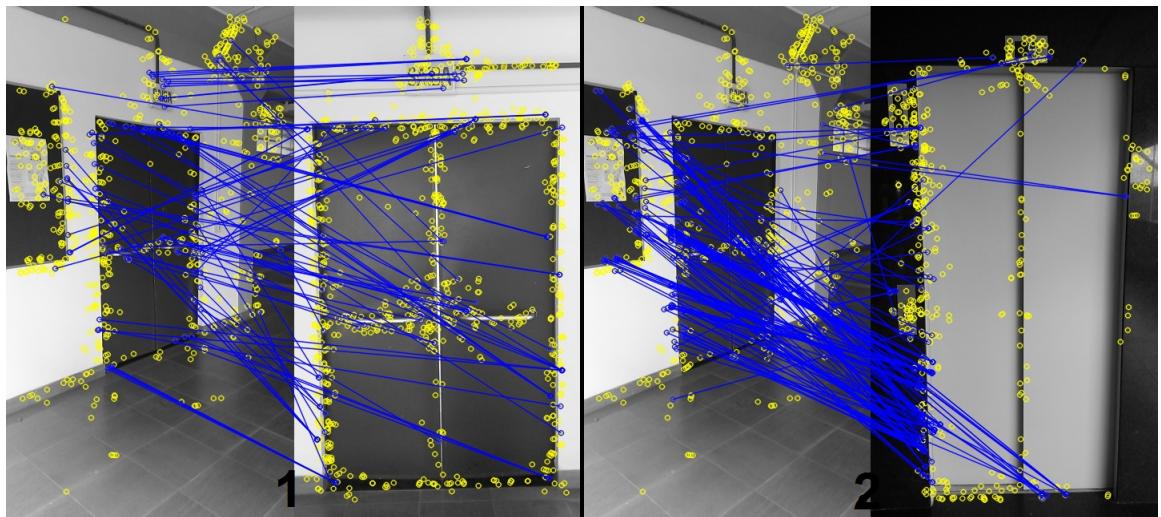


Figura 5.28: Caso de teste 5 com porta de saída, imagem reconhecida incorretamente. (1) Relação entre a imagem capturada pela aplicação e a do banco de dados que deveria ter sido selecionada, com 11,08% dos *matches*. (2) Relação entre a imagem capturada pela aplicação e a imagem escolhida incorretamente com 27,81% dos *matches*

Para uma imagem à esquerda do local ideal e com dois passos atrás da distância ideal, a aplicação reconheceu incorretamente a imagem do banco de dados. A imagem correta ilustrada na Figura 5.29-1 possui 8,64% dos *matches* criados, enquanto a imagem incorreta, que a aplicação definiu ser a mais semelhante, possui 24,94% dos *matches* criados entre as duas

imagens, conforme ilustra a Figura 5.29-2.

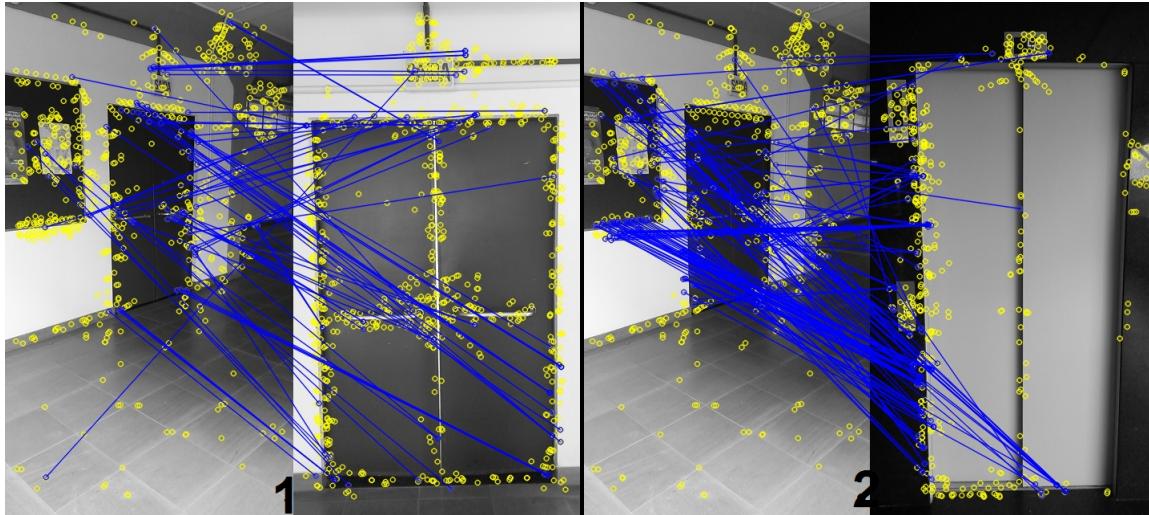


Figura 5.29: Caso de teste 6 com porta de saída, imagem reconhecida incorretamente. (1) Relação entre a imagem capturada pela aplicação e a do banco de dados que deveria ter sido selecionada, com 8,64% dos *matches*. (2) Relação entre a imagem capturada pela aplicação e a imagem escolhida incorretamente com 24,94% dos *matches*

Para uma imagem à direita do local ideal, que é o caso de teste 7, a imagem correta do banco de dados foi reconhecida com 28,81% dos *matches* criados, conforme ilustra a Figura 5.30. Na figura, à esquerda, está a imagem *query* e à direita a imagem armazenada no banco de dados. Para uma imagem à direita do local e com um passo atrás da distância ideal, a imagem correta do banco de dados foi reconhecida com 22,84% dos *matches* criados, conforme ilustra a Figura 5.31. Por fim, para o caso de teste 9 com a porta de saída, a imagem correta do banco de dados foi reconhecida com 16,72% dos *matches* criados entre as duas imagens, conforme ilustra a Figura 5.32.

Os resultados obtidos nos casos de teste da porta de saída mostraram a informação esperada pelo usuário em seis de nove diferentes testes realizados. Isso representa um retorno correto da aplicação em 67% dos casos, conforme ilustra o gráfico da Figura 5.33.

A Figura 5.34 apresenta um gráfico com todos os resultados obtidos nos testes realizados com a porta de saída, em que o eixo x representa os casos de teste, o eixo y a porcentagem de *matches* entre a imagem capturada e a imagem mais provável do banco de dados e a linha representa a relação entre estes dados.

