



# Universidade de Brasília

Instituto de Ciências Exatas  
Departamento de Ciência da Computação

## Identificação e Localização de pessoas em Smartspaces

Danilo Ávila Monte Christo Ferreira  
Tales Mundim Andrade Porto

Monografia apresentada como requisito parcial  
para conclusão do Bacharelado em Ciência da Computação

Orientador  
Prof. Dr. Carla Denise Castanho

Coorientador  
Fabricio Nogueira Buzeto

Brasília  
2011

Universidade de Brasília — UnB  
Instituto de Ciências Exatas  
Departamento de Ciência da Computação  
Bacharelado em Ciência da Computação

Coordenador: Prof. Lamar

Banca examinadora composta por:

Prof. Dr. Carla Denise Castanho (Orientador) — CIC/UnB  
Prof. Dr. Professor I — CIC/UnB  
Prof. Dr. Professor II — CIC/UnB

#### **CIP — Catalogação Internacional na Publicação**

Ferreira, Danilo Ávila Monte Christo  
Porto, Tales Mundim Andrade.

Identificação e Localização de pessoas em Smartspaces / Danilo Ávila Monte Christo Ferreira, Tales Mundim Andrade Porto. Brasília : UnB, 2011.

165 p. : il. ; 29,5 cm.

Monografia (Graduação) — Universidade de Brasília, Brasília, 2011.

1. rastreamento, 2. reconhecimento facial, 3. localização,  
4. computação ubíqua

CDU 004.4

Endereço: Universidade de Brasília  
Campus Universitário Darcy Ribeiro — Asa Norte  
CEP 70910-900  
Brasília-DF — Brasil



# Universidade de Brasília

Instituto de Ciências Exatas  
Departamento de Ciência da Computação

## Identificação e Localização de pessoas em Smartspaces

Danilo Ávila Monte Christo Ferreira

Tales Mundim Andrade Porto

Monografia apresentada como requisito parcial  
para conclusão do Bacharelado em Ciência da Computação

Prof. Dr. Carla Denise Castanho (Orientador)

CIC/UnB

Prof. Dr. Professor I Prof. Dr. Professor II

CIC/UnB

CIC/UnB

Prof. Lamar

Coordenador do Bacharelado em Ciência da Computação

Brasília, 2 de maio de 2011

# Dedicatória

Dedico a....

# Agradecimentos

Agradeço a....

# Resumo

Mark Weiser introduziu o conceito de Computação Ubíqua que consiste em uma nova linha de pesquisa tecnológica caracterizada por um ambiente que interage de maneira inteligente com seus usuários e possui uma ampla e transparente interação entre dispositivos e serviços disponíveis. Observando a realidade de um ambiente como esse, fica claro que informações como posição dos usuários e suas respectivas identidades são de grande valia para tornar possível a interação entre os usuários e os recursos presentes no ambiente.

Esse trabalho propõe, então, um sistema de reconhecimento facial, rastreamento e localização de usuários em um ambiente inteligente afim de prover informações de contexto contendo as identidades e as localizações de cada usuário no ambiente. Tal sistema é chamado de Sistema TRUE (*Tracking and Recognizing Users in the Environment*) que utiliza imagens de cor e de profundidade como dados de entrada.

**Palavras-chave:** rastreamento, reconhecimento facial, localização, computação ubíqua

# Abstract

Mark Weiser introduced the concept of Ubiquitous Computing consisting of a new line of technological research characterized by an intelligent environment that acts with users and has an extensive and transparent interaction between available devices and services. Looking at the reality of such environment, it is clear that informations such as user's positions and identities are very valueable to make possible the interaction between users and resources in the environment.

Therefore, this work proposes a face recognition, tracking and localization system in order to provide context information containing the identity and postion of each user in an intelligent environment. This system is called TRUE System (Tracking and Recognizing Users in the Environment) and uses color and depth images as input data.

**Keywords:** tracking, face recognition, localization, ubiquitous computing

# Sumário

<b>1</b>	<b>Introdução</b>	<b>1</b>
1.1	Organização do trabalho . . . . .	2
<b>2</b>	<b>Fundamentação Teórica</b>	<b>3</b>
2.1	Rastreamento . . . . .	4
2.1.1	Representação da Entidade . . . . .	5
2.1.2	Seleção de características para rastreamento . . . . .	6
2.1.3	Detecção de entidades . . . . .	7
2.1.4	Rastreamento de entidades . . . . .	7
2.2	Localização . . . . .	8
2.3	Identificação . . . . .	10
2.3.1	Biometria . . . . .	11
2.3.2	Reconhecimento Facial . . . . .	14
<b>3</b>	<b>Trabalhos Correlatos</b>	<b>26</b>
3.1	Projeto CHIL . . . . .	26
3.1.1	Rastreamento de pessoas . . . . .	26
3.1.2	Identificação de pessoas . . . . .	28
3.2	UPC Smart Room . . . . .	29
3.2.1	Detecção de Movimento e Rastreamento . . . . .	30
3.2.2	Detecção e Reconhecimento Facial . . . . .	31
3.3	Projeto DIVA . . . . .	32
3.3.1	Rastreamento e Reconhecimento facial . . . . .	33
<b>4</b>	<b>Sistema TRUE</b>	<b>35</b>
4.1	Middleware <i>UbiquitOS</i> . . . . .	36
4.2	Módulo de Reconhecimento . . . . .	36
4.2.1	Pré-processamento e Processamento da Imagem . . . . .	38
4.2.2	Detecção Facial . . . . .	38
4.2.3	Reconhecimento Facial com <i>Eigenfaces</i> . . . . .	39
4.3	Módulo de Rastreamento . . . . .	41
4.4	Relação Rastreamento e Reconhecimento . . . . .	42
4.5	Módulo de Registro . . . . .	45
4.6	Módulo de Integração . . . . .	46

<b>5 Resultados e Análises</b>	<b>50</b>
5.1 Rastreamento dos Usuários . . . . .	50
5.2 Localização dos Usuários . . . . .	53
5.2.1 Teste dos valores no eixo $z$ . . . . .	54
5.2.2 Teste dos valores no eixo $x$ . . . . .	55
5.3 Identificação dos Usuários . . . . .	56
5.4 Integração com middleware <i>UbiquitOS</i> . . . . .	58
<b>6 Conclusão</b>	<b>61</b>
<b>Referências</b>	<b>62</b>
<b>A Kinect</b>	<b>67</b>
<b>B Processamento da imagem</b>	<b>69</b>
B.1 Escala de Cinza . . . . .	69
B.2 Corte da Imagem . . . . .	69
B.3 Redimensionamento . . . . .	70
B.4 Equalização . . . . .	71
<b>C JNI</b>	<b>72</b>
C.1 HelloWorld . . . . .	72

# Listas de Figuras

2.1	Representações de entidades rastreadas. (a) Centróide, (b) múltiplos pontos, (c) representação retangular, (d) representação elíptica, (e) representação de múltiplas partes, (f) esqueleto, (g) contorno por pontos,(h) contorno completo, (i) silhueta (54)	5
2.2	Imagen de profundidade de uma caneca de café (25).	9
2.3	(a) Determina a distância em duas dimensões utilizando lateração. Requer a distância entre a entidade $X$ e três pontos de referência não colineares (22).	10
2.4	Exemplo de uma textura artificial adicionada a cena por meio de pontos de luz infra-vermelha utilizando o método Luz Estruturada (46).	10
2.5	Exemplos de algumas características biométricas (26).	11
2.6	Exemplo de uma imagem de uma pessoa com a mesma expressão facial, vista do mesmo ponto de vista mas sobe diferentes condições de iluminação (37).	15
2.7	Exemplo de um processo de detecção de uma face em uma imagem.	16
2.8	Componentes Básicos do Método <i>Viola-Jones</i> (34).	18
2.9	Exemplo de Características <i>Haar</i> básicas. Adaptada de (2).	18
2.10	Características <i>Haar</i> básicas com dois, três e quatro retângulos (38).	19
2.11	Ilustração de uma classificador em cascata composto com uma cadeia de filtros (2).	20
2.12	Distância euclidiana entre dois pontos em duas dimensões (21).	22
2.13	Direita: imagens de rosto para dez pessoas. Esquerda: os seis primeiros componentes principais, visto como <i>Eigenfaces</i> (21).	23
2.14	Imagens das faces de dois indivíduos. A face de cada indivíduo é apresentada em quatro diferentes condições de iluminação. A variabilidade devido à iluminação aqui é maior do que a variabilidade entre os indivíduos. <i>Eigenfaces</i> tende a confundir as pessoas quando os efeitos de iluminação são muito fortes (21).	24
2.15	As distribuições de dados no reconhecimento com <i>Eigenfaces</i> . Adapatado de (21).	25
3.1	Caixa delimitadora e posição do centro dos olhos no sistema de identificação facial do Projeto CHIL (47).	28
3.2	Fluxo da informação na arquitetura do ambiente inteligente (42).	30
3.3	A configuração do ambiente inteligente (42).	31
3.4	Mapa de ocupação probabilística. Adaptada de (42).	31
3.5	Representação dos ambientes inteligentes MICASA e AVIARY (49).	33
3.6	Método de detecção e reconhecimento facial (49).	34

4.1	Módulos do Sistema TRUE. . . . .	36
4.2	Exemplo de uma imagem composta somente pela região em que o usuário se encontra. . . . .	37
4.3	Módulo de Reconhecimento do Sistema TRUE. . . . .	37
4.4	Exemplo de uma imagem de face normal, em escala de cinza e equalizada. Adaptada de (16). . . . .	38
4.5	Banco de imagens de faces da Universidade de Cambridge (1). . . . .	39
4.6	Fluxo de execução do processo de reconhecimento facial no Sistema TRUE. . . . .	40
4.7	Exemplo de uma imagem de profundidade fornecida pelo <i>Kinect</i> . . . . .	41
4.8	Plano cartesiano de três dimensões onde o <i>Kinect</i> representa a coordenada $(0, 0, 0)$ . . . . .	42
4.9	Imagen do Sistema TRUE de um usuário rastreado e localizado. . . . .	42
4.10	Representação das etapas propostas para o rastreamento. . . . .	43
4.11	Representação da relação que o Módulo de Rastreamento terá com o Módulo de Reconhecimento quando um novo usuário for detectado. . . . .	44
4.12	Exemplo de um usuário rastreado e identificado pelo Sistema TRUE. . . . .	45
4.13	Módulo de Registro do Sistema TRUE. . . . .	45
4.14	Exemplo de imagens resultantes do cadastro do usuário. . . . .	46
4.15	Fluxo básico do treinamento do Sistema TRUE. . . . .	47
4.16	Diagrama de Classe do UserDriver. . . . .	49
5.1	Momento em que um novo usuário foi detectado pelo Sistema TRUE. . . . .	51
5.2	Oclusão parcial de dois usuários. . . . .	51
5.3	Oclusão de usuários. . . . .	52
5.4	Usuários posicionados lado a lado do sensor <i>Kinect</i> a uma distância de 4 metros. . . . .	52
5.5	Usuários sendo rastreado conjuntamente com os objetos que interagem. . . . .	52
5.6	Usuários sofrendo interferência dos que estão ao seu redor. . . . .	53
5.7	Usuários rastreados pelo Sistema TRUE. . . . .	53
5.8	Fotos obtidas pelo Sistema TRUE no teste realizado para os valores do eixo $z$ . . . . .	54
5.9	Gráfico comparativo entre os valores obtidos pelo Sistema TRUE e os valores reais. . . . .	55
5.10	Fotos obtidas pelo Sistema TRUE no teste realizado para os valores do eixo $x$ . . . . .	56
5.11	Gráfico comparativo entre os valores obtidos pelo Sistema TRUE e os valores reais. . . . .	57
5.12	Fluxo básico de execução do <i>listener UserListener</i> . . . . .	60
5.13	Exemplo das mensagens enviadas pelo Twitter aos usuários no ambiente. . . . .	60
A.1	Sensor Kinect da Microsoft. . . . .	67
A.2	Organização interna do Kinect (17). . . . .	68

# **Lista de Tabelas**

2.1	Requisitos teóricos para algoritmos de reconhecimento facial (31). . . . .	13
2.2	Requisitos práticos para algoritmos de reconhecimento facial (31). . . . .	13
4.1	Resultados do teste de reconhecimento feito com o usuário Danilo utilizando as diferentes distâncias. . . . .	40
4.2	Exemplos de dados de reconhecimento mantidos para cada usuário rastreado pelo Módulo de Rastreamento. . . . .	44
5.1	Comparativo entre os valores reais e os valores obtidos pelo Sistema TRUE. .	54
5.2	Comparativo entre as posições reais e os valores obtidos pelo Sistema TRUE. .	56
5.3	Matriz de confusão para apresentar os resultados obtidos. . . . .	58
5.4	Taxas obtidas da Tabela 5.3. . . . .	58
5.5	Matriz de confusão para apresentar os resultados obtidos. . . . .	59
5.6	Taxas obtidas da Tabela 5.5. . . . .	59

# Capítulo 1

## Introdução

A computação ubíqua a tempos vem sendo tema de diversas pesquisas ao redor do mundo. Mark Weiser diz que o computador do futuro deve ser algo invisível (52, 53) proporcionando ao usuário um melhor foco na tarefa e não na ferramenta. A computação ubíqua tenta atribuir essa invisibilidade aos computadores buscando cada vez mais a diminuição do tamanho, a especificidade da tarefa e se acoplando aos objetos do dia-a-dia.

Um ambiente onde a computação ubíqua acontece em sua totalidade é chamado de *SmartSpace* (4). Esse ambiente provê ao usuário uma melhor forma de interagir com os computadores usando diversas tecnologias que estimulam a interatividade natural. Tais tecnologias são capazes de fornecer inteligência ao *SmartSpace*, necessária para concretizar a visão da ubicomp (8).

Para conseguir uma boa interação entre as diversas peças que compõem o *SmartSpace* é necessário que se tenha a disposição informações de contexto, como quem está no ambiente, onde está, o que está fazendo e outras que ajudam o sistema a definir o melhor ajuste dos equipamentos. Com uma base rica de informações de contexto, contendo os perfis dos usuários, garantimos uma maior acurácia na tomada de decisões. Informações de contexto como essas são complicadas de se obter devido a alta dinamicidade do ambiente, no qual usuários entram e saem a todo momento e interagem com diversos equipamentos.

A identificação de um usuário em um *SmartSpace* é feita por meio de sistema de reconhecimento automático. Há alguns anos, um grande número de pesquisas vem sendo desenvolvidas para criação deste tipo de sistema (6). Um dos motivos clássicos é que os métodos baseados em cartões de identificação e senhas não são altamente confiáveis. Estes podem ser perdidos, extraviados e até fraudados (41).

Um ambiente ubíquo capaz de reconhecer seus usuários, pode prover uma personalização automática do ambiente de acordo com as preferências do usuário e até mesmo prover um ambiente mais seguro com controle de acesso físico e prevenção de fraudes (6). Atualmente, os métodos de reconhecimento mais utilizados são baseados no uso de cartões magnéticos e senhas, que requerem sua utilização durante uma transação, mas que não verificam sua idoneidade (11).

Hoje em dia, várias técnicas de reconhecimento por meio de faces, íris, voz, entre outras, vêm sendo estudadas e utilizadas em sistemas de reconhecimento automático (41). O reconhecimento facial pode ser considerada como uma das principais funções do ser humano pois permite identificar uma grande quantidade de faces e aspectos psicológicos

demonstrados pela fisionomia. Pode ser considerada, também, como um problema clássico da computação visual pela complexidade existente na detecção e reconhecimento de características e padrões (6).

O reconhecimento facial vem se desenvolvendo junto a “quarta geração” de computadores através de sua aplicação na nova geração de interfaces que consiste na detecção e reconhecimento de pessoas (6).

É proposta então uma solução para o problema de localização e identificação de perfis de usuários em um *SmartSpace* utilizando como base o middleware *UbiquitOS* (19) integrado com o *Kinect*.

## 1.1 Organização do trabalho

Explicar a estrutura da monografia.

# Capítulo 2

## Fundamentação Teórica

Em ambientes inteligentes, as informações de contexto sobre os usuários presentes como localização e identidade são de grande importância proporcionando uma maior acurácia nas tomadas de decisões. Informações de contexto como essas são muito complicadas de obter devido à dinamicidade do ambiente, no qual usuários entram e saem a todo momento e interagem com diversos equipamentos.

Para obter essas informações é necessário utilizar técnicas de identificação e localização mescladas com técnicas de rastreamento.

O rastreamento pode ser definido como o problema de estimar a trajetória de uma entidade em um plano de imagem à medida em que se move na cena. Em outras palavras, um rastreador atribui rótulos as entidades monitoradas em diferentes quadros de um vídeo (54).

Em um ambiente inteligente, o rastreamento de pessoas é uma das principais ferramentas para detectar novos usuários e serve como base para que novas informações possam ser colhidas. Por exemplo, para que seja possível realizar reconhecimento facial de uma pessoa é necessário, primeiramente, obter imagens de sua face, porém isso só é possível se soubermos onde tal pessoa se encontra na imagem, ou seja, é necessário detectá-la e rastreá-la.

Com o rastreamento mesclado com informações de profundidade é possível, também, obter a localização física das pessoas rastreadas. Tal informação é muito útil em um ambiente inteligente pois, a posição do usuário, possibilita maiores possibilidades para as tomadas de decisões.