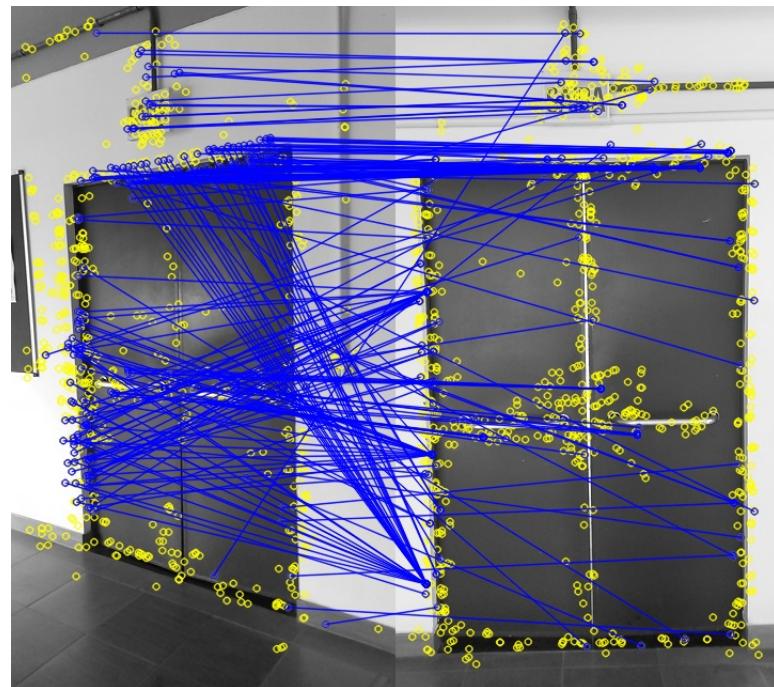


Figura 5.30: Caso de teste 7 com porta de saída. Imagem reconhecida corretamente com 28,81% de *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

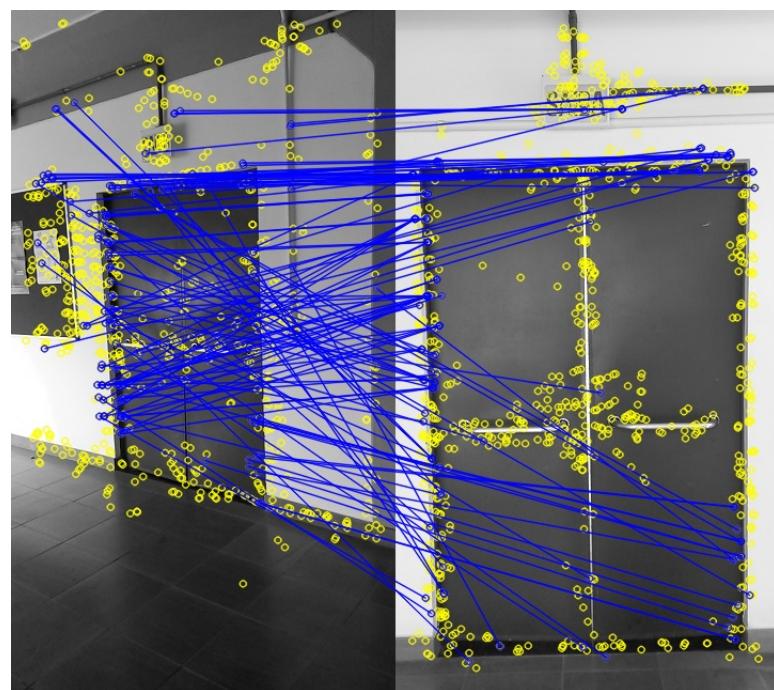


Figura 5.31: Caso de teste 8 com porta de saída. Imagem reconhecida corretamente com 22,84% de *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

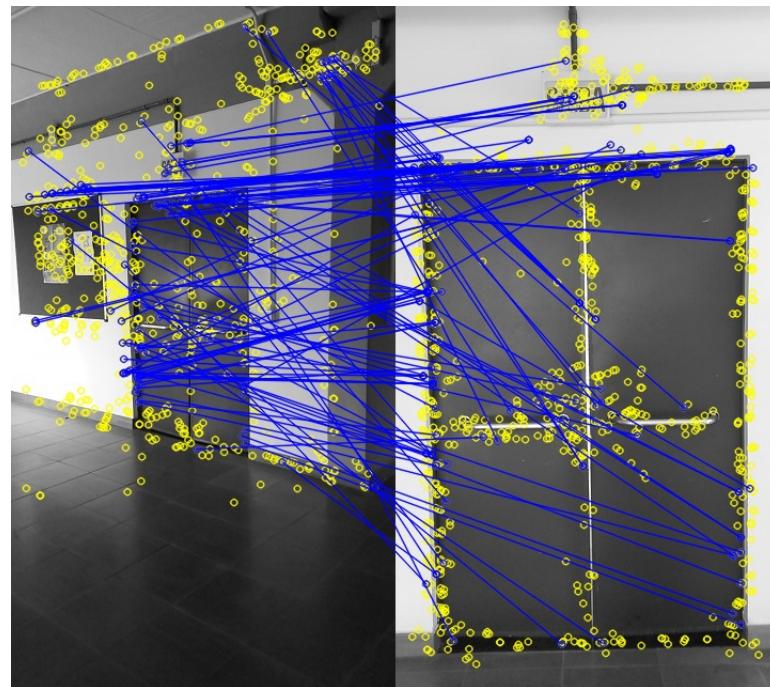


Figura 5.32: Caso de teste 9 com porta de saída. Imagem reconhecida corretamente com 16,72% de *matches*. À esquerda está a imagem da aplicação e à direita a imagem selecionada no banco de dados.

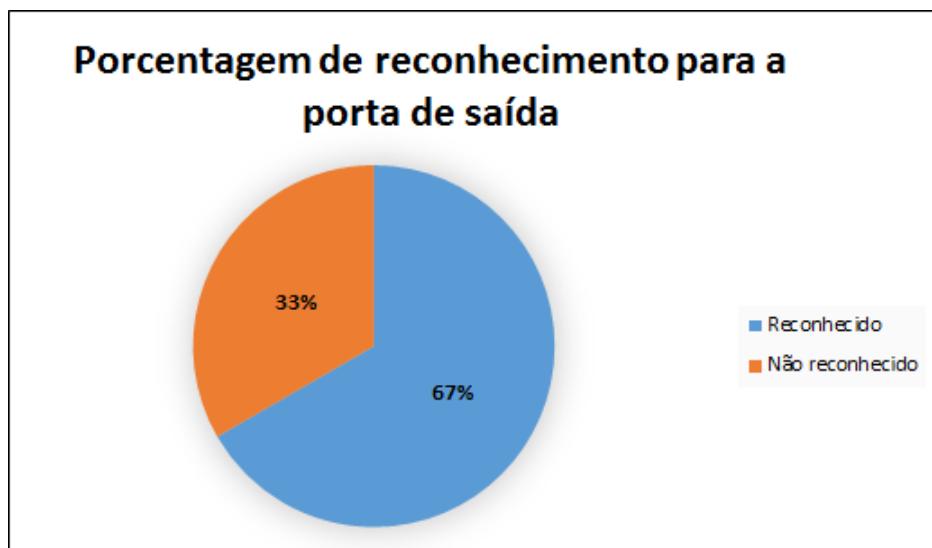


Figura 5.33: Porcentagem de reconhecimento dos testes realizados, utilizando a porta de saída como alvo

5.1.4 Considerações finais

Através dos resultados encontrados com os testes de acurácia, percebe-se que com a foto capturada a uma distância ideal, que são os casos de teste 1, 4 e 7, a aplicação respondeu corretamente em 89% dos casos, conforme ilustra o gráfico da Figura 5.35. Para esses três

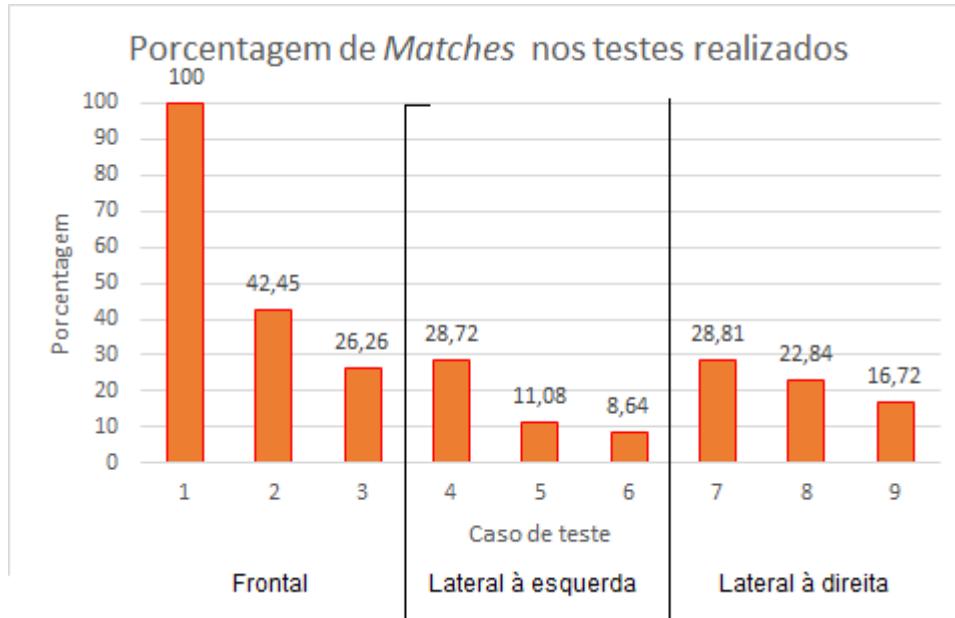


Figura 5.34: Resultados obtidos nos testes realizados com a porta de saída

casos de teste, os três alvos foram reconhecidos corretamente e a aplicação retornou ao usuário a resposta esperada.



Figura 5.35: Porcentagem de reconhecimento dos testes realizados com a foto capturada na distância ideal

Nos testes realizados com fotos capturadas a um metro atrás da distância ideal, que são os casos de teste 2, 5 e 8, a aplicação respondeu corretamente ao usuário em 56% dos casos e incorretamente nos outros 44%, conforme ilustra o gráfico da Figura 5.36. O maior problema encontrado foi para as fotos capturadas à um metro atrás da distância ideal e à esquerda do alvo, na qual para os três alvos a aplicação não reconheceu imagem alguma. Possivelmente, as fotos capturadas à esquerda do alvo ocultaram pontos chaves importantes para o reconhecimento.

Para os testes realizados com fotos capturadas dois metros atrás da distância ideal,

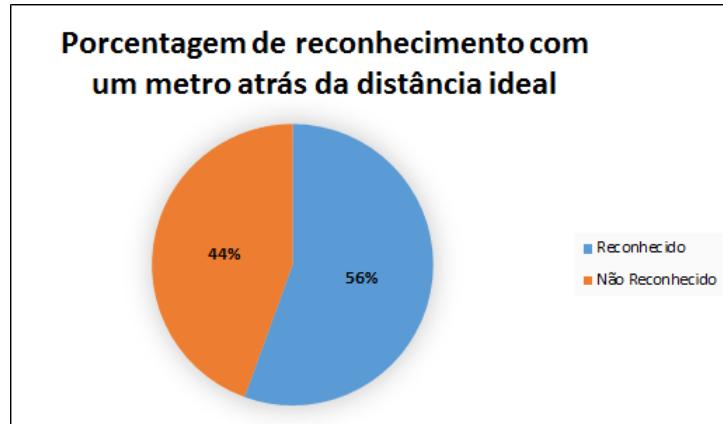


Figura 5.36: Porcentagem de reconhecimento dos testes realizados com a foto capturada um metro atrás da distância ideal

que são os casos de teste 3, 6 e 9, a aplicação respondeu corretamente ao usuário em 56% dos casos de teste e incorretamente nos outros 44%, conforme ilustra o gráfico da Figura 5.37. Nestes testes, novamente o principal problema de reconhecimento foi para as fotos à esquerda do alvo. Para os três alvos a aplicação não reconheceu imagem alguma. Novamente, as imagens capturadas à esquerda do alvo podem ter ocultado pontos chaves importantes para o reconhecimento. Outro fator que provavelmente acarretou os resultados incorretos é a semelhança das imagens de treinamento, onde a similaridade entre a porta de saída e o elevador gerou três casos de falso positivos.

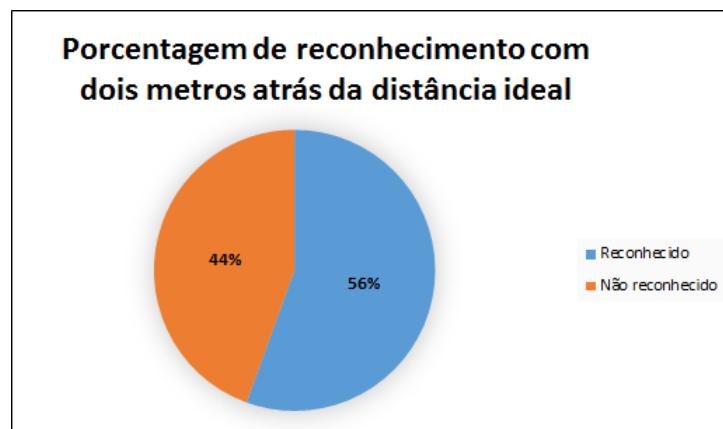


Figura 5.37: Porcentagem de reconhecimento dos testes realizados com a foto capturada dois metros atrás da distância ideal

Os testes realizados tiveram como objetivo testar a acurácia da aplicação. Apesar da aplicação ser baseada na utilização de distância e ângulo ideais, os testes mostraram os possíveis problemas que o usuário pode se deparar ao utilizar a aplicação. Possivelmente o usuário não conseguirá capturar uma foto na distância ou ângulo ideal em relação ao alvo que

deseja reconhecer. Ao realizar os testes simulando condições reais de utilização, nos quais a pessoa captura uma foto de lado ou mais longe que o ideal, pode-se concluir que a abordagem implementada é capaz de reconhecer os alvos. Isso reforça a utilidade da aplicação e viabilidade do presente trabalho.

A tabela 5.2 sintetiza os testes de acurácia realizados com todos os alvos, apresentando o alvo, a distância da foto capturada, o ângulo da foto capturada, o caso de teste, o percentual de *matches* e o resultado da aplicação para aquele caso.

Alvo	Distância	Ângulo	Caso de Teste	Matches	Resultado
Elevador	Ideal	Frontal	1	100%	Sim
Elevador	1 metro atrás da ideal	Frontal	2	38,65%	Sim
Elevador	2 metros atrás da ideal	Frontal	3	26,97%	Sim
Elevador	Ideal	À esquerda	4	15,67%	Sim
Elevador	1 metro atrás da ideal	À esquerda	5	8,74%	Não
Elevador	2 metros atrás da ideal	À esquerda	6	7,54%	Não
Elevador	Ideal	À direita	7	41,92%	Sim
Elevador	1 metro atrás da ideal	À direita	8	31,28%	Sim
Elevador	2 metros atrás da ideal	À direita	9	37,83%	Sim
Extintor	Ideal	Frontal	1	100%	Sim
Extintor	1 metro atrás da ideal	Frontal	2	26,61%	Sim
Extintor	2 metros atrás da ideal	Frontal	3	17,59%	Sim
Extintor	Ideal	À esquerda	4	23,31%	Sim
Extintor	1 metro atrás da ideal	À esquerda	5	5,63%	Não
Extintor	2 metros atrás da ideal	À esquerda	6	5,15%	Não
Extintor	Ideal	À direita	7	20,67%	Sim
Extintor	1 metro atrás da ideal	À direita	8	9,31%	Não
Extintor	2 metros atrás da ideal	À direita	9	6,22%	Não
Porta de Saída	Ideal	Frontal	1	100%	Sim
Porta de Saída	1 metro atrás da ideal	Frontal	2	42,45%	Sim
Porta de Saída	2 metros atrás da ideal	Frontal	3	26,26%	Sim
Porta de Saída	Ideal	À esquerda	4	28,72%	Não
Porta de Saída	1 metro atrás da ideal	À esquerda	5	11,08%	Não
Porta de Saída	2 metros atrás da ideal	À esquerda	6	8,64%	Não
Porta de Saída	Ideal	À direita	7	28,81%	Sim
Porta de Saída	1 metro atrás da ideal	À direita	8	22,84%	Sim
Porta de Saída	2 metros atrás da ideal	À direita	9	16,72%	Sim