Neste capítulo é mostrada uma abordagem conceitual sobre rastreamento de entidades, localização e identificação das mesmas em um ambiente.

Sobre rastreamento é mostrada uma visão geral do processo e quais as dificuldades mais comuns encontradas, bem como as técnicas mais comuns de detecção e rastreamento, as diferentes maneiras de representar as entidades rastreadas e algumas de suas características utilizadas.

Sobre localização é mostrada uma visão geral de como obter a posição relativa a uma câmera utilizando imagens de profundidade.

Sobre identificação é mostrado conceitos gerais sobre biometria e as características biométricas existentes. Sobre reconhecimento facial, é apresentado os conceitos gerais e conceitos mais específicos das diferentes etapas do processo de reconhecimento: detecção

de faces e reconhecimento das mesmas. Além desses conceitos, é apresentado alguns métodos utilizados atualmente em cada uma dessas etapas.

## 2.1 Rastreamento

O rastreamento de entidades, como pessoas ou objetos, é uma importante tarefa do campo da computação visual. A proliferação de computadores com um alto poder computacional, a disponibilidade de câmeras de alta qualidade a um preço acessível e a crescente necessidade de sistemas automáticos de análise de vídeos têm gerado um grande interesse em algoritmos de rastreamento (54).

A detecção e o rastreamento de pessoas tem um grande potencial em aplicações em domínios tão diversos como animação, interação humano-computador, vigilância automatizada (monitorar uma cena para detectar atividades suspeitas), entre outros (54). Por esta razão, tem havido um crescimento notável na investigação deste problema.

O rastreamento de pessoas em um ambiente é considerada como uma tarefa complexa devido à:

1. complexidade do corpo humano;
2. alta dinamicidade do ambiente;
3. ruído nas imagens (54);
4. complexidade do movimento das pessoas;
5. oclusões parciais ou totais de pessoas;
6. variação na iluminação do ambiente (54);
7. processamento em tempo-real (54);

Algumas dessas dificuldades podem ser vencidas com a utilização de imagens de profundidade ao invés de imagens de cor ou intensidade. As imagens de profundidade, além de serem muito pouco sensíveis às variações de iluminação, provêem um fácil entendimento da estrutura da cena, que pode ser utilizada para simplificar as tarefas de rastreamento. Além disso, as câmeras que obtêm imagens de profundidade estão comercialmente disponíveis a um preço acessível (20).

Várias abordagens para rastreamento de objetos já foram propostas. Basicamente, elas se diferem na forma que tratam as seguintes perguntas (54):

- Qual representação do objeto é adequada para o rastreamento (54)?
- Quais características na imagem devem ser utilizadas (54)?
- Como o movimento, aparência e a forma do objeto deve ser modelada (54)?

As respostas para estas perguntas dependem do contexto/ambiente onde o rastreamento será utilizado e do uso final das informações de rastreamento (54).

O rastreamento de pessoas geralmente inicia com o processo de segmentação da imagem da pessoa do resto da imagem. Depois, essas imagens segmentadas são transformadas

em outras representações para reduzir a quantidade de informação ou para atender a um determinado algoritmo. Com isso, deve-se definir como as pessoas vão ser rastreadas quadro a quadro (35).

Basicamente, o processo de rastreamento pode ser dividido em duas etapas:

1. Detecção do objeto;
2. Rastreamento do objeto detectado;

Antes de falar mais sobre cada uma dessas etapas e os métodos existentes para cada, serão apresentadas as maneiras existentes de representar as entidades rastreadas e sobre as características presentes nas imagens que podem ser utilizadas.

### 2.1.1 Representação da Entidade

Nos sistemas de rastreamento, as entidades rastreadas devem ser representadas de alguma maneira. Geralmente, as representações são baseados em suas formas, existindo uma forte relação entre a representação e o algoritmo de rastreamento escolhido (54). A representação é escolhida baseada no domínio da aplicação e as mais utilizadas são:



Figura 2.1: Representações de entidades rastreadas. (a) Centróide, (b) múltiplos pontos, (c) representação retangular, (d) representação elíptica, (e) representação de múltiplas partes, (f) esqueleto, (g) contorno por pontos, (h) contorno completo, (i) silhueta (54)

- **Pontos:** a entidade é representada por um ponto, como por exemplo a centróide (9) da Figura 2.1(a), ou por vários pontos (43), como por exemplo na Figura 2.1(b). Essa representação é mais adequada para rastreamento de entidades que ocupam uma pequena região na imagem (54);

- **Formas geométricas primitivas:** a entidade é representada por formas geométricas simples como um retângulo ou uma elipse, como mostrados nas Figuras 2.1(c) e (d) (10). Essa representação é mais adequada para simples entidades rígidas (54);
- **Silhueta e Contorno:** representação por contorno define os limites de uma entidade, como mostrado nas Figuras 2.1(g) e (h). A região interna do contorno é chamada de Silhueta, como mostrado na Figura 2.1(i). Essa representação é mais adequada para rastrear entidades complexas de forma não rígida (54, 55). Ela é popular devido à sua simplicidade. A silhueta ou contorno de uma entidade pode ser obtida definindo métodos de limiarização ou subtração, podendo ser utilizada tanto com imagens 2D quanto 3D. A representação 2D geralmente é mais simples (35);
- **Modelos de formas articuladas:** entidades articuladas são compostos por partes do corpo que se ligam por meio de juntas. Para representá-las, utiliza-se figuras geométricas para cada parte do corpo, como mostrado na Figura 2.1(e) (54);
- **Modelos de esqueletos:** modelos de esqueletos são extraídos da entidade rastreada, como mostrado na Figura 2.1(f). Essa representação pode ser utilizada tanto para entidades articuladas rígidas quanto não rígidas (54);

Para rastreamento de pessoas a representação por meio de contorno ou silhuetas são as mais adequadas (54).

O rastreamento de várias pessoas de maneira simultânea é considerada uma tarefa muito complexa. As representações das pessoas rastreadas podem se dividir ou fundir em novas representações devido a possíveis oclusões ou ruídos na imagem, e a aparência do objeto pode variar devido a sombras e mudanças da iluminação (35).

### 2.1.2 Seleção de características para rastreamento

A seleção das características é uma tarefa crítica para o rastreamento e está fortemente relacionada com a representação da entidade. Em geral, a seleção procura as características mais singulares para que a entidade rastreada seja facilmente distinguida (54). As características mais comuns utilizadas atualmente são:

- **Cor:** a cor da entidade é influenciada principalmente por duas características: a distribuição da iluminação e a propriedade de reflectância da entidade. Geralmente, a representação *RGB* é utilizada para representar a cor (54);
- **Borda:** os limites de uma entidade geram uma grande variação de intensidade na imagem e são menos sensíveis a variações na iluminação comparado com as cores. A detecção por meio das bordas é utilizada para identificar essas variações de intensidade na imagem. Os algoritmos que detectam as bordas geralmente as utilizam para representação dos mesmos (54);
- **Fluxo óptico:** é um campo denso de vetores de deslocamento que define a tradução de cada *pixel* em uma região. Ele é calculado a partir da restrição de luminosidade, que pressupõe a constância de brilho nos *pixels* correspondentes nos quadros consecutivos (23, 54);

- **Textura:** é a medida da intensidade da variação da superfície que quantifica propriedades como suavidade e regularidade. A Textura é menos sensível a variação da iluminação comparado com a cor (54);

De todas as características, a mais utilizada para rastreamento é a cor (54).

### 2.1.3 Detecção de entidades

Todo método de rastreamento requer um mecanismo de detecção de entidades que pode ser realizada a cada quadro obtido ou na primeira vez que a entidade aparece no vídeo. Os métodos mais populares são:

- **Detector de pontos:** esses detectores são usados para encontrar pontos de interesse dentro da imagem que tem uma expressiva textura na sua respectiva localização. Pontos de interesse são amplamente usados no contexto do movimento e no rastreamento. A qualidade desejável para o ponto de interesse é que seja invariante diante das mudanças de iluminação e ângulo de visão da câmera (54).
- **Subtração de fundo:** é um método popular para segmentação de movimento, especialmente nas situações em que o plano de fundo é relativamente estático. Ele detecta as regiões de movimento na imagem obtendo a diferença *pixel a pixel* entre o quadro corrente e o quadro referente ao plano de fundo (24). Geralmente, um algoritmo de componentes conectadas é aplicado para obter regiões conectadas que correspondem a uma entidade (54).
- **Segmentação:** o objetivo do algoritmo de segmentação é partitionar a imagem em regiões com certo grau de similaridade. Todo algoritmo de segmentação tem dois problemas: o critério para definir uma boa partição e o método para arquivar particionamentos eficientes (44, 54).
- **Aprendizagem supervisionada:** a detecção de entidades pode ser feita pelo aprendizado automático de diferentes entidades de um conjunto de exemplos por meio de um mecanismo de aprendizado. Esse aprendizado requer o armazenamento de um conjunto de *templates*. A partir desse conjunto de informações, o algoritmo gera uma função que mapeia as possíveis entradas para as saídas desejadas. Um problema padrão é a classificação onde a função gera um valor contínuo a partir de um determinado comportamento da entidade. No contexto da detecção de entidades as informações armazenadas são compostas por um par de características de entidades e uma classe associada onde ambos os valores são manualmente definidos. A seleção de características tem um papel importante no desempenho da classificação, portanto, é importante usar um conjunto de características que seja possível discriminar uma classe das outras (54).

### 2.1.4 Rastreamento de entidades

O objetivo do rastreamento de entidades é conhecer a trajetória do mesmo no tempo localizando sua posição em cada quadro. O rastreamento também pode prover a região completa na imagem ocupada pela entidade a cada instante (54).

As atividades de detecção de entidades e de estabelecimento de correspondências entre elas e as instâncias dos quadros podem ser realizadas tanto separadamente como concomitantemente. No primeiro caso, as prováveis regiões de entidades são obtidas e o rastreador encontra correspondência entre as entidades e os quadros. No último caso, as prováveis regiões e as correspondências são feitas juntas e estimadas pela atualização iterativa da localização da entidade e de regiões de informações obtidas dos quadros anteriores (54).

Os métodos de rastreamento de entidades mais utilizados atualmente são:

- **Rastreamento de pontos:** as entidades detectadas a cada quadro são representadas por pontos e a associação entre os pontos é feita com base no estado anterior da entidade que pode incluir a posição e o movimento. Essa abordagem requer um mecanismo externo para detectar a entidade a cada quadro (54);
- **Rastreamento de núcleo:** as entidades são rastreados pelo cálculo do movimento do núcleo em quadros consecutivos. Esse movimento geralmente está na forma de transformações paramétricas como translação e rotação. O núcleo se refere ao formato ou aparência da entidade (54).
- **Rastreamento de silhuetas:** o rastreamento é feito estimando a região da entidade a cada quadro. Esse método utiliza informações contidas dentro da região estimada. Esta informação pode ser na forma de densidade de aparência e modelos de forma que estão, geralmente, na forma de mapas de borda. Dado os modelos de entidades, silhuetas são rastreadas por qualquer forma de correspondência ou evolução de contorno. Ambos os métodos podem ser essencialmente considerados como segmentação de entidades aplicada no domínio temporal utilizando os priores gerados a partir dos quadros anteriores (54).

Com as entidades rastreadas, para obter a posição dos mesmos deve-se obter informações de profundidade. Na próxima seção falaremos sobre como essas informações são obtidas utilizando imagens de profundidade.

## 2.2 Localização

Calcular a distância de vários pontos na cena relativa a posição da câmera é uma importante tarefa de sistemas de computação visual (25). Para isso, deve-se obter informações de profundidade das entidades em interesse. Essas informações podem ser obtidas utilizando imagens de intensidade ou de profundidade.

Uma maneira comum de se obter informações de profundidade de imagens de intensidade é adquirir um par de imagens usando duas câmeras deslocadas entre si por uma distância conhecida. Como alternativa, duas ou mais imagens obtidas de uma câmera em movimento também pode ser utilizadas para calcular informações de profundidade (25). Esse método é conhecido como *Stereo Vision* e necessita ser bem calibrado. Além disso, os algoritmos que o implementa geralmente são computacionalmente caros e não funcionam em ambiente com baixa condição de iluminação (40).

Informações de profundidade também podem ser obtidas indiretamente através de imagens de intensidade utilizando sinais na imagem, como sombreamento e textura (25).

Em contraste com imagens de intensidade, imagens cujo valor em cada pixel é uma função da distância do ponto correspondente na cena do sensor são chamadas de imagens de profundidade, exemplificada na Figura 2.2. Tais imagens podem ser adquiridas diretamente utilizando sensores específicos (25). Alguns dos métodos mais conhecidos são:

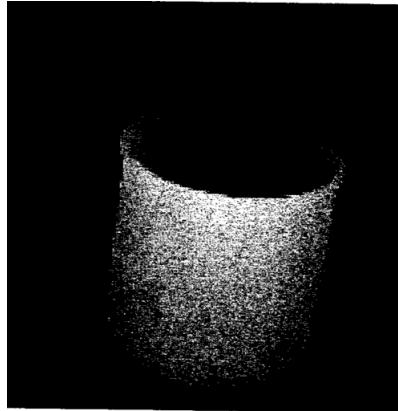


Figura 2.2: Imagem de profundidade de uma caneca de café (25).

1. **Triangulação:** utiliza as propriedades geométricas do triângulo para calcular a localização de entidades. Pode ser dividida em duas subcategorias: lateração e angulação. Lateração computa a posição de uma entidade estimando sua distância de múltiplos ponto de referência. Calcular a posição de uma entidade em duas dimensões requer estimativas de distância de três pontos não colineares como mostrado na Figura 2.3a. Já em três dimensões são necessários quatro pontos não coplanares. Ângulação utiliza ângulos para determinar a distância da entidade. Em geral, ângulação em duas dimensões requer estimativas de dois ângulos e a estimativa da distância entre dois pontos de referência como mostrado na Figura 2.3b (22);
2. **Tempo de Vôo (TOF - *Time of flight*):** a distância até a entidade é calculada observando a diferença de tempo entre o pulso eletromagnético transmitido e recebido. A informação de profundidade também pode ser obtida através da detecção da diferença de fase entre as ondas transmitidas e as recebidas de um feixe de amplitude modulada (25, 40). Câmeras TOF provêm imagens de profundidade com melhor acurácia em relação ao método de *Stereo Vision*, porém são muito caras e pouco acessíveis (40);
3. **Luz Estruturada:** uma imagem de profundidade não pode ser obtida utilizando somente um sensor de vídeo. Porém, adicionando uma textura artificial na cena, como na Figura 2.4, uma imagem de profundidade pode ser recuperada. Esse princípio consiste na projeção de pontos de luz infra-vermelhos na cena que são recuperados por uma câmera infra-vermelha que lê a textura. Trata-se de um método mais acessível que o TOF, porém é pouco eficiente para estimar a distância dos pontos nas bordas dos objetos e em áreas muito longe do sensor (40);

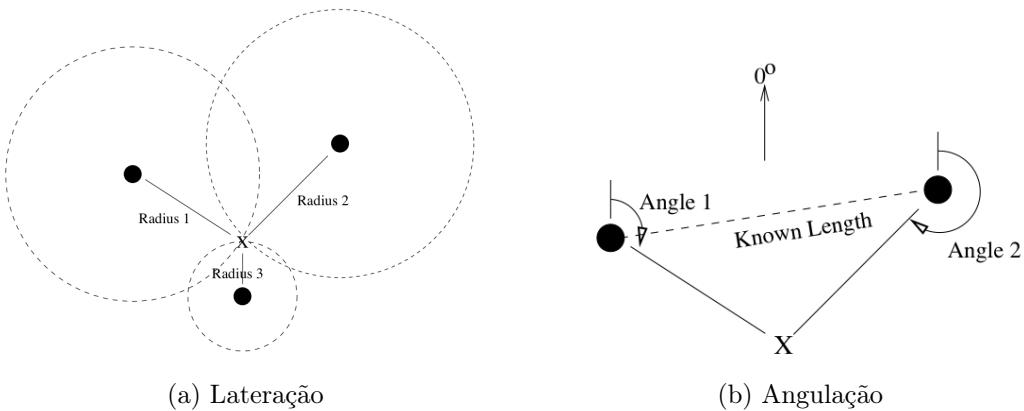


Figura 2.3: (a) Determina a distância em duas dimensões utilizando lateração. Requer a distância entre a entidade  $X$  e três pontos de referência não colineares (22).  
 (b) Exemplo de uma angulação em duas dimensões em que se localiza a entidade  $X$  utilizando ângulos relativos a um vetor de referência  $0^\circ$  e a distância entre dois pontos de referência (22).

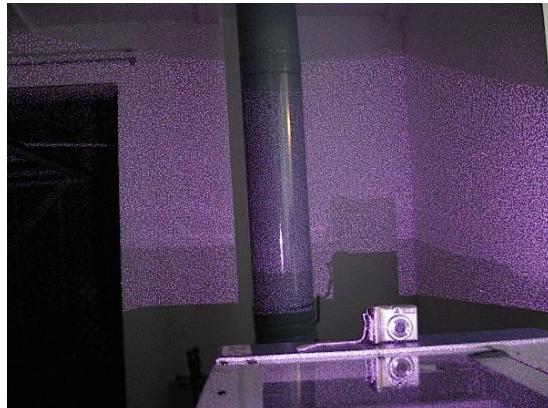


Figura 2.4: Exemplo de uma textura artificial adicionada a cena por meio de pontos de luz infra-vermelha utilizando o método Luz Estruturada (46).

Imagens de profundidade são úteis devido a sua especificação explícita de valores de profundidade. Ao mesmo tempo acreditava-se que se a informação de profundidade fosse disponibilizada de maneira explícita, o processamento posterior da imagem seria facilitado. Tornou-se claro que a informação de profundidade ajuda, porém a tarefa básica de interpretação de imagens mantém todas as suas dificuldades (25).

## 2.3 Identificação

Um ambiente ubíquo capaz de identificar seus usuários, pode prover uma personalização automática do ambiente de acordo com as preferências do usuário e até mesmo prover um ambiente mais seguro com controle de acesso físico e prevenção de fraudes (6). A

identificação de um usuário em um ambiente inteligente é feita por meio de sistemas de reconhecimento automático. Há alguns anos, um grande número de pesquisas vem sendo desenvolvidas para criação deste tipo de sistema (6).

As abordagens de identificação pessoal que utilizam “alguma coisa que você sabe”, como um número de identificação pessoal (PIN - *Personal Identification Number*), ou “alguma coisa que você tenha”, como um cartão de identificação, não são confiáveis o suficiente para satisfazer os requisitos de segurança de um sistema de identificação porque não têm a capacidade de diferenciar um usuário legítimo de um impostor que adquiriu de forma ilegal o privilégio de acesso (27). Esta fragilidade pode ser evitada utilizando biometria: alguns traços físicos ou comportamentais são muito mais complicados de serem forjados que uma cadeia de caracteres (26).

Hoje em dia, várias técnicas de reconhecimento biométrico por meio de faces, íris, voz, entre outras, vêm sendo estudadas e utilizadas em sistemas de reconhecimento automático (41). Dentre essas, o reconhecimento por meio de faces se destaca pois sua aquisição é realizada de maneira fácil e não intrusiva, tornando-a ideal para ser utilizada em um ambiente inteligente.

### 2.3.1 Biometria

Biometria é uma tecnologia utilizada para identificação de um indivíduo baseado em suas características físicas ou comportamentais para realizar a identificação e, por isso, tem a capacidade de diferenciar entre um indivíduo legítimo de um impostor (27). As características físicas estão relacionadas a composição do corpo humano e seu formato e as comportamentais estão relacionadas a forma como o corpo humano faz algo (26). A Figura 2.5 contém alguns exemplos desses dois tipos diferentes de características biométricas.

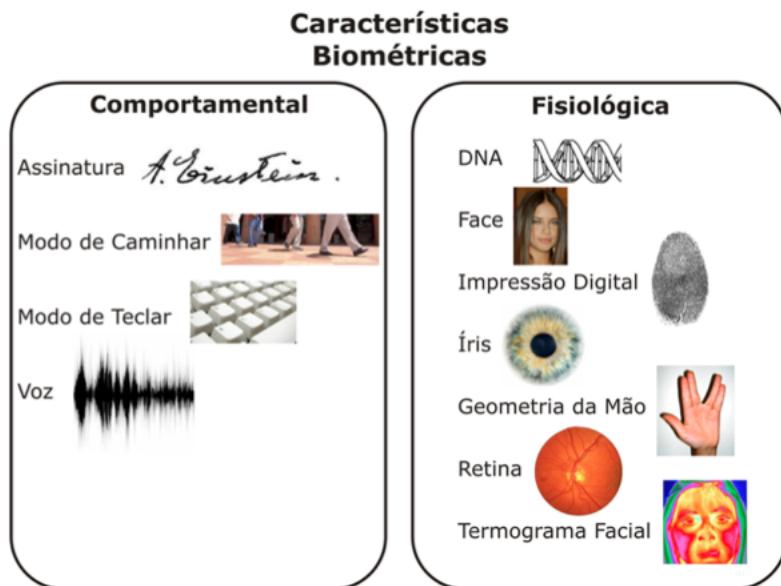


Figura 2.5: Exemplos de algumas características biométricas (26).

Teoricamente, qualquer característica física/comportamental pode ser utilizada para identificação caso siga alguns dos seguintes requisitos (31):

1. **universalidade:** qualquer pessoa comum pode ser avaliada sobre essa característica;
2. **singularidade:** dada duas pessoas distintas, elas não podem ter a mesma característica dentro de uma proporção satisfatória;
3. **permanência:** a característica não pode mudar significativamente de acordo com o tempo;
4. **exigibilidade:** pode ser mensurada quantitativamente;

Porém, na prática também são considerados outros requisitos (31):

1. **desempenho:** o processo de identificação deve apresentar um resultado aceitável;
2. **aceitação:** indica em que ponto as pessoas estão dispostas a aceitar o sistema biométrico;
3. **evasão:** refere-se a facilidade de ser adulterado;

São várias as vantagens que os sistemas biométricos têm em relação aos sistemas convencionais. Listamos as vantagens principais (26):

- características biométricas não podem ser perdidas ou esquecidas;
- características biométricas são difíceis de serem copiadas, compartilhadas e distribuídas;
- os sistemas biométricos necessitam que a pessoa esteja presente no local da autenticação;

Na prática um sistema biométrico deve ser capaz de (27):

1. atingir uma acurácia aceitável e com uma velocidade razoável;
2. não ser prejudicial aos indivíduos e ser aceito pela população alvo;
3. ser suficientemente robusto para métodos fraudulentos;

Novas técnicas de reconhecimento por meio de face, íris, retina e voz, entre outras, têm sido abordadas para aplicações em sistemas de reconhecimento automático (6, 41). Das nove características utilizadas atualmente a face é uma das mais populares (31). Nas Tabelas 2.1 e 2.2 são mostradas as nove características e seus respectivos comportamentos baseados nos requisitos mencionados acima.

Os sistemas biométricos podem ser classificados em sistemas de verificação ou identificação:

Tabela 2.1: Requisitos teóricos para algoritmos de reconhecimento facial (31).

Biometria	Universidade	Singularidade	Permanência	Exigibilidade
<b>Face</b>	Alta	Baixa	Média	Alta
<b>Digital</b>	Média	Alta	Alta	Média
<b>Geometria da Mão</b>	Média	Média	Média	Alta
<b>Veia da Mão</b>	Média	Média	Média	Média
<b>Iris</b>	Alta	Alta	Alta	Média
<b>Retina</b>	Alta	Alta	Média	Baixa
<b>Assinatura</b>	Baixa	Baixa	Baixa	Alta
<b>Voz</b>	Média	Baixa	Baixa	Média
<b>Termograma</b>	Alta	Alta	Baixa	Alta

Tabela 2.2: Requisitos práticos para algoritmos de reconhecimento facial (31).

Biometria	Desempenho	Aceitação	Evasão
<b>Face</b>	Baixa	Alta	Baixa
<b>Digital</b>	Alta	Média	Alta
<b>Geometria da Mão</b>	Média	Média	Média
<b>Veia da Mão</b>	Média	Média	Alta
<b>Iris</b>	Média	Média	Alta
<b>Retina</b>	Alta	Baixa	Alta
<b>Assinatura</b>	Baixa	Alta	Baixa
<b>Voz</b>	Baixa	Alta	Baixa
<b>Termograma</b>	Média	Alta	Alta

- Sistemas de verificação são aqueles que autenticam a identidade dos usuários comparando-os com os próprios *templates*. Eles conduzem uma comparação “um para um” e determinam se o usuário é quem realmente diz ser. O maior desafio para esse tipo de sistema é a acurácia. Geralmente, não é muito difícil satisfazer o requisito de tempo de resposta pois somente uma comparação “um para um” é feita (27).
- Sistemas de identificação reconhecem um indivíduo pesquisando em todo o banco de dados procurando por uma correspondência. Eles conduzem uma comparação “um para muitos” para estabelecer a identidade do indivíduo. Ao contrário dos sistemas de verificação, nesse tipo de sistema tanto a acurácia quanto o tempo são os grandes desafios, por causa da necessidade de explorar todo o banco de dados (27).

A face é uma característica biométrica bastante atrativa. Um dos motivos que incentivou os diversos estudos sobre a utilização da face para reconhecimento são as vantagens que ela possui em relação a impressão digital e a íris. No reconhecimento por impressão digital a desvantagem consiste no fato que nem todas as pessoas possuem uma impressão digital com “qualidade” suficiente para ser reconhecida por um sistema. Já o reconhecimento por íris apresenta uma alta confiabilidade e larga variação, sendo estável pela vida

toda. Porém, a desvantagem está relacionada ao modo de captura da íris que necessita de um alinhamento entre a câmera e os olhos da pessoa (6).

Além disso, aquisição da face é feita de forma fácil e não-intrusiva e possui uma baixa privacidade de informação, como a face é exposta constantemente, caso uma base de faces seja roubada, essas informações não representam algum risco e não possibilitam um uso impróprio.

### 2.3.2 Reconhecimento Facial

O reconhecimento facial é uma das atividades mais comuns realizadas diariamente por seres vivos dotados de certa inteligência. Essa simples atividade vem despertando o interesse de pesquisadores que trabalham com computação visual e inteligência artificial. O objetivo desses pesquisadores é de construir sistemas artificiais capazes de realizar o reconhecimento de faces humanas e a partir desta capacidade, construir os mais diferentes tipos de aplicações: sistemas de vigilância, controles de acesso, definições automáticas de perfis, entre outros (36).

No anos 70, os estudos do reconhecimento facial eram baseados sobre atributos faciais mensuráveis como olhos, nariz, sobrancelhas, bocas, entre outros. Porém, os recursos computacionais eram escassos e os algoritmos de extração de características eram ineficientes. Nos anos 90, as pesquisas na área ressurgiram, inovando os métodos existentes (6, 27) e disseminando a técnica.

Umas das maiores dificuldades dos sistemas de reconhecimento é tratar a complexidade dos padrões visuais. Mesmo sabendo que todas as faces possuem padrões reconhecidos, como boca, olhos e nariz, elas também possuem variações únicas que devem ser utilizadas para determinar as características relevantes. Outra dificuldade encontrada em relação a essas características é que elas possuem uma larga variação estatística para serem consideradas únicas para cada indivíduo. O ideal seria que a variância inter-classe seja grande e a intra-classe pequena, pois assim imagens de diferentes faces geram os códigos mais diferentes possíveis, enquanto imagens de uma mesma face geram os códigos mais similares possíveis. Portanto, estabelecer uma representação que capture as características ideais é um desafio (6).

Do ponto de vista geral, o reconhecimento facial continua sendo um problema aberto por causa de várias dificuldades que aumentam a variância intra-classe (27). Entre estas, destacamos as mais comuns (6):

- iluminação;
- ângulos e poses;
- expressões;
- cosméticos e acessórios;
- extração da face do contexto ou do fundo;

Na Figura 2.6, temos um exemplo de como uma mesma pessoa, com a mesma expressão facial, vista do mesmo ponto de vista, pode parecer drasticamente diferente quando as fontes de luz possuem diferentes direções (37).

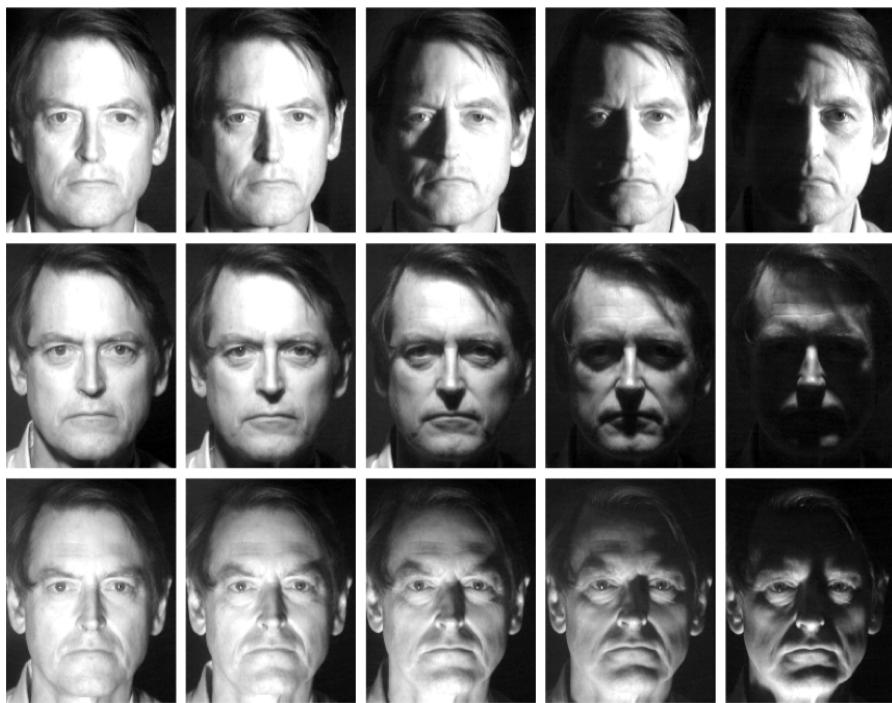


Figura 2.6: Exemplo de uma imagem de uma pessoa com a mesma expressão facial, vista do mesmo ponto de vista mas sobe diferentes condições de iluminação (37).

No contexto de identificação, o reconhecimento facial se resume no reconhecimento de um “retrato” frontal, estático e controlado. Estático pois os “retratos” utilizados nada mais são que imagens e controlado pois a iluminação, o fundo, a resolução dos dispositivos de aquisição e a distância entre eles e as faces são essencialmente fixas durante o processo de aquisição da imagem (27).

Basicamente, o processo de reconhecimento facial pode ser dividido em duas tarefas principais (27):

1. Detecção de faces em imagens;
2. Reconhecimento das faces encontradas;

### **Detecção de faces em imagens**

A primeira etapa para o reconhecimento de faces é a detecção de um rosto, e a partir daí a comparação do mesmo com modelos conhecidos pelo sistema (27, 36). Em um sistema de reconhecimento facial, tanto o tempo de resposta quanto a confiabilidade desta etapa influenciam diretamente no desempenho e o emprego deste sistema (36).

A detecção de faces é definida como o processo que determina a existência ou não de faces em uma imagem e uma vez encontrada alguma face, sua localização deve ser apontada através de um enquadramento ou através de suas coordenadas dentro da imagem (36). A Figura 2.7 representa um exemplo da detecção de uma face em uma imagem.

O processo de detecção de faces geralmente é dificultado pelas seguintes razões mostradas a seguir:

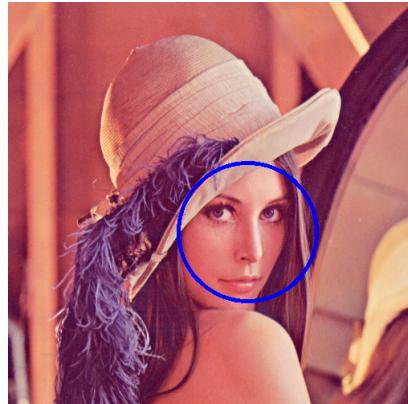


Figura 2.7: Exemplo de um processo de detecção de uma face em uma imagem.

1. **Pose:** as imagens de uma face podem variar de acordo com a posição relativa entre a câmera e a face (frontal, 45 graus, perfil, “de cabeça para baixo”), e com isso algumas características da face, como olhos e nariz, podem ficar parcialmente ou totalmente ocultadas (33).
2. **Presença de acessórios:** características faciais básicas importantes para o processo de detecção podem ficar ocultadas pela presença de acessórios, como óculos, bigode, barba, entre outros (33, 36).
3. **Expressões faciais:** embora a maioria das faces apresente estruturas semelhantes (olhos, bocas, nariz, entre outros) e são dispostas aproximadamente na mesma configuração de espaço, pode haver um grande número de componentes não rígidos e texturas diferentes entre as faces. Um exemplo são as flexibilizações causadas pelas expressões faciais (33, 36);
4. **Obstrução:** faces podem ser obstruídas por outros objetos. Em uma imagem com várias faces, uma face pode obstruir outra (33).
5. **Condições da imagem:** a não previsibilidade das condições da imagem em ambientes sem restrições de iluminação, cores e objetos de fundo (33, 36).

Atualmente, já existem diferentes métodos/técnicas de detecção de faces. Tais métodos podem ser baseados em imagens de intensidade e em imagens de cor ou também em imagens de três dimensões. Os principais métodos de imagens de cor e de intensidade podem ser divididos em quatro categorias:

1. **Métodos baseados em conhecimento:** métodos, desenvolvidos principalmente para localização facial, baseados em regras derivadas do conhecimento dos pesquisadores do que constitui uma típica face humana. Normalmente, captura as relações existentes entre as características faciais. É fácil encontrar regras que descrevem as características faciais, por exemplo, uma face sempre é constituída por dois olhos simétricos, um nariz e uma boca. As relações entre essas características podem ser representadas pelas distâncias relativas e posições. Este método possui desvantagens em relação à construção do conjunto de regras. Se estas são muito gerais,

corre-se o risco, de que o sistema que as utilizam, apresentar uma alta taxa de falsos positivos. Se são muito específicas, podem ser ineficazes ao tentar detectar faces por não satisfazerem todas as regras, diminuindo muito a precisão da detecção (29, 33);

2. **Métodos baseados em características invariantes:** esses algoritmos tem como objetivo principal encontrar as características estruturais que existem mesmo quando a pose, o ângulo e as condições de iluminação variam. E por meio dessas características localizar a face. São desenvolvidos principalmente para localização facial (33). A principal desvantagem desse método é que tais características invariantes podem ser corrompidas devido a algum tipo de ruído ou as fortes variações nas condições de iluminação, comprometendo a eficiência. A cor da pele e a textura da face são as principais características invariantes que podem ser utilizadas para separar a face de outros objetos (29);
3. **Métodos baseados em padrões:** vários padrões comuns de um rosto são armazenados tanto para descrever o rosto como um todo quanto para descrever as características faciais separadamente. As correlações entre as imagens de entrada e os padrões armazenados são computados para detecção. Esses métodos são desenvolvidos para serem utilizados tanto para localização e como para detecção facial (33);
4. **Métodos baseados em aparência:** recebem este nome devido ao fato de não utilizarem nenhum conhecimento, a priori, sobre o objeto ou características a serem detectadas. Em contraste com os métodos baseado em *templates*, os modelos são retirados de um conjunto de imagens de treinamento que devem capturar a variabilidade da face. Esses modelos retirados são utilizados para detecção. Nesta classe de algoritmos surge os conceitos de aprendizado e treinamento, uma vez que as informações necessárias para realizar a tarefa de detecção são retiradas do próprio conjunto de imagens sem intervenção externa (29, 33).