Tabela 5.2: Resultados do teste de acurácia

5.2 Teste de tempo de execução

A aplicação desenvolvida tem como objetivo auxiliar as pessoas com deficiência visual em sua orientação em ambientes internos. Para tal, a aplicação deve responder de acordo com entradas em prazos hábeis. Em vista disso, realizaram-se testes de tempo de execução, com o intuito de validar a aplicação em relação ao seu tempo de execução durante sua utilização.

Vários parâmetros influenciam no tempo de execução da aplicação, porém foram realizados testes diferenciados ao tamanho das imagens de treinamento e seu impacto. As imagens alteram o tempo total de execução, pois seu tamanho altera o número de descritores e pontos chaves criados para realizar os *matches*. Os testes foram realizados com as imagens de treinamento frontais a um metro atrás da posição ideal. As imagens diferem em 1280x720 *pixels*, mostrados na seção 5.2.1, além de testes com imagens de treinamento possuindo 640x360 *pixels*, apresentados na seção 5.2.2.

5.2.1 Testes realizados com imagens de treinamento com 1280x720 *pixels*

Os testes realizados utilizaram como cenário as mesmas imagens de treinamento usadas no teste de acurácia, visto na seção 5.1, porém as imagens de treino, agora, possuem 1280x720 *pixels*, ao contrário do teste anterior, que possuíam 640x360 *pixels*. A Tabela 5.3 apresenta a configuração da rede durante a realização destes testes, sendo que a velocidade de *download* estava em 2.04 Mbps, a velocidade de *upload* estava em 2.00 Mbps e o ping estava em 48.

Velocidade de <i>Download</i>	2.04 Mbps
Velocidade de <i>Upload</i>	2.00 Mbps
Ping	48

Tabela 5.3: Configuração da rede durante a realização dos testes

Os testes utilizaram o elevador, o extintor, a porta de saída e a sacada como alvos para o reconhecimento. Considera-se como tempo total o instante em que o usuário captura a foto, até o momento em que a aplicação lê em voz alta a descrição do local reconhecido. Nestes testes, os quatro alvos foram reconhecidos corretamente: o elevador foi reconhecido em 8,08 segundos, o extintor em 6,52 segundos, a porta de saída em 12,04 segundos e a sacada em 36,44 segundos, conforme ilustra o gráfico da Figura 5.38.

O tempo total de execução é composto por outros dois períodos: tempo de reconhecimento e tempo de comunicação. O tempo de reconhecimento é o tempo decorrido

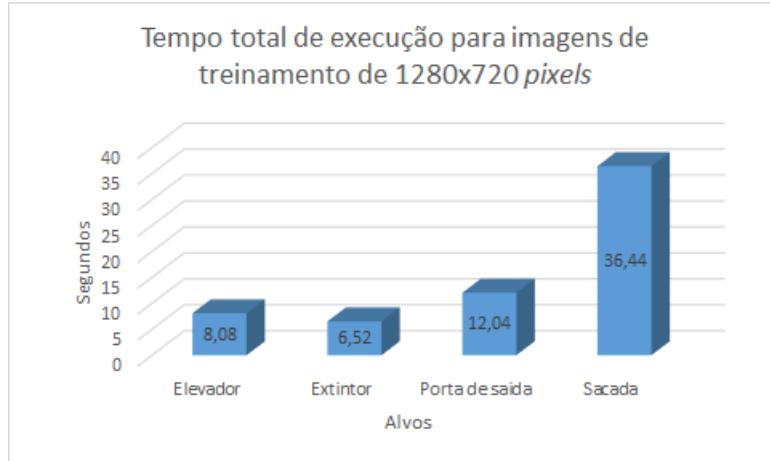


Figura 5.38: Tempo total de execução com imagens de treinamento de 1280x720 pixels

desde a criação dos *matches* entre a imagem *query* e as imagens de treinamento até a escolha da imagem mais semelhante com a *query*. O tempo de comunicação é o tempo utilizado para a comunicação da aplicação móvel com o servidor, o processamento do servidor, atualização da imagem *query* e o tempo de retorno para o usuário.

O gráfico da Figura 5.39 ilustra a divisão do tempo total de execução em tempo de reconhecimento e tempo de comunicação. O caso do elevador possui aproximadamente 5,15 segundos de reconhecimento e 2,92 segundos de comunicação, o extintor possui 3,33 segundos de reconhecimento e 3,18 segundos de comunicação, a porta de saída possui 8,73 segundos de reconhecimento e 3,30 segundos de comunicação e a sacada possui 31,63 segundos de reconhecimento e 4,80 de comunicação.

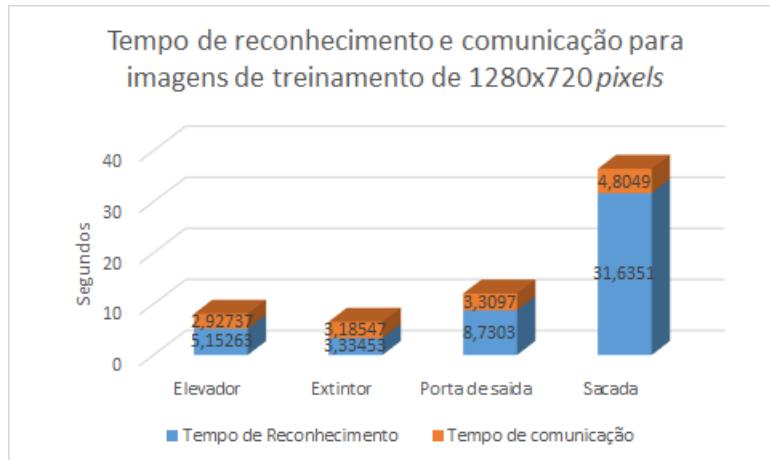


Figura 5.39: Tempo de reconhecimento e comunicação com imagens de treinamento de 1280x720 pixels

5.2.2 Testes realizados com imagens de treinamento com 640x360 pixels

Finalmente foram realizados testes utilizando como cenário as mesmas imagens de treinamento utilizadas no teste de acurácia, visto na seção 5.1, sendo que agora as imagens foram ajustadas para o tamanho 640x360pixels. A Tabela 1.1 apresenta a configuração da rede durante a realização destes testes, sendo que a velocidade de *download* estava em 1.96 Mbps, a velocidade de *upload* estava em 2.00Mbps e o ping estava em 46.