Um problema relacionado e muito importante é como avaliar a performance dos métodos de detecção de faces propostos. Para isso, muitas métricas foram adotadas como tempo de aprendizagem, número de amostras necessárias no treinamento e a proporção entre taxas de detecção e “falso alarme”. Esta última é dificultada pelas diferentes definições para as taxas de detecção e falso alarme adotadas pelos pesquisadores (33).

O método *Viola-Jones* é baseado em padrões para detecção de objetos, o que minimiza o tempo de computação, e possui uma alta acurácia permitindo a detecção de faces em tempo real. Este método pode ser utilizado para construir uma abordagem de detecção facial rápida e eficaz utilizando apenas imagens em tons de cinza distinguindo-se dos outros métodos que utilizam informações auxiliares como a diferença em sequência de vídeos e o uso de cores (34, 38). Apesar da simplicidade obtém altas taxas de detecção (34). O método é implementado pela biblioteca *OpenCV* (7) (*Open Source Computer Vision*) e é amplamente utilizado.

O Método *Viola-Jones* (2, 34, 38) para detecção facial utiliza quatro conceitos chaves ilustrados na Figura 2.8:

1. **Características Haar básicas:** simples características retangulares avaliadas rapidamente por meio de uma nova representação da imagem chamada “Imagem Integral”;

2. **Imagen Integral:** uma nova representação da imagem que permite uma rápida avaliação de recursos e características. Basicamente, o método consiste em acrescentar pequenas unidades juntas. Neste caso, pequenas unidades são valores de *pixels*. O valor “integral” para cada *pixel* é a soma de todos os *pixels* acima e a esquerda. Começando pelo canto superior esquerdo da imagem e atravessando para direita e para baixo, toda a imagem pode ser “integrada” com poucas operações por *pixel*;
3. **O método AdaBoost:** um algoritmo de aprendizado *boosting* utilizado para construir um classificador, selecionando um pequeno número de características importantes;
4. **Classificadores em cascata:** classificadores fracos *boosted* em uma estrutura de árvore que geram inferências rápidas e robustas na construção de um classificador forte;

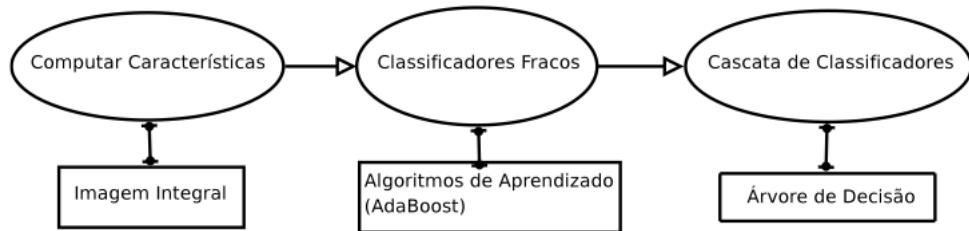


Figura 2.8: Componentes Básicos do Método *Viola-Jones* (34).

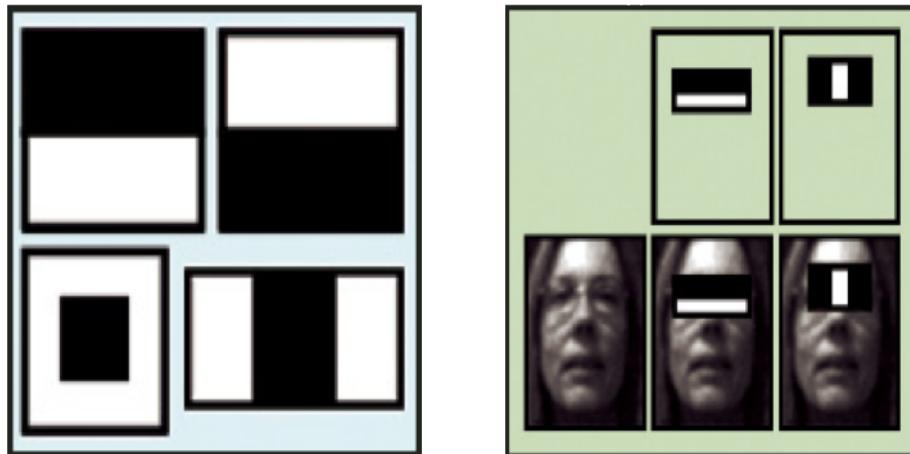


Figura 2.9: Exemplo de Características *Haar* básicas. Adaptada de (2).

A detecção facial em imagens é baseada em simples características retangulares, exemplificada na Figura 2.9. Existem inúmeros motivos para se usar as características retangulares, uma das principais é que sistemas baseados em características são muito mais rápidos que sistemas baseados em *pixels* (38).

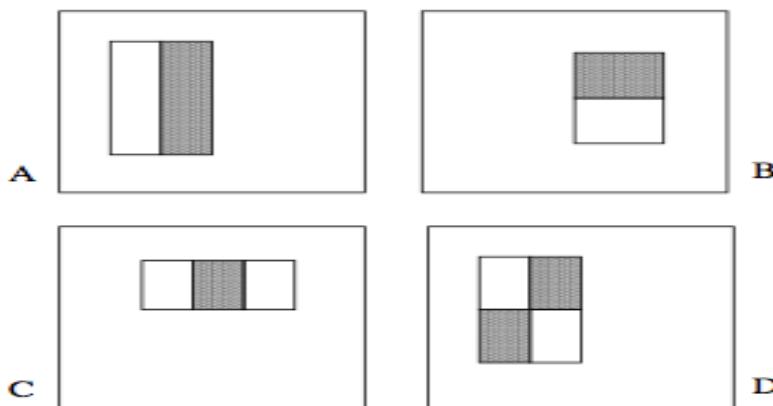


Figura 2.10: Características *Haar* básicas com dois, três e quatro retângulos (38).

Estas simples características são remanescentes das funções de base *Haar*. O método utiliza três tipos de características, exemplificadas na Figura 2.10: características com dois, três ou quatro retângulos (38). A presença de uma característica em uma imagem é determinada pela subtração da média dos valores dos *pixels* da região escura pela média dos valores dos *pixels* da região clara. Caso o valor seja maior que um limiar, então essa característica é tida como presente (2).

O método *Viola-Jones* não trabalha diretamente com as intensidades da imagem. Para determinar a presença ou ausência de centenas de características *Haar* em cada posição de imagem e em várias escalas de forma eficiente, utiliza-se uma técnica chamada “Imagen Integral” (2, 38), em que uma nova representação de imagem é criada. Com isso, qualquer característica pode ser computada para qualquer escala e localização em um tempo constante (38).

Para selecionar características *Haar* básicas que serão utilizados e para definir os limiares, o método *Viola-Jones* utiliza um método de aprendizagem de máquina chamado *AdaBoost*. Este combina vários classificadores “fracos” para criar um classificador “forte”.

Um classificador fraco é aquele que só obtém a resposta correta um pouco mais frequente que um “palpite aleatório”. A combinação desses classificadores “fracos”, onde cada um “empurra” a resposta final um pouco na direção certa, pode ser considerado como um classificador “forte”. O método *AdaBoost* seleciona um conjunto de classificadores “fracos” para combinar e atribui pesos a cada um. Essa combinação ponderada resulta em um classificador “forte” (2).

Em qualquer região de uma imagem, o número total de características *Haar* básicas é muito grande. Para assegurar uma classificação rápida, o processo de aprendizagem deve excluir o maior número de características disponíveis, e focar nas que são críticas. A seleção dessas características é alcançada através de uma simples modificação no método *AdaBoost*: o mecanismo de aprendizagem é construído de forma que cada classificador “fraco” retornado dependa de somente uma única característica. Como resultado, cada estágio do processo seleciona um novo classificador “fraco” o que pode ser visto como um processo de seleção de características. O *AdaBoost* fornece um algoritmo de aprendizagem eficaz (38).

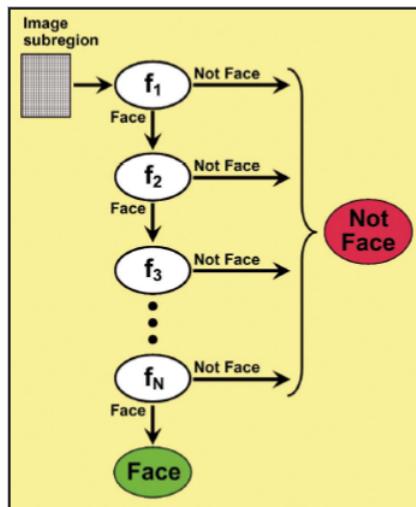


Figura 2.11: Ilustração de uma classificador em cascata composto com uma cadeia de filtros (2).

O método *Viola-Jones* combina uma série de classificadores *AdaBoost* na forma de uma cadeia de filtros, como na Figura 2.11, que recebe o nome de “Classificadores em Cascata”. Cada filtro em si é um classificador *AdaBoost* com um número relativamente pequeno de classificadores “fracos” (2).

O limiar de aceitação, em cada nível, é definido “baixo” o suficiente para que passe por todos, ou quase todos, os exemplos de face do conjunto de treinamento (um grande banco de imagens contendo faces). Os filtros em cada nível são treinados para classificar imagens de treinamento que passaram por todas fases anteriores (2).

Durante a utilização, se alguma região de uma imagem falhar em passar em um desses filtros, esta é imediatamente classificada como “não face”. Quando uma região de uma imagem passa por um filtro, ela vai para o próximo filtro na cadeia. As regiões das imagens que passarem por todos os filtros na cadeia são classificadas como “faces” (2).

O algoritmo utilizado para construção dos “Classificadores em Cascata” alcança um ótimo desempenho e, ao mesmo tempo, reduz drasticamente o tempo de computação. O aspecto chave é que os menores classificadores (filtros), e por isso mais eficientes, podem ser utilizados para rejeitar a maioria das regiões das imagens que não são faces antes que os classificadores mais complexos sejam utilizados (38). A ordem dos filtros no classificador é baseado nos pesos que o método *AdaBoost* define para cada filtro. Os filtros com maior peso são colocados no início, para eliminar as regiões das imagens que não são faces o mais rápido possível (2).

O método *Viola-Jones* é adequado para utilização em sistemas de detecção de faces em tempo real. Agora, o próximo passo para o processo de “Reconhecimento Facial” é comparar as faces encontradas com modelos conhecidos pelo sistema para realizar a identificação.

## Reconhecimento das Faces encontradas

Na etapa de reconhecimento, as faces detectadas, serão comparadas com um banco de dados de faces conhecidas. Várias técnicas são usadas para acelerar essa comparação já

que para identificar o usuário é necessário um grande número de comparações. Dentre as técnicas utilizadas existem duas variações principais, as que usam como entrada dados de imagens de intensidade e de cor e as que usam como entradas dados de imagens de profundidade.

As técnicas que utilizam imagens de cor e de intensidade são as mais antigas e comuns e são amplamente utilizadas. Dentre elas destacam-se:

1. **Eigenfaces:** extrai as informações relevantes de uma face, codifica-a da maneira mais eficiente possível, e a compara com um banco de faces codificadas de maneira similar. Extrai as informações relevantes contidas em uma imagem de uma face de uma maneira simples captando a variação em uma coleção de imagens de face e usando essa informação para codificar e comparar imagens individuais de faces (32). É baseado em projeção linear das imagens em uma espaço de imagens de menor dimensão. Para redução dimensional utiliza PCA - “*Principal Component Analisys* (Análise de componente principal)” maximizando a dispersão total entre todas as imagens (37);
2. **Redes Neurais:** uma rede neural artificial é um modelo computacional capaz de, entre outras funções, armazenar, classificar padrões, realizar interpolação de funções não-lineares e apresentar soluções heurísticas para problemas de otimização. Isso é conseguido através de um processo denominado “aprendizado”. O uso de redes neurais visa tornar o sistema de reconhecimento capaz de absorver pequenas variações ocorridas no momento da coleta de medidas faciais. Espera-se, portanto, mais robustez a falhas e que responda de forma mais confiável (36);
3. **Fisher Faces:** igual ao *Eigenfaces*, é um método que procura uma projeção linear das faces para um espaço dimensional menor onde os impactos causados pelas variações de luz e expressões faciais são minimizados. O método é derivado da discriminante linear de Fisher (FLD) (37).

O *Eigenfaces* trata-se de uma técnica bastante satisfatória quando utilizada sobre uma base de dados (faces) relativamente grande, permitindo ao sistema inferir, das imagens suas principais características e, partindo delas, realizar o reconhecimento das imagens utilizando um número bastante reduzido de cálculos (12), permitindo, assim, um reconhecimento em tempo real.

Os princípios básicos por trás dele, como PCA - “*Principal Component Analisys*” (Análise de componente principal) e *Distance-Based Matching* (Correspondência Baseada na Distância) aparecem cada vez mais na computação visual e em diversas aplicações de inteligência artificial (21).

Basicamente, as etapas do processo de reconhecimento são simples e bem definidas. Dada uma imagem de um rosto desconhecido e imagens do rosto das pessoas conhecidas executa as seguintes ações (21):

1. Computa a “distância” entre a nova imagem e cada uma das imagens já conhecidas.
2. Seleciona a imagem mais próxima do rosto em questão.
3. Se a “distância” da nova imagem para a imagem já catalogada for menor que o limite pré-definido, “reconhece” a imagem caso contrário classifica como “desconhecida”.

Em *Eigenfaces*, a distância entre as imagens é medida ponto a ponto entre as imagens conhecidas e as desconhecidas. A distância Euclidiana, medida entre dois pontos  $P_1$  e  $P_2$  em duas dimensões, é dada pela fórmula  $d_{12} = \sqrt{(d_x + d_y)}$ , onde  $d_x = (x_2 - x_1)^2$  e  $d_y = (y_2 - y_1)^2$  e representada na Figura 2.12 (21). Para se medir a distância entre as

imagens usa-se a seguinte formula  $d(\vec{X}, \vec{Y}) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$  (39).

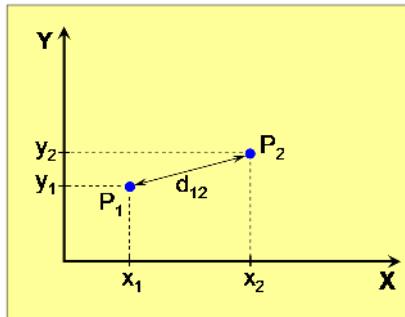


Figura 2.12: Distância euclidiana entre dois pontos em duas dimensões (21).

A distância Euclidiana é simples e fácil de implementar mas estudos mostram que é possível obter melhores resultados utilizando uma outra distância, conhecida como Mahalanobis. Essa distância, usada bastante na estatística, é útil para determinar a similaridade de uma amostra desconhecida de uma conhecida. Ao contrário da distância Euclidiana, a Mahalanobis correlaciona todos os pontos de modo a detectar melhor os padrões existentes. O calculo é feito através da formula  $d(\vec{X}, \vec{Y}) = -\frac{1}{\sqrt{\sum_{i=1}^n x_i^2} \sqrt{\sum_{i=1}^n y_i^2}} \sum_{i=1}^n z_i x_i y_i$

onde  $z_i = \sqrt{\frac{\lambda_i}{\lambda_i + \alpha^2}}$  e  $\alpha = 0,25$  (39).

Imagens possuem “ruídos” e vamos definir ruído como qualquer coisa que atrapalhe na identificação, como por exemplo, as diferenças na luminosidade. Cada *pixel* possui uma intensidade de ruído diferente. Com cada *pixel* contribuindo para o ruído total, este se torna muito elevado comparado com a informação útil que se possa retirar da imagem, dificultando o processo de reconhecimento. Uma solução é diminuir a dimensionalidade da imagem, tornando assim o ruído menor e possibilitando extrair, da imagem, as informações importantes (21).

O *Eigenfaces* utiliza o método PCA - “*Principal Component Analisys*” (Análise de componente principal) para reduzir a dimensionalidade da imagem (21). Este método é interessante para dar-nos alguma intuição sobre as componentes principais para o nosso conjunto de dados. O lado esquerdo da Figura 2.13 mostra as imagens das faces de dez pessoas. O lado direito mostra os seis primeiros componentes principais deste conjunto de dados, apresentados como *eigenfaces*. Uma *eigenface* da componente principal é uma imagem média de todas as *eigenfaces* que estão projetadas na mesma. Essas *eigenfaces* muitas vezes têm um olhar fantasmagórico, porque combinam elementos de várias faces. As regiões de *pixels* mais brilhantes e as regiões mais escuras em cada imagem são as que

mais contribuem para as componentes principais (21). Cada *eigenface* está associada a um *eigenvalue* que representa o quanto as imagens de treinamento variam da *eigenface* média naquela direção.