Velocidade de <i>Download</i>	1.96 Mbps
Velocidade de <i>Upload</i>	2.00 Mbps
Ping	46

Tabela 5.4: Configuração da rede durante a realização dos testes

Os testes utilizaram novamente o elevador, o extintor, a porta de saída e a sacada como alvos para o reconhecimento. O gráfico da Figura 5.40 ilustra o tempo total de execução com este cenário, sendo que considera-se como tempo total o instante em que o usuário captura a foto até que a aplicação leia em voz alta a descrição do local reconhecido. Nestes testes, os quatro alvos foram reconhecidos corretamente: o elevador foi reconhecido em 6,05 segundos, o extintor em 5,07 segundos, a porta de saída em 7,26 segundos e a sacada em 16,45 segundos, conforme ilustra o gráfico da Figura 5.38.

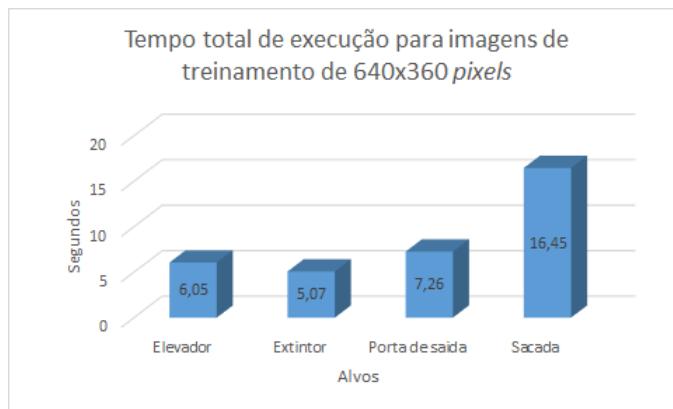


Figura 5.40: Tempo total de execução com imagens de treinamento de 640x360 pixels

Conforme descrito anteriormente, o tempo total de execução pode ser dividido em tempo de reconhecimento e tempo de comunicação. O gráfico da Figura 5.41 ilustra a divisão do tempo total de execução em tempo de reconhecimento e tempo de comunicação. O elevador possui aproximadamente 3,08 segundos de reconhecimento e 2,96 segundos de comunicação, o extintor possui 2,16 segundos de reconhecimento e 2,90 segundos de comunicação, a porta

de saída possui 3,73 segundos de reconhecimento e 3,52 segundos de comunicação e a sacada possui 13,95 segundos de reconhecimento e 2,49 de comunicação.

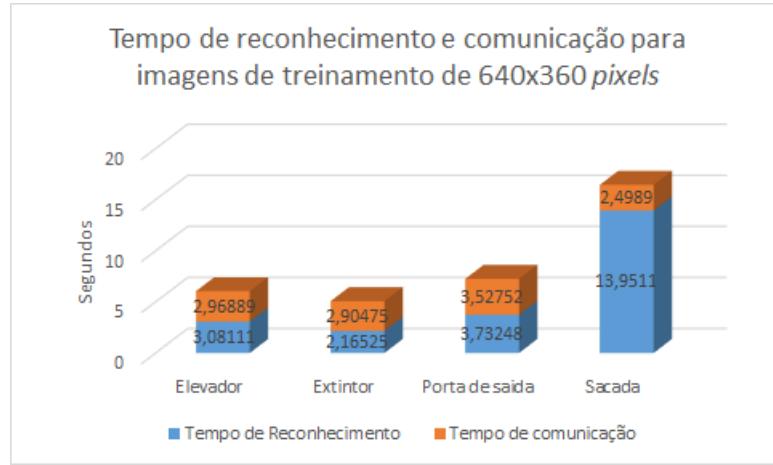


Figura 5.41: Tempo de reconhecimento e comunicação para imagens de treinamento de 640x360 pixels

Através dos testes realizados, percebe-se que a diferença de tempo entre os alvos se dá principalmente pelo número de descritores de cada imagem. Para o caso de teste utilizando a sacada como alvo, a aplicação levou cerca de 120% a mais no tempo total para as imagens com 1280x720 pixels em relação às imagens com 640x360 pixels. O tempo de comunicação também possui impacto considerável no tempo total, chegando a 57% para o caso de teste do extintor com imagens de 640x360 pixels.

5.3 Teste com a alteração do número de imagens de treinamento

Os testes realizados alternando-se o número de imagens do banco de dados tem como objetivo validar a aplicação para diferentes locais de uso, onde o número de imagens para mapear o local difere de acordo com seu tamanho ou o número de contextos que deseja-se descobrir. A Tabela 5.5 apresenta a configuração da rede durante a realização destes testes, sendo que a velocidade de *download* estava em 1.98 Mbps, a velocidade de *upload* estava em 1.95 Mbps e o ping estava em 60.

Velocidade de <i>Download</i>	1.98 Mbps
Velocidade de <i>Upload</i>	1.95 Mbps
Ping	60

Tabela 5.5: Configuração da rede durante a realização dos testes

A fim de aproximar-se às situações reais de uso, foram realizados testes alterando o

número de imagens de treinamento do banco de dados com: 40, 50, 70 e 80 imagens de treinamento. Todas as imagens possuem 1280x720 *pixels* e foi utilizado como alvo para o reconhecimento a sacada, o elevador e o extintor. Para capturar as imagens de treinamento, utilizou-se como base as dependências do prédio da Universidade Federal da Fronteira Sul (UFFS) e para completar o número de imagens de treinamento, alguns cenários tiveram mais do que uma foto descrevendo-o, conforme ilustra a Figura 5.42, onde para 80 imagens de treinamento, à esquerda é mostrada a imagem de treinamento número 5 que descreve a porta de saída e à direita a imagem de treinamento número 43 que também descreve a porta de saída. Com isso, descobriu-se que um número maior de imagens que descrevem o mesmo local aumenta a possibilidade de reconhecimento para aquele contexto.



Figura 5.42: Imagens de treinamento que descrevem a porta de saída. (1) Imagem de treinamento 5. (2) Imagem de treinamento 43.

Após os testes realizados, obteve-se um resultado satisfatório no reconhecimento em relação a acurácia. Para todos os casos, as imagens foram reconhecidas corretamente, conforme ilustra a Tabela 5.6, que mostra todos os casos de testes separados pelo alvo, pelo número de imagens de treinamento armazenadas no banco de dados e se foi reconhecido corretamente ou não.

A gráfico da Figura 5.43 apresenta a relação entre o tempo total de execução para cada

Alvo	Número de imagens de treinamento	Reconhecido
Sacada	40 imagens	Sim
Elevador	40 imagens	Sim
Extintor	40 imagens	Sim
Sacada	50 imagens	Sim
Elevador	50 imagens	Sim
Extintor	50 imagens	Sim
Sacada	70 imagens	Sim
Elevador	70 imagens	Sim
Extintor	70 imagens	Sim
Sacada	80 imagens	Sim
Elevador	80 imagens	Sim
Extintor	80 imagens	Sim

Tabela 5.6: Resultado dos testes realizados alterando o número de imagens do banco de dados

caso, de acordo com a alteração do número de imagens de treinamento armazenado no banco de dados. Na coluna y é apresentado o tempo total em segundos, na coluna x é mostrado o número de imagens de treinamento armazenados no banco de dados. A linha azul representa a relação entre estes dois parâmetros utilizando a sacada como alvo, a linha vermelha utilizando o elevador e a linha verde utilizando o extintor.

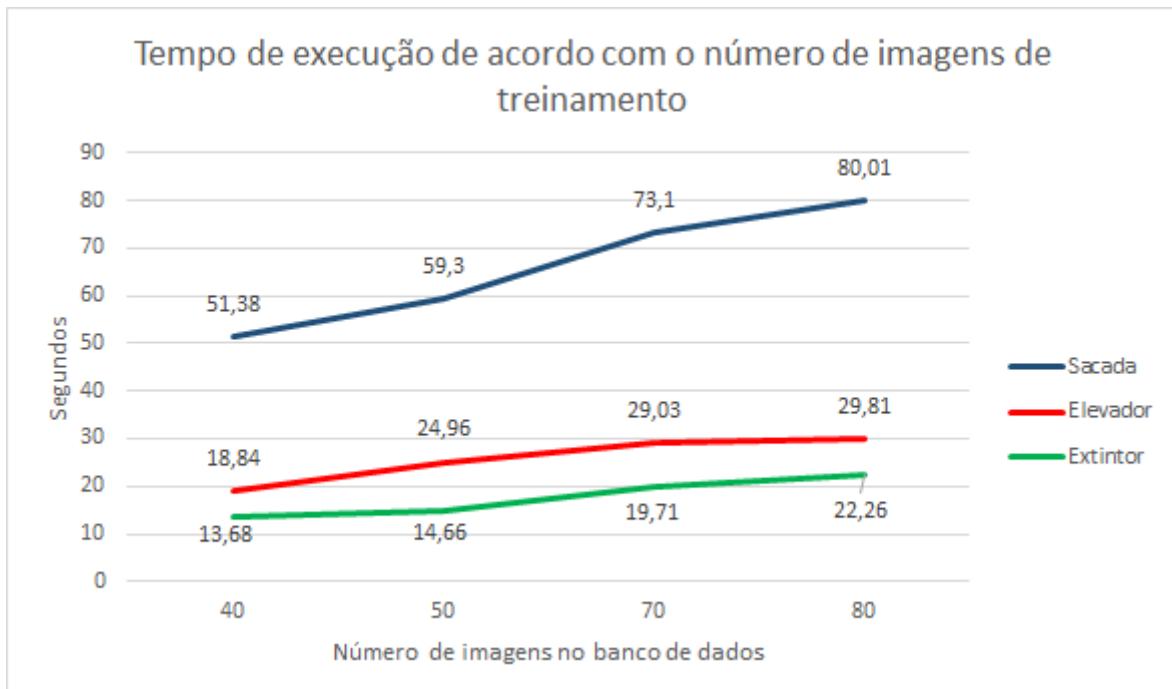


Figura 5.43: Relação entre tempo de execução e número de imagens de treinamento

Conforme ilustra o gráfico da Figura 5.43, o tempo total de execução aumenta conforme o número de imagens de treinamento aumenta, isso ocorre pelo fato de que todas as imagens

de treinamento são comparadas com a imagem *query*, portanto, quanto mais imagens de treinamento mais tempo de reconhecimento é necessário. Percebe-se que o elevador e o extintor possuem resultados aproximados, enquanto a sacada possui valores mais altos. Isso ocorre, pois a sacada possui um número maior de descritores do que os outros dois alvos.

O gráfico da Figura 5.44 ilustra a divisão do tempo total de execução em tempo de reconhecimento e tempo de comunicação, utilizando a sacada como alvo. O gráfico da Figura 5.45 apresenta o tempo de reconhecimento e de comunicação utilizando o elevador como alvo e o gráfico da Figura 5.46 ilustra a divisão do tempo em reconhecimento e comunicação, utilizando o extintor como alvo para o reconhecimento. No eixo y são mostrados os segundos de reconhecimento, no eixo x o número de imagens de treinamento no banco de dados. As barras em azul apresentam o tempo de reconhecimento de acordo com o número de imagens de treinamento e as barras em laranja apresentam o tempo de comunicação.

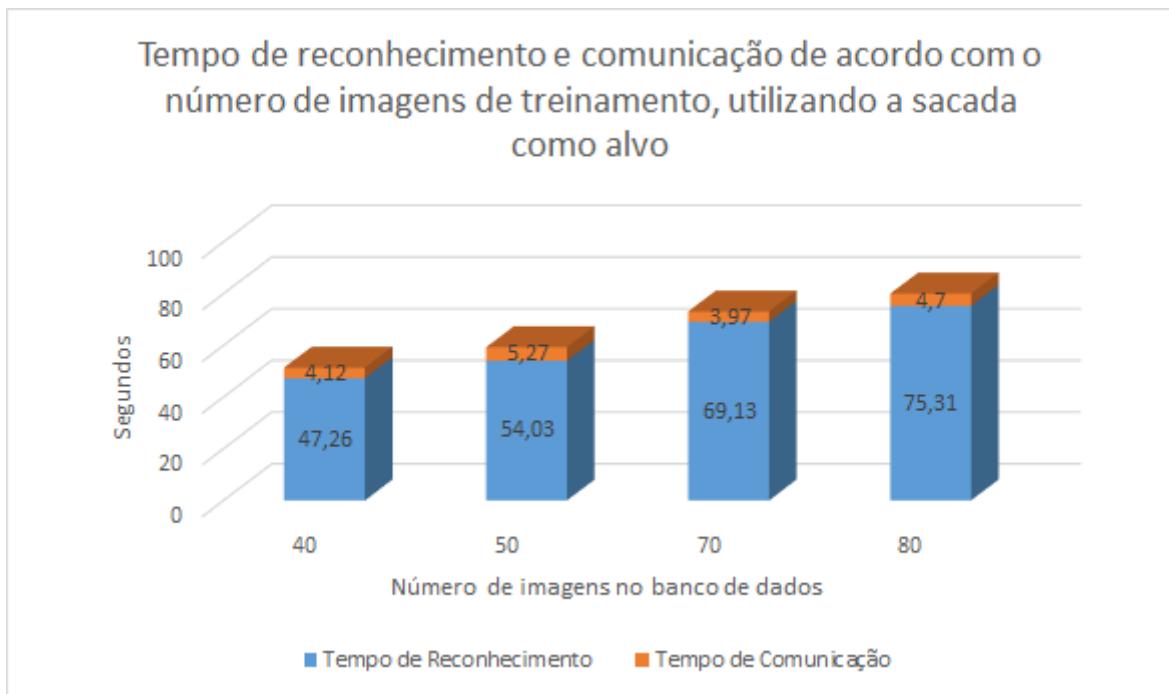


Figura 5.44: Tempo de reconhecimento e comunicação alterando o número de imagens de treinamento e utilizando a sacada como alvo

Conforme ilustram as Figuras 5.44, 5.45 e 5.46 o tempo de comunicação não sofre mudanças com a alteração no número de imagens de treinamento armazenadas no banco de dados, pois este tempo é influenciado pela rede utilizada. Percebe-se também que o tempo de reconhecimento aumenta, conforme o número de imagens de treinamento aumenta, pois faz-se a comparação de todas as imagens de treinamento com a imagem *query*.