Figura 2.13: Direita: imagens de rosto para dez pessoas. Esquerda: os seis primeiros componentes principais, visto como *Eigenfaces* (21).

As *eigenfaces* são usadas para que a partir delas possa se estimar a distância entre a imagem que se deseja reconhecer e as imagens presentes no banco e a partir dessas distâncias dizer de qual delas a nova imagem mais se aproxima (30).

As componentes principais encontradas pelo PCA apontam para a maior variação de dados. Uma das premissas do *Eigenfaces* é que a variabilidade das imagens subjacentes corresponde à diferença entre as faces. Esta suposição nem sempre é válida. A Figura 2.14 mostra as faces de dois indivíduos apresentadas em quatro diferentes condições de iluminação (21). Na verdade, elas são imagens de faces de duas das dez pessoas mostradas na Figura 2.13. Quando a iluminação é muito variável esse algoritmo não é muito efetivo (21).

Outros fatores que podem aumentar a variabilidade da imagem em direções que tendem a diluir a identidade no espaço PCA incluem mudanças na expressão, ângulo da câmera e posição da cabeça (21).

A Figura 2.15 mostra como a distribuição de dados afeta o desempenho do *Eigenfaces*. Quando os pontos referentes as imagens de cada indivíduo ficam aglutinadas e satisfatoriamente separadas das imagens do conjunto de imagens de outros indivíduos temos o melhor caso para o funcionamento do *Eigenfaces*.

Caso os pontos referentes as imagens dos indivíduos tenham uma variabilidade muito grande, a probabilidade de choque de imagens de dois indivíduos num mesmo ponto do subespaço PCA se torna muito grande tornando extremamente difícil separar os dois indivíduos (21).

Na prática, a projeção de determinadas imagens de uma pessoa no subespaço PCA provavelmente colidirá com projeções de imagens de outras pessoas. Como as *eigenfaces*



Figura 2.14: Imagens das faces de dois indivíduos. A face de cada indivíduo é apresentada em quatro diferentes condições de iluminação. A variabilidade devido à iluminação aqui é maior do que a variabilidade entre os indivíduos. *Eigenfaces* tende a confundir as pessoas quando os efeitos de iluminação são muito fortes (21).

são determinados pela variabilidade dos dados, ficamos limitados a quanto grande deve ser essa. Podemos tomar medidas para limitar, ou para gerir de outra forma, as condições ambientais que podem confundi-lo. Por exemplo, colocar a câmera na altura do rosto irá reduzir a variabilidade no ângulo da câmera. Porém, alguns fatores são mais difíceis de controlar como as condições de iluminação, tais como iluminação lateral vinda de uma janela (21).

Mesmo com sistemas altamente ajustados, sistemas de reconhecimento facial estão sujeitos a casos de identidade equivocada (21).

Basicamente, a abordagem para o reconhecimento facial com *Eigenfaces* requer as seguintes operações de inicialização (32):

1. Adquirir um conjunto inicial de imagens para serem usadas como conjunto inicial de dados;
2. Treinar o algoritmo calculando a *eigenface* média;

Com o sistema inicializado, os seguintes passos devem ser seguidos para reconhecer novas imagens de faces (32):

1. Projetar a nova imagem na componente principal;
2. Calcular a distância entre a *eigenface* média e a *eigenface* nova;
3. Comparar com as distâncias das outras imagens e dizer se é “conhecida” ou não de acordo com o limiar de proximidade.

Após realizar os passos acima, concluindo a detecção e a identificação da face, é possível inferir uma label para a imagem com uma determinada confiança.

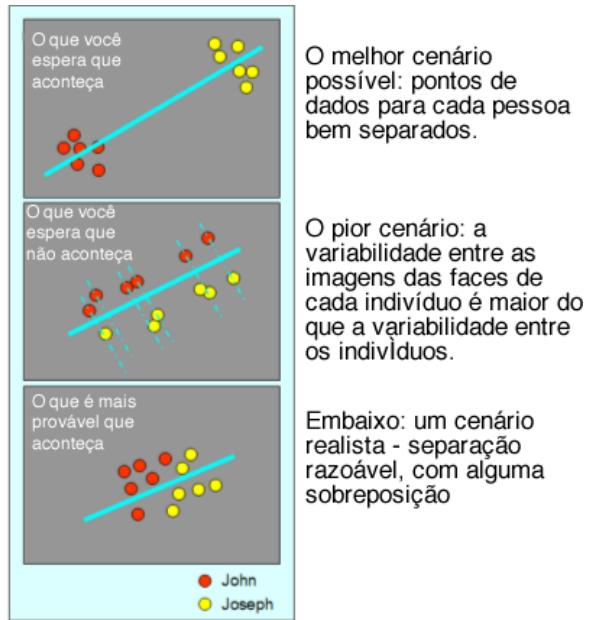


Figura 2.15: As distribuições de dados no reconhecimento com *Eigenfaces*. Adapatado de (21).

# Capítulo 3

## Trabalhos Correlatos

Neste capítulo serão analisados alguns projetos que focam rastreamento, identificação e localização de pessoas em um ambiente inteligente. Embora alguns dos projetos estudados utilizem abordagens multimodais, abordagens que usam mais de um tipo de dado de entrada como dados audiovisuais, este trabalho tem como foco identificação, localização e rastreamento monomodais baseados somente na utilização de imagens.

A seguir é apresentada uma descrição dos projetos analisados que focam em soluções para rastreamento, localização e identificação dos usuários em um ambiente inteligente utilizando abordagens baseadas em imagens.

### 3.1 Projeto CHIL

O Projeto CHIL (*Computers in the Human Interaction Loop*) (47, 51) envolve uma rede de quinze laboratórios internacionais de pesquisa acadêmica e industrial. Eles colaboram entre si conduzindo pesquisas que tem por objetivo auxiliar a pessoas de forma proativa durante suas atividades diárias no ambiente de trabalho. Em especial, o projeto foca no desenvolvimento de sistemas que auxiliam as interações entre grupos de pessoas como, por exemplo, sistemas que facilitam a colaboração em reuniões e em salas de palestras. Dentre os protótipos que foram desenvolvidos no projeto destacam-se um ambiente de trabalho perceptivo e colaborativo e um sistema perceptivo de apoio a um escritório virtual.

Este projeto utiliza informações obtidas de diferentes algoritmos monomodais para criar um sistema de rastreamento e identificação multimodal a partir de dados audiovisuais das pessoas no ambiente. Apesar do foco desta monografia ser identificação, rastreamento e localização utilizando imagens, vale salientar que o Projeto CHIL representa uma das primeiras tentativas de realizar e avaliar sistematicamente rastreamento acústico com uma rede distribuída de microfones. Os relatos a seguir se concentram somente nas investigações que utilizam dados visuais (imagens) para rastreamento e identificação.

#### 3.1.1 Rastreamento de pessoas

As pesquisas sobre rastreamento no âmbito do Projeto CHIL concentraram-se principalmente no rastreamento de pessoas dentro de um ambiente inteligente. Neste contexto,

o rastreamento tem o objetivo de determinar, de maneira contínua, as coordenadas dos ocupantes na imagem capturada.

Os sensores utilizados no ambiente inteligente do Projeto CHIL incluem:

- um mínimo de quatro câmeras fixas instaladas nos cantos do ambiente, com campos de visão sobrepostos;
- uma câmera com grande ângulo de visão fixa com vista para o ambiente inteiro;
- três arrays de microfones em forma de T de 4 canais cada;
- um microfone de Mark III de 64 canais.

Esperava-se que essa grande quantidade de sensores disponíveis fôsse uma vantagem, tendo em vista a alta redundância nas informações capturadas e boa cobertura do ambiente minimizando assim problemas como oclusão. Entretanto, foi relatado que tal redundância se tornou um grande desafio, pois surgiram problemas relativos à sincronização dos dados, transferência de processamento distribuído, fusão de espaço-temporal, entre outros.

Inicialmente, os sistemas do Projeto CHIL eram de uma única modalidade com inicialização manual, utilizando recursos simples e rastreamento de no máximo uma pessoa. Estes sistemas evoluíram para um sistema totalmente automático, com auto-inicialização, em tempo real, utilizando uma combinação de recursos capaz de rastrear alvos múltiplos.

Sobre os algoritmos de rastreamento visual, duas abordagens principais foram seguidas pelos vários sistemas de rastreamento desenvolvidos no Projeto CHIL:

1. Abordagem baseada em modelos, em que a redundância entre as câmeras são exploradas matendo um modelo 3D da pessoa rastreada e renderizando-o nos pontos de vista das câmeras. Os dados obtidos de cada câmera são utilizados para atualizar os parâmetros do modelo 3D a cada imagem (3, 5, 28).
2. Abordagem orientada a dados, onde sistemas de rastreamento 2D operam de forma independente sobre os diferentes ângulos de visão das câmeras e os dados do rastreamento pertencentes a um mesmo alvo são coletados no formato de um rastreamento 3D (50, 56). A elegância desta abordagem reside na sua capacidade inerente de lidar com os problemas encontrados por várias técnicas de rastreamento 2D como, por exemplo, oclusão.

Em termos de desempenho, foi observado que a abordagem baseada em modelos (1) geralmente prevê uma melhor acurácia, porém menos precisão do que a abordagem orientada a dados (2). Por outro lado, o tratamento das oclusões e da associação dos dados dos sistemas de rastreamentos independentes são as desvantagens do modelo orientado a dados. Foi identificado, no Projeto CHIL, que rastrear rostos ao invés de corpos inteiros diminui o impacto desses problemas.

Também foi observado que a abordagem baseada em modelos facilita a incorporação de diversos tipos de recursos, como segmentos de primeiro plano, histogramas de cor, etc., que aumentam a robustez do rastreamento. Contudo, as dificuldades encontradas nesta abordagem se concentram na inicialização automática e na atualização dos modelos das pessoas.

Os testes do sistema de rastreamento desenvolvido no Projeto CHIL foram feitos utilizando os dados dos seminários e reuniões do próprio projeto.

### 3.1.2 Identificação de pessoas

A fim de realizar a identificação de pessoas de forma natural e implícita em um ambiente inteligente, sensores distribuídos no ambiente devem monitorar continuamente o espaço de modo que o sistema de identificação opere em segundo plano sem necessitar de atenção e cooperação dos usuários.

Para a identificação de pessoas, o Projeto CHIL elegeu o reconhecimento facial, tendo em vista que a face é uma característica biométrica que permite uma identificação de forma natural e implícita.

Na tentativa de realizar o reconhecimento facial das pessoas presentes no ambiente inteligente, vários desafios foram encontrados no Projeto CHIL, como: grande variação da iluminação, sensores com baixa resolução, oclusão visual. Além disso dependendo da localização e distância dos sensores da pessoa a ser identificada os dados recebidos podem variar. Foi observado, também, que o fato das pessoas possuírem diferentes expressões faciais, diferentes cortes de cabelo, maquiagem, entre outros, dificultava ainda mais a tarefa. Entretanto, apesar de todos esses desafios, foi notado que o reconhecimento facial podia ser realizado de forma robusta utilizando múltiplos sensores no ambiente.

O sistema de identificação do Projeto CHIL utiliza sequências de imagens fornecidas pelas várias câmeras no ambiente inteligente. A cada  $200ms$  imagens das caixas delimitadoras das faces<sup>1</sup> e posições dos centros dos olhos são extraídas. O canto inferior direito da Figura 3.1 exemplifica a imagem da face extraída de uma pessoa no ambiente.

As faces extraídas são, então, alinhadas utilizando os centros dos olhos ou as caixas delimitadoras. Para obter robustez e minimizar erros, o sistema gera algumas imagens adicionais modificando as posições dos rótulos do centro dos olhos ou os rótulos das caixas delimitadoras das faces alterando, então, o alinhamento das faces nas imagens.



Figura 3.1: Caixa delimitadora e posição do centro dos olhos no sistema de identificação facial do Projeto CHIL (47).

No Projeto CHIL testou-se diferentes abordagens para reconhecimento facial. Uma delas realiza reconhecimento baseado em aparência utilizando transformada discreta de

<sup>1</sup>Caixas delimitadoras são uma maneira de apontar a localização da face na imagem por meio de um enquadramento.

cosseno (DCT - *Discrete Cosine Transform*) (13, 15). Abordagens baseadas em PCA (*Principal Component Analisys*) também foram testadas, baseadas em uma versão modificada da distância euclidiana com pesos (14, 45) como medida da distância entre imagens. Foi testado também uma abordagem baseada em análise discriminante linear (LDA - *Linear Discriminant Analysis*), porém sem sucesso pois as imagens das faces obtidas não eram linearmente separáveis (14, 45).

Todas as abordagens testadas utilizam um classificador do “vizinho” mais próximo. As decisões obtidas dos vários pontos de vista das várias câmeras utilizando tal classificador são, então, combinados por meio de uma regra de soma ponderada (14, 45).

Experimentos realizados no projeto, utilizando os mesmos métodos para extração da face e normalização, mostraram que a abordagem baseada em aparência utilizando DCT apresentou os melhores resultados. Além disso, foi observado que selecionando somente as imagens frontais de faces, ao invés de todas as amostras disponíveis, como imagens de perfil, reduz a performance do sistema de reconhecimento.

## 3.2 UPC Smart Room

Salah et al (42) propõe um sistema que realiza detecção de movimento, rastreamento de pessoas, reconhecimento facial, identificação baseado em características, localização baseada em áudio e módulos de identificação baseado em áudio. Além disso, o trabalho realiza a fusão de todas essas informações utilizando filtro de partículas para obter um sistema robusto de identificação e localização. O sistema foi projetado para operar de uma maneira completamente automática, ou seja, sem intervenção do usuário.

O projeto utiliza diferentes métodos para prover serviços de reconhecimento e rastreamento multimodal no ambiente inteligente. A Figura 3.2 mostra o fluxo das informações no sistema multimodal: a identificação e localização multimodal utiliza informações de identificação e localização por meio de áudio e informações de identificação e rastreamento utilizando imagens. O objetivo do sistema é identificar cada usuário ao entrar pela porta e rastreá-lo no ambiente.

Os *streams* de dados são processados com ajuda de um *middleware* genérico cliente-servidor chamado *SmartFlow*, resultando em uma arquitetura portável para diferentes plataformas. Este *middleware* permite transporte de grande quantidade de dados de sensores para algoritmos de reconhecimento em nós distribuídos na rede.

Os sensores utilizados e as condições do ambiente inteligente são (Figura 3.3):

- quatro câmeras nos cantos da sala (rotuladas como Cam1 a Cam4 na Figura 3.3);
- uma câmera *zenithal fish-eye* no telhado (rotulada como Cam8 na Figura 3.3);
- uma câmera ativa apontada e com zoom para a porta de entrada para capturar as faces das pessoas que entram na sala (rotulada como PTZ na Figura 3.3);
- um *array* de microfones NIST Mark III de 64 canais ;
- três *clusters* de microfone de 4 canais no formato de T;
- oito microfones no teto.



Figura 3.2: Fluxo da informação na arquitetura do ambiente inteligente (42).

A seguir serão descritas a detecção de movimento, rastreamento, detecção e reconhecimento facial desse projeto que são realizados sobre as imagens coletadas pelas câmeras no ambiente. Embora o sistema do projeto também realize a localização dos usuários no ambiente, a técnica utilizada amostras de áudio como dados de entrada que está fora do escopo deste trabalho.

### 3.2.1 Detecção de Movimento e Rastreamento

A detecção de movimento implementada tenta separar o “primeiro plano” do “fundo”. O método que o projeto utiliza é baseado na detecção de objetos em movimento sob a suposição que imagens em uma cena sem objetos em movimento mostram regularidades, que podem ser modeladas utilizando métodos estáticos. O conjunto de treinamento é construído por sequências pequenas de gravações *offline* feitas da sala vazia.

Para realizar o rastreamento foi utilizado a abordagem de um mapa de ocupação probabilística (POM - *probabilistic occupancy map*) (18) simplificado para ambientes internos, onde as trajetórias de movimentos são curtas e menos frequentes quando comparadas com trajetórias em ambientes externos. Somente as quatro câmeras nos cantos do ambiente são utilizadas.

Nesta abordagem, o mapa de ocupação é utilizado para projetar a imagem de um esboço de uma pessoa (um retângulo simples) em cada visão das câmeras. As sobreposições entre os esboços e as detecções de movimento na imagem entre as várias câmeras indicam a presença de uma pessoa. A Figura 3.4 ilustra a distribuição da probabilidade de ocupação produzida pela solução implementada.



Figura 3.3: A configuração do ambiente inteligente (42).

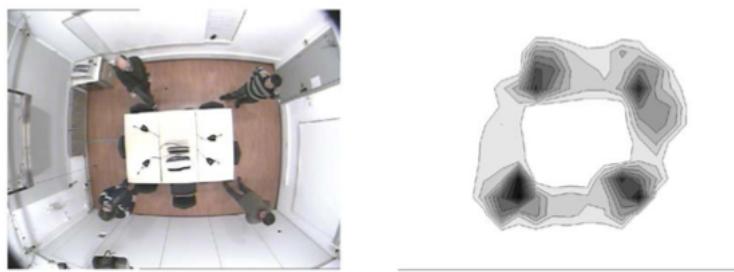


Figura 3.4: Mapa de ocupação probabilística. Adaptada de (42).