Com os testes realizados, descobriu-se que a aplicação aumenta o tempo de resposta

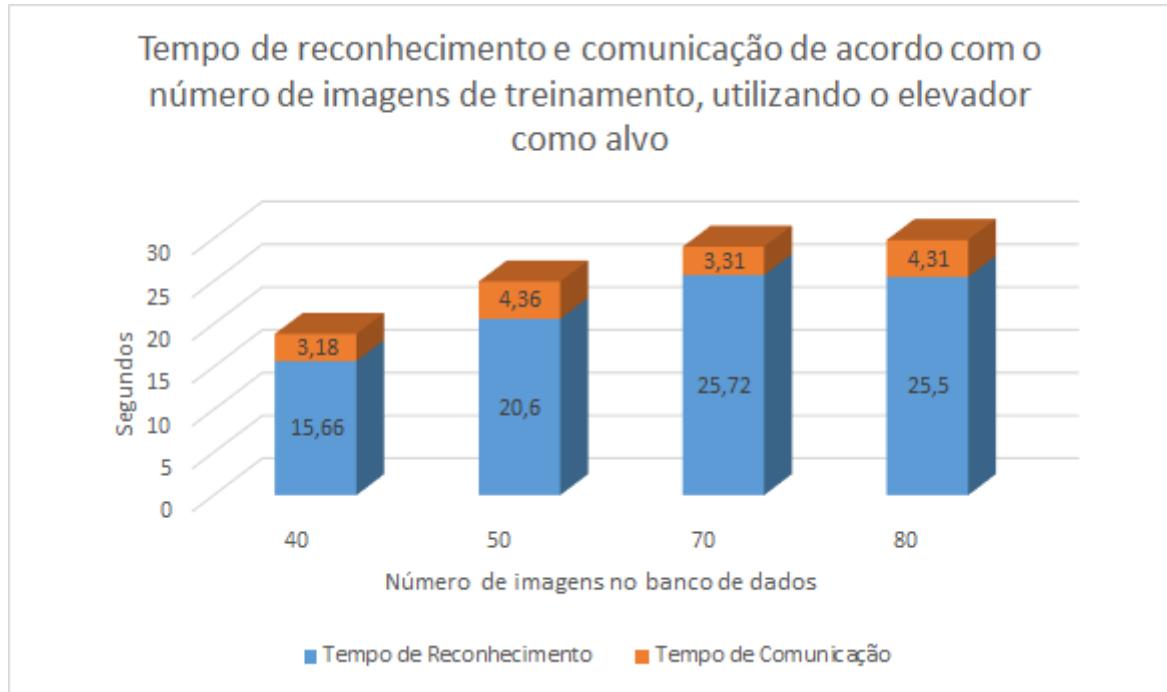


Figura 5.45: Tempo de reconhecimento e comunicação alterando o número de imagens de treinamento e utilizando o elevador como alvo

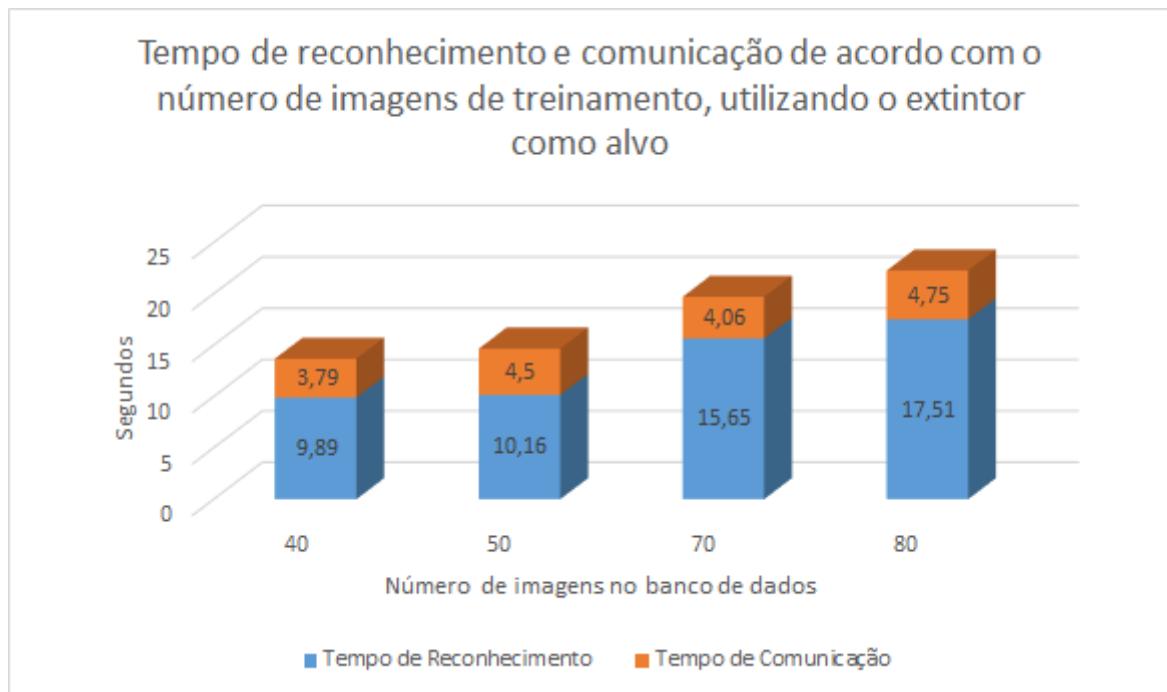


Figura 5.46: Tempo de reconhecimento e comunicação alterando o número de imagens de treinamento e utilizando o extintor como alvo

conforme aumenta o número de imagens no banco de dados, entretanto percebe-se que o tamanho das imagens tem um impacto maior no tempo de execução da aplicação, conforme apresentado na seção 5.2. Em relação a acurácia, em contra-partida, a aplicação teve um

resultado satisfatório, retornando ao usuário a descrição correta do local em todos os testes realizados.

6 CONSIDERAÇÕES FINAIS

São evidentes os desafios enfrentados pelas pessoas com deficiência visual na questão de locomoção e em diversas tarefas diárias. Neste contexto, diversas alternativas estão sendo pesquisadas para uma maior qualidade de vida destas pessoas. O presente trabalho propôs uma alternativa utilizando visão computacional para auxiliar na navegação de pessoas com deficiência visual em ambientes internos. Para tal, realizou-se uma revisão bibliográfica acerca dos conceitos utilizados na implementação do trabalho, abrangendo visão computacional e detecção e reconhecimento de imagens.

Este trabalho foi desenvolvido utilizando conceitos de visão computacional, pois identifica-se um local através da comparação de imagens. Implementou-se uma aplicação móvel, para a captura de fotos pelo usuário e uma aplicação de reconhecimento de imagens, que recebe como entrada uma imagem, seleciona em um banco de dados uma imagem mais semelhante com a entrada e gera como saída a descrição da imagem selecionada. Também foi desenvolvido um módulo do servidor, que recebe a foto capturada pelo usuário, avisa a aplicação de reconhecimento de imagens, e após o local ser escolhido, retorna ao usuário a descrição do local.

Após o desenvolvimento, realizaram-se testes para verificar a capacidade da aplicação de reconhecer locais e descrever seu contexto. Conclui-se que o algoritmo proposto funciona corretamente em várias condições, possuindo uma boa taxa de reconhecimento. Em condições ideais, o algoritmo gera a resposta esperada pelo usuário. Entretanto, alguns erros de falso positivos foram encontrados durante os testes realizados com imagens em condições não ideais.

Finalmente, pode-se perceber que a utilização de técnicas de visão computacional como alternativas para a solução de problemas encontrados no cotidiano é válida, bem como a abordagem apresentada neste trabalho.

6.1 Trabalhos futuros

No presente trabalho, todo o processamento de reconhecimento é realizado no servidor. Estudos podem ser feitos para realizar esta tarefa no próprio dispositivo. Com isso a aplicação poderia reduzir o tempo de processamento, pelo fato de não ser necessário enviar e receber dados do servidor pela rede.

Através dos resultados vistos no capítulo 5, percebe-se que esta aplicação auxilia

na orientação das pessoas com deficiência visual, porém não é capaz de identificar textos nas imagens, dificultando o reconhecimento de números das salas ou andares no prédio. Uma área que merece atenção e que pode gerar contribuições para a interação humano computador utilizando visão computacional é o reconhecimento de caracteres acoplados às demais funcionalidades da aplicação [10, 26]. Tal mudança poderia fornecer ao usuário, maiores possibilidades de reconhecimento, melhorando em sua orientação. Desta forma, é possível ao usuário reconhecer através da aplicação informações relevantes, como o número de uma sala escrita em sua porta ou o andar que se encontra no elevador.

Além do reconhecimento de textos, a aplicação não proporciona ao usuário um retorno em relação a distâncias ou dimensões. Poderia ser realizado um processamento nas imagens com relação a diferença de distância entre os objetos. Este processamento poderia identificar quando um objeto está relativamente perto ou longe do usuário, melhorando a orientação e retornando ao usuário uma melhor descrição do local.

Como contribuição futura, pode-se realizar pesquisas nos métodos e algoritmos utilizados, para estudar a possibilidade da utilização da aplicação também em ambientes externos. Isso ampliaria os objetivos do projeto atual, que possui como finalidade orientar pessoas com deficiência visual em ambientes internos. Caso necessário, novos métodos podem ser implementados, com o objetivo de garantir uma eficiência também em ambientes externos, onde o local sofre mais influências, como a claridade.

O trabalho apresentou diversos testes que validam a aplicação desenvolvida, entretanto uma maior coleta de resultados poderia ser realizada, com o intuito de descobrir possíveis falhas ou mais contribuições da aplicação. Com um número maior de testes, é possível definir por exemplo, um tamanho ideal para a imagem *query*, ou o número de imagens de treinamento armazenados no banco de dados, a ser utilizado no reconhecimento dos locais, de forma que o resultado em relação a assertividade e tempo sejam satisfatórios para o usuário.

Outra contribuição que pode ser desenvolvida, é melhorar a usabilidade do software, tanto no aplicativo móvel quanto na interface de uso do servidor, a fim de facilitar na utilização e no cadastramento de diferentes locais.

REFERÊNCIAS

- [1] K. Alsabti, S. Ranka, and V. Singh. An efficient k-means clustering algorithm. 1997.
- [2] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.
- [3] H. Bay, B. Fasel, and L. Van Gool. Interactive museum guide: Fast and robust recognition of museum objects. 2006.
- [4] J. P. Bigham, C. Jayant, H. Ji, G. Little, A. Miller, R. C. Miller, R. Miller, A. Tatarowicz, B. White, S. White, et al. Vizwiz: nearly real-time answers to visual questions. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 333–342. ACM, 2010.
- [5] A. L. H. Bischof and A. Pinz. Computer vision–eccv 2006.
- [6] Z. Z. Bittencourt and E. L. Hoehne. Qualidade de vida de deficientes visuais. *Medicina (Ribeirao Preto. Online)*, 39(2):260–264, 2006.
- [7] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua. Brief: Computing a local binary descriptor very fast. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(7):1281–1298, 2012.
- [8] J. Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):679–698, 1986.
- [9] F. d. CASTRO. Reconhecimento e localização de padrões em imagens utilizando redes neurais artificiais como estimadores de correlação espectral. *Porto Alegre: PUC-RS*, 1995.
- [10] D. Chen, J.-M. Odobez, and H. Bourlard. Text detection and recognition in images and video frames. *Pattern Recognition*, 37(3):595–608, 2004.
- [11] A. CONCI. *AZEVEDO, Eduardo. LETA, Fabiana R*, volume 407. 2008.
- [12] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pages 1–2, 2004.

- [13] E. R. Davies. *Machine vision: theory, algorithms, practicalities*. Elsevier, 2004.
- [14] N. Ezaki, M. Bulacu, and L. Schomaker. Text detection from natural scene images: towards a system for visually impaired persons. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 2, pages 683–686. IEEE, 2004.
- [15] D. A. Forsyth and J. Ponce. *Computer vision: a modern approach*. Prentice Hall Professional Technical Reference, 2002.
- [16] A. Helal, S. E. Moore, and B. Ramachandran. Drishti: An integrated navigation system for visually impaired and disabled. In *Wearable Computers, 2001. Proceedings. Fifth International Symposium on*, pages 149–156. IEEE, 2001.
- [17] M. Hersh and M. A. Johnson. *Assistive technology for visually impaired and blind people*. Springer, 2010.
- [18] I. Jacobs. Uris, addressability, and the use of http get and post. *World Wide Web Consortium. <http://www.w3.org/2001/tag/doc/whenToUseGet.html>. Version*, 2004.
- [19] B. Jähne. *Digital image processing*. 2005.
- [20] R. Kaur, M. Verma, and H. K. Kalpana. Classification of various edge detectors. *Department of Computer Science, RIEIT*, 2002.
- [21] J. M. Loomis, R. G. Golledge, and R. L. Klatzky. Navigation system for the blind: Auditory display modes and guidance. *Presence: Teleoperators and Virtual Environments*, 7(2):193–203, 1998.
- [22] D. G. Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.
- [23] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.
- [24] E. Mäkinen. *Computer vision techniques and applications in human-computer interaction*. 2004.

- [25] O. Marques Filho and H. V. Neto. *Processamento digital de imagens*. Brasport, 1999.
- [26] A. Mishra, K. Alahari, and C. Jawahar. Top-down and bottom-up cues for scene text recognition. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2687–2694. IEEE, 2012.
- [27] M. Nixon, M. S. Nixon, and A. S. Aguado. *Feature extraction & image processing for computer vision*. Academic Press, 2012.
- [28] G. L. Santos. *Localização de Robôs Móveis Autônomos Utilizando Fusão Sensorial de Odometria e Visão Monocular*. PhD thesis, Universidade Federal do Rio Grande do Norte, 2010.
- [29] M. Sharifi, M. Fathy, and M. Tayefeh Mahmoudi. A classified and comparative study of edge detection algorithms. In *Information Technology: Coding and Computing, 2002. Proceedings. International Conference on*, pages 117–120. IEEE, 2002.
- [30] W. Skarbek, A. Koschan, Z. Veröffentlichung, et al. Colour image segmentation-a survey. 1994.
- [31] A. P. Sonza and L. M. C. Santarosa. Ambientes digitais virtuais: acessibilidade aos deficientes visuais. *RENOTE*, 1(1), 2003.
- [32] R. Szeliski. *Computer vision: algorithms and applications*. Springer, 2010.
- [33] E. Tezuka. Um modelo de visão computacional para identificação do estágio de maturação e injúrias no pós-colheita de bananas. *Programa de Pós-Graduação em Ciência da Computação*, 2009.
- [34] C. Valgren and A. J. Lilienthal. Sift, surf and seasons: Long-term outdoor localization using local features. In *EMCR*, 2007.
- [35] A. White, B.; Miller. Open shades. <http://www.openshades.com/>, 2013. [Online; Acesso em: 16/06/2014].