### 3.2.2 Detecção e Reconhecimento Facial

A detecção de face implementada no projeto utiliza o método *Viola-Jones* e a biblioteca *OpenCV* (7) (*Open Source Computer Vision*). Somente a câmera que está apontada para a porta é utilizada, pois esta é a única que fornece imagens na qualidade necessária. As demais, fornecem imagens em baixa resolução.

O módulo de detecção detecta somente faces frontais, em que os dois olhos, o nariz e a boca são visíveis. O resultado da detecção de face é uma imagem contendo uma caixa delimitadora da face detectada.

O reconhecimento facial no ambiente inteligente utiliza uma técnica que aproveita a vantagem do ambiente ser constantemente monitorado e combina a informação de várias imagens. Para cada sequência de imagens, as faces de um mesmo indivíduo são agrupadas. Então, para cada grupo, o sistema compara as imagens com uma galeria de imagens. Para cada grupo de imagens, as decisões individuais são combinadas em uma decisão única para o grupo.

Uma abordagem baseada em PCA (*Principal Component Analysis*) foi utilizada para comparação entre as imagens, pois possui uma baixa complexidade computacional necessária para aplicações em tempo real (42).

### 3.3 Projeto DIVA

Trivedi et al (49) apresenta detalhes de um projeto de pesquisa de longo prazo, onde ambientes inteligentes com funcionalidades úteis são projetados, construídos e avaliados sistematicamente. Neste projeto foi desenvolvido uma nova arquitetura para ambientes inteligentes chamado DIVA. Esta nova arquitetura pode ser vista como uma rede inteligente de câmeras, que são controladas para prover diversas funcionalidades, tais como, detecção de intrusos, rastreamento de várias pessoas, pose do corpo e análise de postura, identificação de pessoas, modelagem do corpo humano e análise do movimento.

O sistema proposto por este projeto monitora o ambiente em baixa resolução de forma contínua, detectando somente a presença e suas dimensões. Formas de aquisição de imagens mais detalhadas são ativadas quando um evento ou atividade de potencial interesse é detectado. Esses eventos são os focos de atenção do sistema.

O monitoramento de baixa resolução e de grande área do ambiente é alcançado graças a algumas câmeras de amplo ângulo de visão e estáticas. Com um pequeno número de câmeras PTZ (*pan/tilt/zoom*) ativas, múltiplos focos de atenção podem ser mantidos de forma simultânea.

O Projeto possui dois ambientes inteligentes separados fisicamente, porém conectados. O primeiro, chamado de AVIARY, foi projetado para ser uma pequena sala de conferências. O segundo, chamado de MICASA, foi projetado para ser uma pequena sala de aula. A Figura 3.5 ilustra os dois ambientes e a disposição dos sensores neles utilizados.

Os sensores no ambiente inteligente AVIARY incluem:

- uma rede de quatro câmeras omnidirecionais;
- quatro câmeras PTZ;
- quatro câmeras retilíneas estáticas;
- dois *arrays* de microfones com quatro microfones cada.

As quatro câmeras omnidirecionais (ODVSs - *Omni-Directional Vision Sensors*) estão perto dos cantos de uma mesa de reunião, cobrindo a sala inteira de dentro para fora, como mostrado na Figura 3.5. As câmeras PTZ e retilíneas estão nos “vértices” da sala a 1.4m acima do chão. Os dois *arrays* de microfones foram instalados na parede e no teto.

O ambiente inteligente MICASA é duas vezes maior que o AVIARY. Os sensores utilizados são:

- uma rede de quatro câmeras omnidirecionais;
- quatro câmeras PTZ;
- oito câmeras retilíneas estáticas.

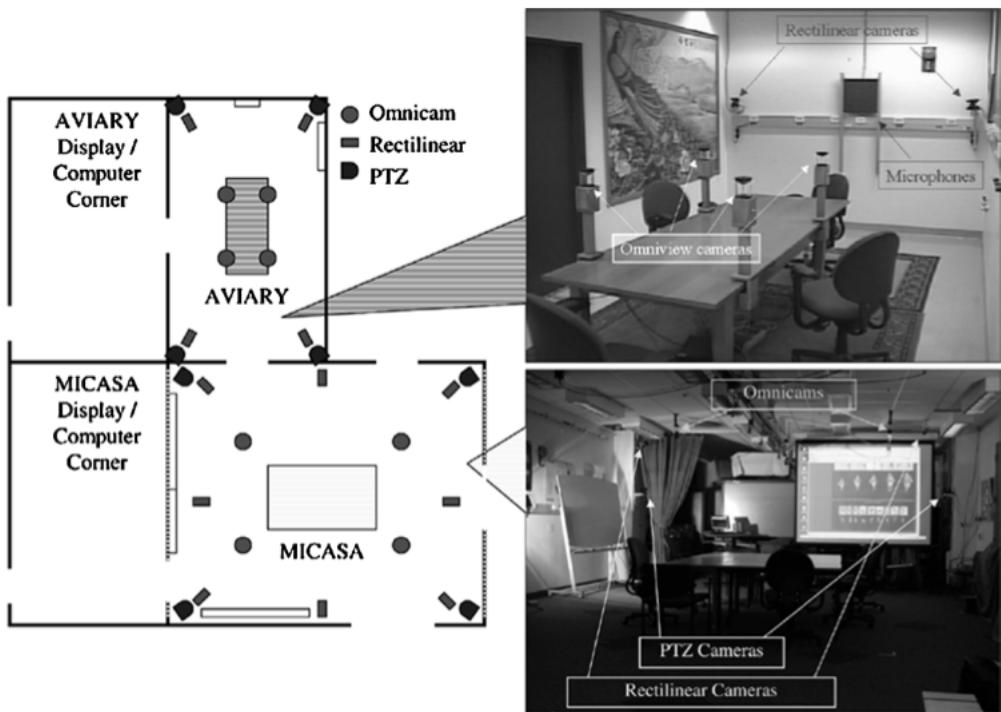


Figura 3.5: Representação dos ambientes inteligentes MICASA e AVIARY (49).

As câmeras omnidirecionais são instaladas no teto, como mostrado na Figura 3.5. As câmeras PTZ e quatro câmeras retilíneas foram instaladas de maneira similar ao ambiente AVIARY. As quatro câmeras retilíneas restantes foram instaladas nas paredes como mostrado como mostrado na Figura 3.5. As câmeras nos vértices possuem maior campo de visão para cobrir toda a sala e fazem parte do array de câmeras para rastreamento.

### 3.3.1 Rastreamento e Reconhecimento facial

Neste projeto foi desenvolvido um sistema em tempo-real que utiliza a rede de câmeras omnidirecionais e é responsável pelo rastreamento e reconhecimento facial.

O rastreamento é feito utilizando subtração do fundo com remoção de sombras para detectar as silhuetas dos novos usuários no ambiente. Com isso, essas silhuetas são rastreadas imagem por imagem.

Os resultados do rastreamento são utilizados para controlar o reconhecimento facial. As imagens obtidas do rastreamento são processadas utilizando um método de segmentação de tom de pele e são recortadas para obter as imagens com possíveis faces.

Essas imagens, então, são classificadas para rejeitar as imagens sem faces. Para essa classificação é utilizado o método *Eigenfaces*. Este último também é utilizado para realizar o reconhecimento facial utilizando a classificação do “vizinho” mais próximo.

O treinamento do algoritmo *Eigenfaces* implementado é feito utilizando um banco com 200 imagens de faces de várias pessoas diferentes.



Figura 3.6: Método de detecção e reconhecimento facial (49).

# Capítulo 4

## Sistema TRUE

Observando a realidade de um ambiente inteligente fica claro que as informações como posição das pessoas e suas respectivas identidades são de grande valia para que decisões possam ser tomadas. Atualmente, a maioria das soluções encontradas para fornecer esse tipo de informação foram projetadas para funcionar em ambientes rigidamente definidos. Com isso, não seria adequado tentar incorporar soluções como estas em um ambiente com diferentes dimensões, condições de iluminação, posição dos móveis, diferentes sensores pois este resultaria em um cenário diferente. Além disso, seria interessante que a solução fosse integrada com o middleware *UbiquitOS* (Seção 4.1), disponibilizando, então, as informações obtidas dos usuários as diversas aplicações presentes no ambiente.

Esse trabalho propõe, então, um sistema aberto de rastreamento, localização e identificação de pessoas em um ambiente inteligente integrado com o middleware *UbiquitOS*. Tal sistema é chamado de TRUE, *Tracking and Recognizing Users in the Environment*.

O Sistema TRUE é um sistema monomodal implementado em C++ que utiliza somente dados visuais, como imagens de cor e profundidade. As imagens de profundidade serão utilizadas no rastreamento e localização dos usuários no ambiente, e as imagens de cor serão utilizadas no reconhecimento facial e no cadastro dos usuários. Portanto, os dispositivos presentes no ambiente devem ser capazes de fornecer esses tipos de dados a um taxa e qualidade adequada.

Para obter os dados necessários, o sistema utiliza o sensor *Kinect* da Microsoft descrito no Apêndice A, um dispositivo bastante acessível e capaz de fornecer imagens de cor e de profundidade sincronizadas a uma taxa e qualidade necessária.

O Sistema TRUE é dividido em quatro módulos principais:

- **Módulo de Rastreamento:** parte do sistema responsável pelo rastreamento e localização dos usuários no ambiente.
- **Módulo de Reconhecimento:** parte do sistema responsável por identificar os usuários rastreados.
- **Módulo de Registro:** parte do sistema responsável pelo cadastro de novos usuários e treino do sistema.
- **Módulo de Integração:** parte do sistema responsável pela integração e comunicação do sistema com o Middleware *UbiquitOS*.

A relação entre os módulos é descrito na Figura 4.1 e descrita nas seções deste capítulo.

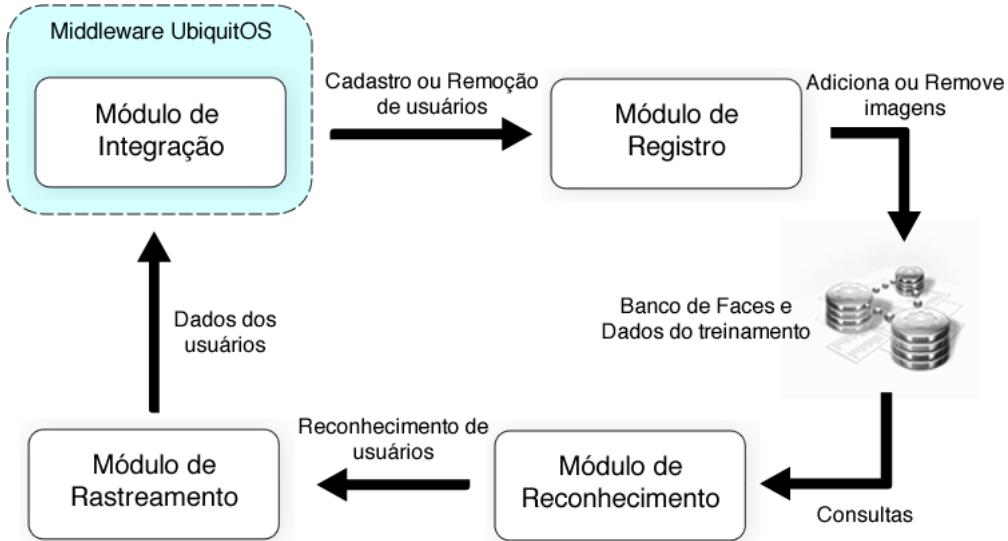


Figura 4.1: Módulos do Sistema TRUE.

## 4.1 Middleware *UbiquitOS*

O Middleware *UbiquitOS*<sup>1</sup> é um projeto do grupo de pesquisa *UnBiquitous* da Universidade de Brasília cujo foco está na adaptabilidade de serviços.

O *UbiquitOS* é uma implementação de um middleware para auxílio de aplicações para ambiente de computação ubíqua, cuja proposta é compatível com os conceitos apresentados pela *DSOA* (*Device Service Oriented Architecture*) (8). Este é visto como um componente de auxílio para desenvolvimento de drivers de recurso, aplicações e *plugins* de rede a serem utilizados em ambientes ubíquos (8).

Um recurso é um grupo de funcionalidades de um dispositivo logicamente relacionadas acessíveis através de interfaces pré-definidas. Tais funcionalidades, por sua vez, são representadas no ambiente através de serviços relacionados (8).

Uma aplicação é a implementação de um conjunto de comportamentos e regras relacionadas ao ambiente inteligente, cujo o objetivo é a tomada de ação ou a interação junto ao usuário. As aplicações ficam no dispositivos do ambiente e se utilizam dos recursos e serviços do mesmo durante a execução (8).

## 4.2 Módulo de Reconhecimento

O Módulo de Reconhecimento é responsável pela identificação dos usuários no ambiente utilizando a face como característica biométrica. A face foi escolhida pois ela permite um reconhecimento não intrusivo, como mencionada na Seção 2.3.1. A detecção e o reconhecimento são feitos em imagens de usuários que são passadas pelo Módulo de Rastreamento. Tais imagens são compostas somente pela região em que o usuário se encontra, como mostrado na Figura 4.2.

<sup>1</sup>Este trabalho é parte integrante do projeto *UbiquitOS*

Basicamente, o processo de reconhecimento é realizado pelas seguintes etapas e ilustrado na Figura 4.3:

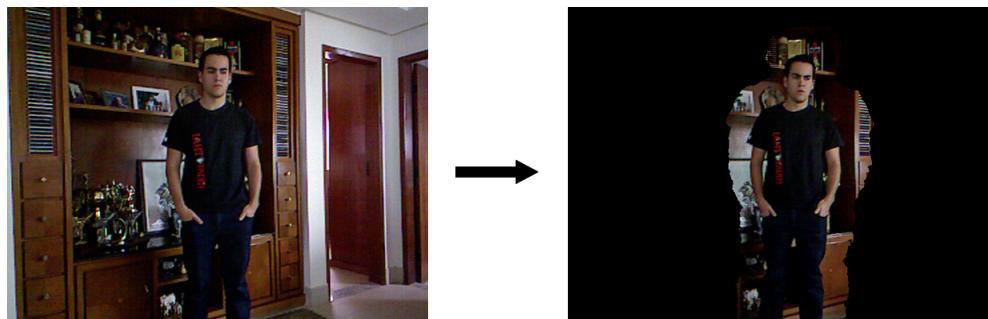


Figura 4.2: Exemplo de uma imagem composta somente pela região em que o usuário se encontra.

1. Obtém a imagem de entrada correspondente a imagem formada somente pelo usuário cujo reconhecimento foi requisitado.
2. Pré-processamento da imagem: a imagem é convertida em escala de cinza.
3. Realiza detecção facial na imagem. Caso nenhuma face seja encontrada, retorna “vazio”. Vale ressaltar que no máximo uma face pode ser encontrada nesta imagem.
4. Processamento da imagem: uma nova imagem é criada recortando a região da face encontrada, a imagem, então, é redimensionada e equalizada criando assim um padrão de tamanho, brilho e contraste nas imagens aumentando a acurácia do reconhecimento.
5. Reconhecimento facial com *Eigenfaces* é realizado.
6. Retorna o nome da face “mais parecida” e a confiança do reconhecimento.

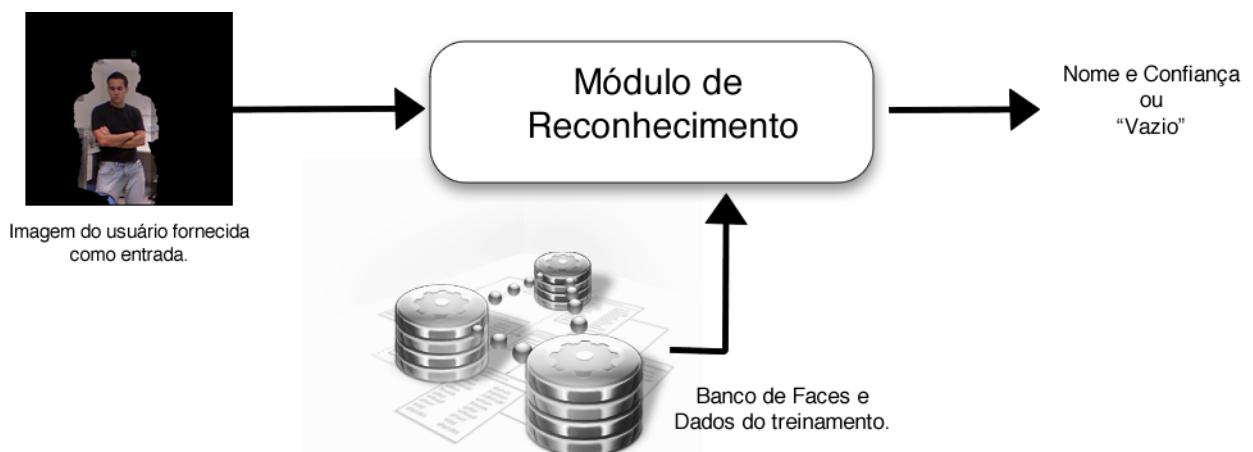


Figura 4.3: Módulo de Reconhecimento do Sistema TRUE.

#### 4.2.1 Pré-processamento e Processamento da Imagem

As etapas de processamento das imagens permitem criar um padrão nas mesmas aumentando a acurácia do reconhecimento. No Sistema TRUE as etapas de processamento consistem em converter a imagem em escala de cinza, recorta-la, redimensiona-la e equaliza-la criando, assim, um padrão de cor, tamanho, brilho e contraste nas imagens.

A Figura 4.4 exemplifica uma imagem normal de uma face, depois a mesma convertida em escala de cinza e equalizada. O Apêndice B mostra trechos de código em linguagem C que implementam tais etapas.



Figura 4.4: Exemplo de uma imagem de face normal, em escala de cinza e equalizada. Adaptada de (16).

#### 4.2.2 Detecção Facial

A detecção facial foi desenvolvida utilizando o método *Viola-Jones* 2.3.2. Um método que pode ser utilizado para construir uma abordagem de detecção facial rápida e eficaz (38) em tempo real. Além disso, este método é implementado pela biblioteca *OpenCV* (7) (*Open Source Computer Vision*) onde bons classificadores em cascata de características *Haar* são fornecidos.

Basicamente, o processo de detecção facial procura por uma face em uma imagem pré-processada. Para realizar detecção facial utilizando o método *Viola-Jones* é necessário a utilização de um classificador em cascata, como mencionado na Subseção 2.3.2. Portanto, entre os diversos classificadores em cascata presentes na biblioteca *OpenCV*, foi utilizado o classificador *haarcascade\_frontalface\_alt.xml*, um classificador treinado para detectar faces frontais em imagens.

O processo de básico de detecção de faces no Sistema TRUE possui as seguintes etapas:

1. Lê um classificador treinado para detectar faces em uma image.
2. Obtém a imagem de entrada. Tal imagem é composta somente pelo usuário, como mostrado na Figura 4.2, cujo reconhecimento foi requisitado, além de estar em escala de cinza.
3. Utilizando o classificador, tentar obter uma face na imagem.
4. Retorna a região da face detectada ou retorna “vazio” caso nenhuma face tenha sido encontrada. Vale salientar que existe no máximo uma face na imagem, pois contém somente um usuário.

### 4.2.3 Reconhecimento Facial com *Eigenfaces*

O reconhecimento facial foi desenvolvido utilizando *Eigenfaces* 2.3.2, e teve como base algumas etapas descritas em (16). Consiste em uma técnica bastante satisfatória quando utilizada sobre uma base de dados (faces) relativamente grande, permitindo ao sistema inferir, das imagens suas principais características e, partindo delas, realizar o reconhecimento das imagens utilizando um número bastante reduzido de cálculos (12), permitindo, assim, um reconhecimento em tempo real.

A base de dados utilizada no Sistema TRUE é formada por imagens no formato PGM (*Portable Gray Map*) com tamanho de 92x112 pixels e em escala de cinza. A base é composta por um banco de faces de alunos de Ciência da Computação da Universidade de Brasília e por um banco de imagens de faces da Universidade de Cambridge (1), mostrado na Figura 4.5. Este último, é formado por imagens de faces de 40 pessoas diferentes. Para cada pessoa, existem 10 diferentes imagens tiradas em diferentes épocas, com diferentes condições de iluminação, com diferentes expressões faciais (olhos abertos e fechados, sorrindo e não sorrindo, entre outros) e com diferentes detalhes faciais (óculos, sem óculos).

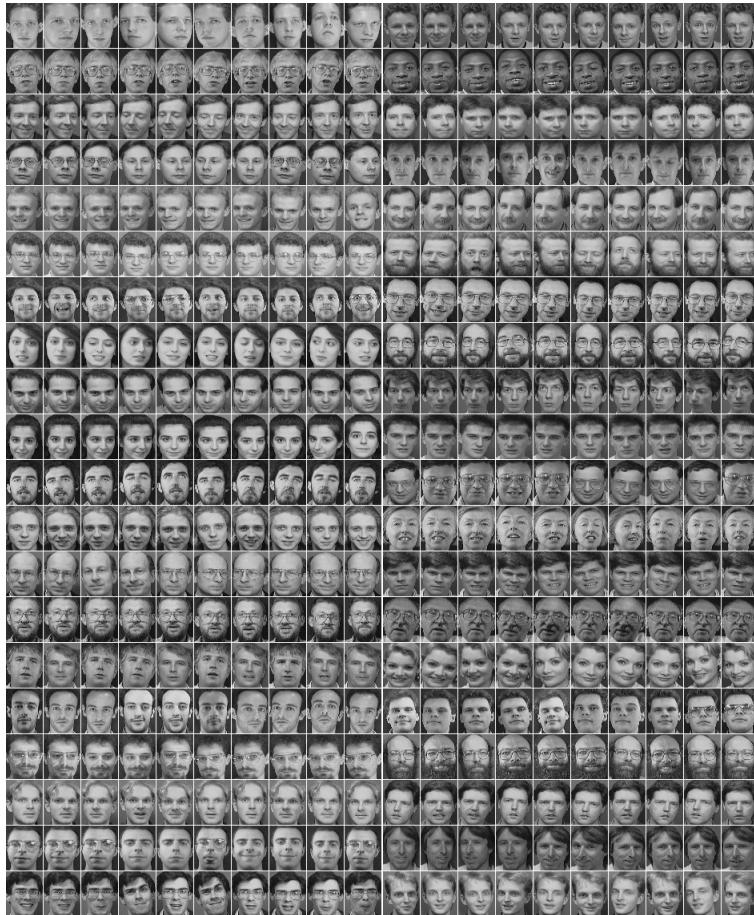


Figura 4.5: Banco de imagens de faces da Universidade de Cambridge (1).

A Figura 4.6 mostra o fluxo básico do processo de reconhecimento facial no Sistema TRUE.

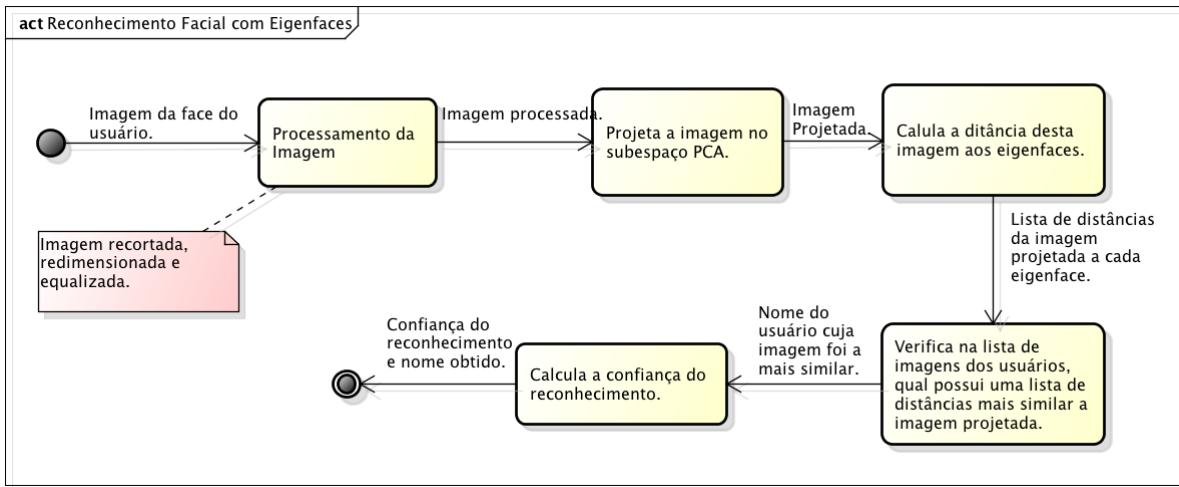


Figura 4.6: Fluxo de execução do processo de reconhecimento facial no Sistema TRUE.

Uma das etapas intermediárias consiste no cálculo da distância entre a imagem projetada no subespaço PCA aos *eigenfaces*. Inicialmente, o cálculo desta distância era feito utilizando distância Euclidiana. Contudo, testes foram realizados com alguns usuários, em que o sistema realizava 20 tentativas de reconhecimento de usuários, e os resultados não foram satisfatórios. Portanto, os mesmos testes foram realizados utilizando distância Mahalanobis. Porém, os resultados também não foram satisfatórios. Então, os mesmos testes foram feitos utilizando as duas distâncias de maneira conjunta: uma imagem só é tida como reconhecida quando o resultado das duas distâncias apontarem para a mesma identidade. Com isso, houve uma melhora significativa dos resultados. Os resultados destes testes são mostrados na Tabela 4.1, onde fica claro a melhora dos resultados quando se utiliza ambas distâncias. Nessa tabela, os nome Pessoa1, Pessoa2, Pessoa3, são nomes dados as pessoas cujas fotos estão presentes no banco de imagens de faces da Universidade de Cambridge (1).

Tabela 4.1: Resultados do teste de reconhecimento feito com o usuário Danilo utilizando as diferentes distâncias.

	Danilo	Pedro	Ana	Pessoa1	Pessoa2	Pessoa3	Desconhecido
<b>Euclidiana</b>	45%	40%		5%	5%	5%	
<b>Mahalanobis</b>	40%		15%	35%	10%		
<b>Ambas Distâncias</b>	75%						25%

A última etapa consiste no cálculo da confiança do reconhecimento. Este cálculo foi feito utilizando a distância da imagem de entrada do usuário à imagem mais similar das

imagens de treinamento. O valor da confiança varia de 0.0 a 1.0, em que uma confiança de 1.0 significaria uma “correspondência perfeita”. A fórmula utilizada para calcular a confiança é uma métrica muito básica que não necessariamente é muito real, dada pela Fórmula 4.1 (16). Nesta fórmula  $d_e$  é a distância da imagem de entrada do usuário a imagem mais similar das imagens de treinamento,  $n_t$  é o número de imagens utilizadas no treinamento e  $n_e$  é o número de *eigenfaces*.

$$Confiança = 1 - \frac{\sqrt{\frac{d_e}{n_t * n_e}}}{255} \quad (4.1)$$

### 4.3 Módulo de Rastreamento

O Módulo de Rastreamento é responsável por rastrear os usuários no ambiente, determinar a sua localização física em relação ao sensor *Kinect* e gerenciar suas identidades. Para realizar o rastreamento e localização dos usuários é utilizado a biblioteca *OpenNI* (*Open Natural Interaction*). Trata-se de um *framework* que define *APIs* para o desenvolvimento de aplicações de interação natural. Utilizando as imagens de profundidade, a detecção e o rastreamento são feito utilizando subtração de fundo (Seção 2.1.3), e os objetos detectados são representados por suas silhuetas (Seção 2.1.1).

As imagens utilizadas para o rastreamento são imagens de profundidade, exemplificada na Figura 4.7, providas pelo *Kinect* que são obtidas utilizando o método de Luz Estruturada descrito na Seção 2.2. Tais imagens de profundidade nada mais são que *depth maps* (mapas de profundidade), em que cada pixel da imagem contém o valor estimado da distância em relação ao sensor. O *Kinect* fornece esses dados a uma taxa de *30fps* (*frames* por segundo) com uma resolução *640px x 480px*.



Figura 4.7: Exemplo de uma imagem de profundidade fornecida pelo *Kinect*.

Utilizando os mapas de profundidade é possível calcular as coordenadas  $(x, y, z)$  de um plano cartesiano de três dimensões em que o ponto  $(0, 0, 0)$  corresponde a posição do *Kinect*, como mostrado na Figura 4.8. Dessa forma, a posição de um usuário em relação

ao sensor é estimada utilizando as coordenadas referente ao pixel que representa seu centro de massa geométrico. Sendo assim, ao fixar a posição do *Kinect* no ambiente, é possível estimar a localização de qualquer usuário rastreado em tempo real. A Figura 4.9 mostra um usuário rastreado pelo Sistema TRUE onde os valores das coordenadas  $(x, y, z)$  estão em milímetros.

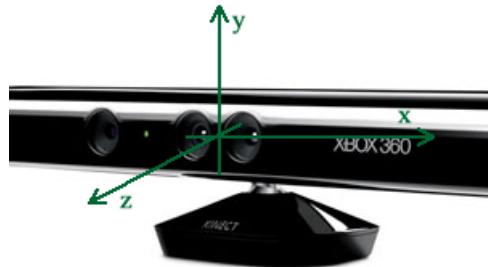


Figura 4.8: Plano cartesiano de três dimensões onde o *Kinect* representa a coordenada  $(0, 0, 0)$ .

A Figura 4.10 mostra o fluxo básico do Módulo de Rastreamento.



Figura 4.9: Imagem do Sistema TRUE de um usuário rastreado e localizado.

## 4.4 Relação Rastreamento e Reconhecimento

Até agora, foi mostrado como os Módulos de Rastreamento e de Reconhecimento funcionam de maneira isolada, mas não como se relacionam. O Módulo de Rastreamento detém as informações sobre todos os usuários rastreados no ambiente e é responsável por requisitar reconhecimento ao Módulo de Reconhecimento, que deverá acontecer quando

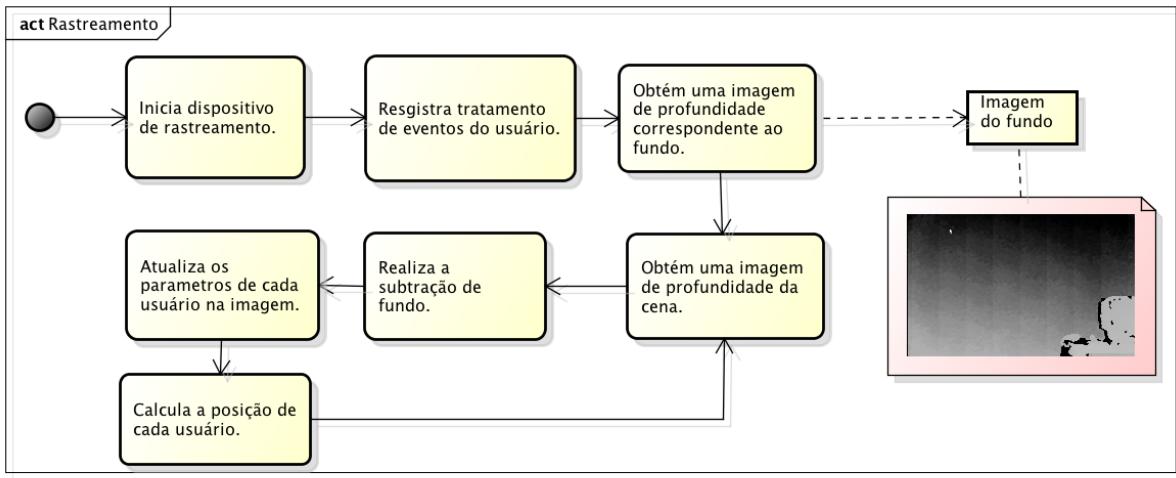


Figura 4.10: Representação das etapas propostas para o rastreamento.

um novo usuário for detectado ou quando for necessário reconhecer um usuário já rastreado.

Basicamente, quando um novo usuário for detectado, a relação entre rastreamento e reconhecimento acontecerá de acordo com as etapas descritas na Figura 4.11.

Ao invés de tentar realizar o reconhecimento somente quando novos usuários são detectados, o Sistema TRUE continua a tentar reconhecer os usuários já reconhecidos para melhorar a confiança no reconhecimento. Essas tentativas de reconhecer novamente os usuários ocorrerão em intervalos de tempo pré-definidos seguindo as mesmas etapas de quando um novo usuário for detectado. A única etapa que se difere é a primeira: ao invés de obter várias imagens de um mesmo usuário, são obtidas uma imagem de cada usuário rastreado e as mesmas são enviadas ao Módulo de Reconhecimento.

Como visto na Figura 4.11, ao obter um resultado de reconhecimento para determinado usuário, o Módulo de Rastreamento deve computar qual identidade será atribuída ao mesmo. Para isso, este módulo mantém para cada usuário o número total de vezes que já foi reconhecido, os diferentes nomes obtidos pelo Módulo de Reconhecimento bem como a confiança média para cada nome e o número de vezes que cada nome foi atribuído ao usuário. Com todos esses dados, a identidade do usuário é escolhida por meio de uma média ponderada entre as confianças de cada nome onde os pesos utilizados são compostos pelo número de vezes que o respectivo nome foi atribuído ao usuário. Vejamos o seguinte exemplo para ilustrar esse cálculo:

Seja João um usuário do ambiente inteligente que está sendo rastreado a algum tempo e que já foi reconhecido algumas vezes. No momento, o Módulo de Rastreamento mantém vários dados sobre o João descritos na Tabela 4.2. Então, para computar qual identidade será atribuída ao usuário, o Módulo de Rastreamento realiza os cálculos 4.2. Como o resultado da média ponderada se aproxima mais da confiança do nome João, este foi escolhido como sendo sua identidade.

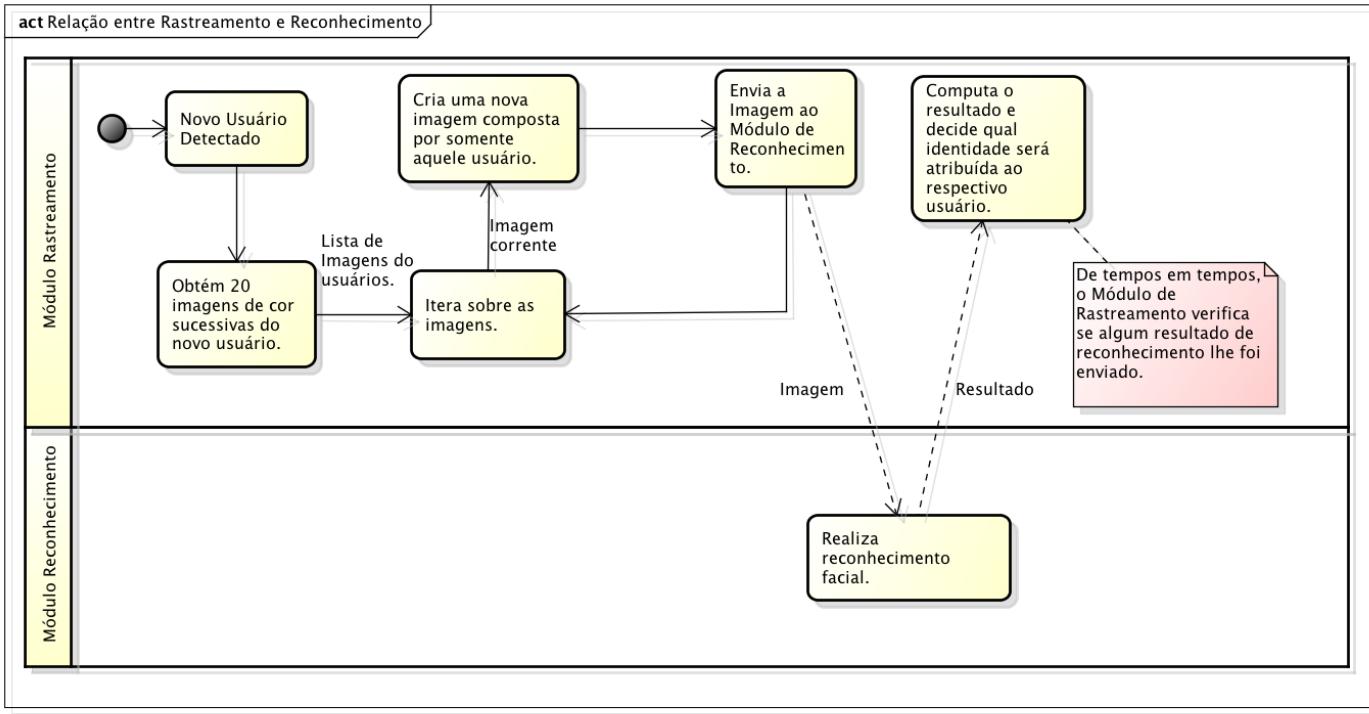


Figura 4.11: Representação da relação que o Módulo de Rastreamento terá com o Módulo de Reconhecimento quando um novo usuário for detectado.

$$M_p = \frac{15 * 0.947302 + 1 * 0.934010 + 3 * 0.950320}{19} = 0.947079 \quad (4.2)$$

$$Joao : 0.947079 - 0.947302 = -0.000223$$

$$Danilo : 0.934010 - 0.947302 = -0.013292$$

$$Tales : 0.950320 - 0.947302 = 0.003018$$

Tabela 4.2: Exemplos de dados de reconhecimento mantidos para cada usuário rastreado pelo Módulo de Rastreamento.

Nome	Confiança Média	Número de Vezes
João	0.947302	15
Danilo	0.934010	1
Tales	0.950320	3
<b>Total</b>		19

Com todos esses dados de reconhecimento, além dos dados de rastreamento e localização de cada usuário, o Módulo de Rastreamento consegue centralizar todas as informações necessárias de cada usuário. A Figura 4.12 exemplifica um usuário rastreado pelo Sistema TRUE onde as informações de localização e identificação estão presentes.



Figura 4.12: Exemplo de um usuário rastreado e identificado pelo Sistema TRUE.

## 4.5 Módulo de Registro

O Módulo de Registro é responsável por cadastrar novos usuários no sistema e treiná-lo para também reconhecer esse novo usuário. Basicamente, o processo de registro, ilustrado na Figura 4.13, segue as seguintes etapas :

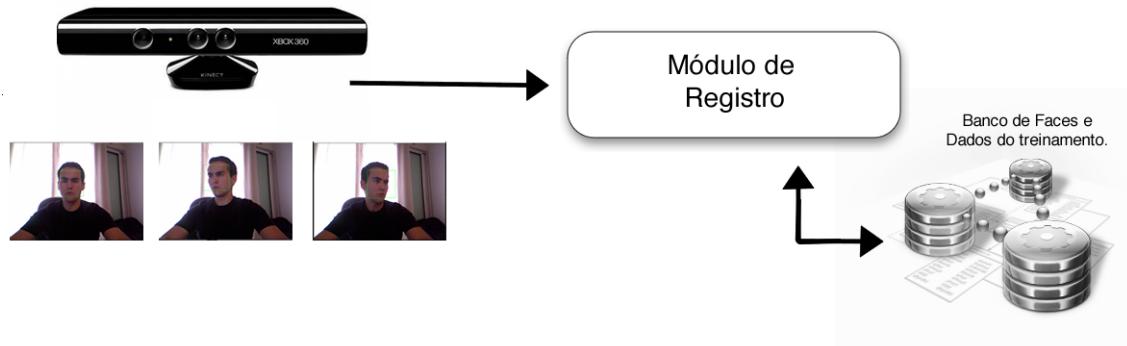


Figura 4.13: Módulo de Registro do Sistema TRUE.

1. O novo usuário fica em uma posição fixa e frontal em relação ao *Kinect*.
2. O sistema obtém seis imagens frontais do usuário. O usuário, então, deve rotacionar um pouco a face para a esquerda e o sistema obtém mais duas imagens do usuário. Depois, deve rotacionar um pouco para direita e o sistema obtém outras duas imagens do usuário.
3. As imagens obtidas são processadas: as imagens são convertidas em escala de cinza, novas imagens são criadas recortando a região da face encontrada, as imagens, então, são redimensionadas e equalizadas criando assim uma padrão de tamanho, brilho e contraste nas imagens.
4. Armazena-se as imagens.
5. O sistema é treinado para reconhecer esse usuário.

Após o treinamento, o Sistema TRUE reiniciará para que o reconhecimento seja feito utilizando as novas informações obtidas com o treinamento.

Como descrito, durante o processo de captura das imagens, o usuário deve rotacionar a face um pouco para direita e para esquerda obtendo, além de imagens frontais, imagens um pouco mais de perfil do usuário, como mostrado na Figura 4.14. Isso foi feito para que o sistema se torne um pouco mais robusto em relação a posição da face dos usuários ao tentar reconhecê-los.

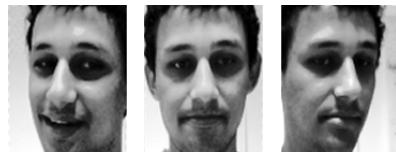


Figura 4.14: Exemplo de imagens resultantes do cadastro do usuário.

A última etapa do módulo de registro consiste no treinamento do sistema, descrito na Figura 4.15.

## 4.6 Módulo de Integração

O Módulo de Integração é responsável pela integração do Sistema TRUE com o Middleware *UbiquitOS*. Os dados providos pelo Sistema TRUE (localização e identificação dos usuários em um ambiente) são importantes informações de contexto para ambientes inteligentes. O Middleware *UbiquitOS* foi escolhido como meio de fornecê-las as diversas aplicações no ambiente. Então, para integrá-los foi desenvolvido um driver para o middleware se comunicar com o sistema. Tal driver foi nomeado de *UserDriver* cujo diagrama de classe é mostrado na Figura 4.16. Tal integração foi feita utilizando suporte JNI (*Java Native Interface* - Apêndice C) devido as diferentes plataformas: o middleware foi desenvolvido em JAVA e o Sistema TRUE em C++.

Os métodos do *UserDriver* podem ser divididos basicamente em três grupos: métodos de inicialização, métodos nativos e de serviços e eventos:

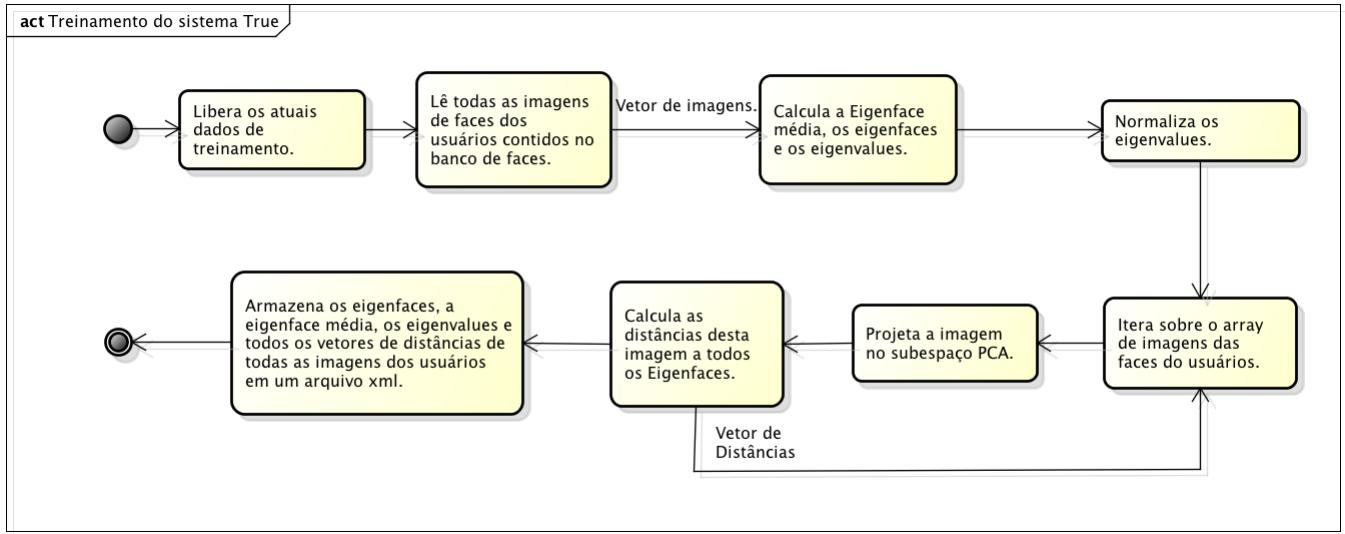


Figura 4.15: Fluxo básico do treinamento do Sistema TRUE.

- **Métodos de Inicialização:** métodos responsáveis pela inicialização do Sistema TRUE e pelo registro dos serviços que o driver possui e os eventos que pode gerar.
- **Métodos Nativos:** métodos declarados no driver que permitem acessar alguns métodos implementados pelo Sistema TRUE. Tais métodos permitem que o *UserDriver* tenha controle sobre o sistema, podendo iniciá-lo, encerra-lo, retreina-lo e, até mesmo, cadastrar e remover novos usuários a qualquer momento. Para poder implementar os métodos nativos foi utilizado JNI (*Java Native Interface*), descrito no Apêndice C.
- **Serviços e Eventos:** são os métodos que implementam os serviços que o *UserDriver* disponibiliza as aplicações presentes no ambiente e os eventos que pode gerar.

O *UserDriver* disponibiliza as aplicações registradas no *middleware* os seguintes serviços:

- **Consultas as informações dos usuários no ambiente:** através dessas consultas, as aplicações tem acesso ao nomes, emails, posições correntes e confiança do reconhecimento de todos os usuários presentes no ambiente.
- **Cadastro:** as aplicações podem cadastrar novos usuários fornecendo ao *UserDriver* o nome, o email e as imagens do novo usuário.
- **Treino do sistema:** após cadastrar novos usuários as aplicações podem retreinar o sistema para poder reconhecer o novo usuário cadastrado.
- **Remoção:** as aplicações podem remover usuários cadastrados fornecendo o email do usuário.
- **Registro de *listeners*:** as aplicações podem registrar *listeners* para “escutar” os eventos gerados pelo *UserDriver*.

Além dos serviços, o *UserDriver* gera os seguintes eventos:

- **Novo Usuário:** evento gerado assim que um novo usuário foi detectado pelo Sistema TRUE.
- **Usuário Perdido:** evento gerado assim que um usuário deixou de ser rastreado pelo Sistema TRUE.
- **Atualização dos dados do usuário:** evento gerado a cada cinco segundos atualizando os dados de todos os usuários rastreados.

O Módulo de Integração comunica diretamente com os Módulos de Rastreamento e de Registro através do *UserDriver*. Enquanto a comunicação com o Módulo de Registro acontece somente quando alguma aplicação solicita o cadastramento ou a remoção de usuários, a comunicação com o Módulo de Rastreamento acontece constantemente:

- a cada 5 segundos o Módulo de Rastreamento envia informações correntes sobre todos os usuários no ambiente atualizando as informações que o Módulo de Integração detém;
- sempre que um usuário entra ou sai do ambiente o Módulo de Rastreamento informa o Módulo de Integração;

Através do *UserDriver*, o Middleware *UbiquitOS* consegue ter acesso as informações sobre identidade e localização dos usuários presentes no ambiente em tempo real. Tais informações ficam disponíveis a qualquer aplicação registrada no middleware.

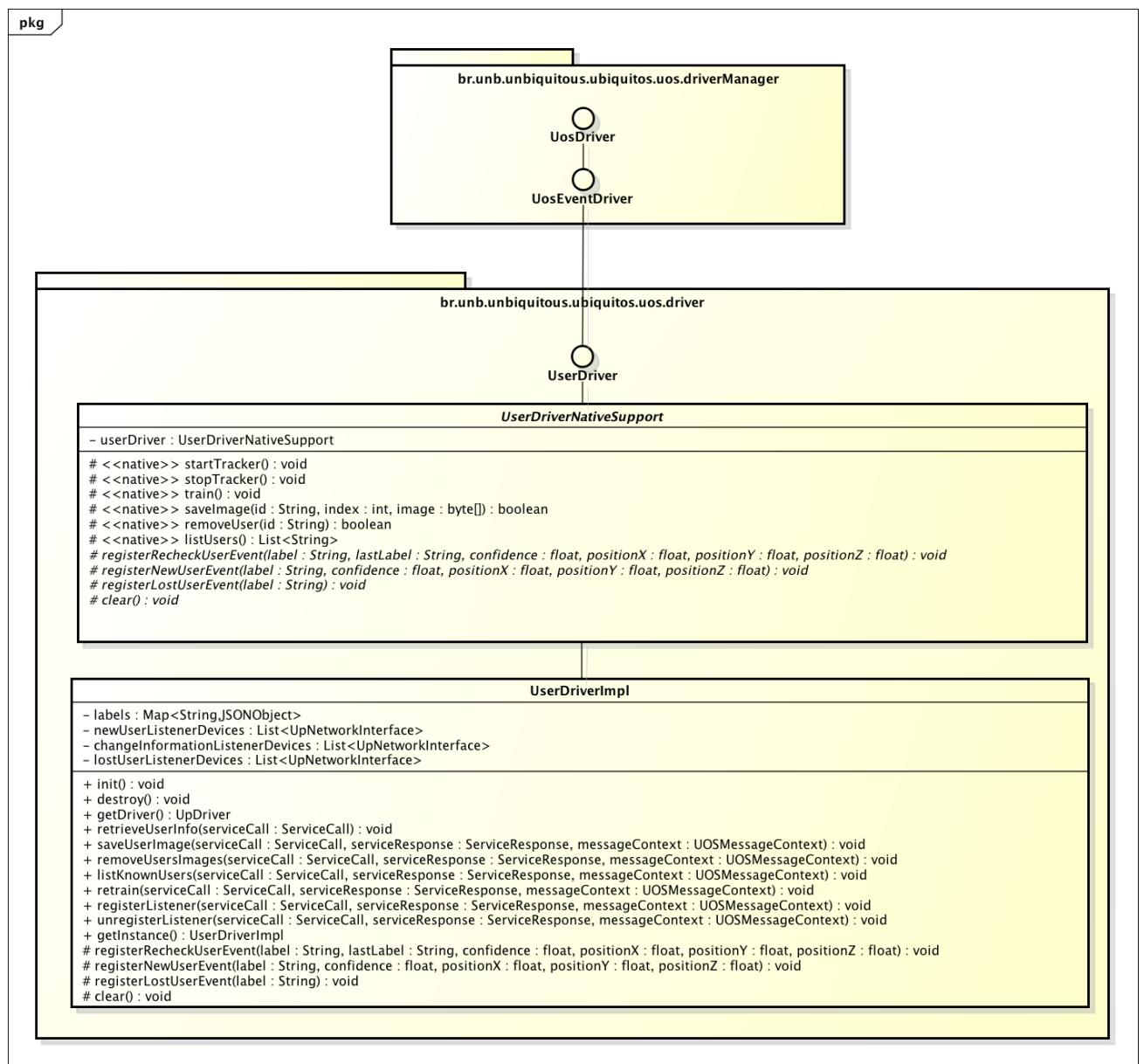


Figura 4.16: Diagrama de Classe do UserDriver.

# Capítulo 5

## Resultados e Análises

Com intuito de identificar a acurácia e as limitações do Sistema TRUE perante seus requisitos (identificação, rastreamento e localização) foram feitos uma série de testes funcionais. Grande parte dos testes foram realizados no LAICO (**L**Aboratório de Sistemas Integrados e **C**Oncorrente), um laboratório do Departamento de Ciência da Computação da Universidade de Brasília.

Cada teste tinha como foco um das funcionalidades do Sistema TRUE:

- **Rastreamento:** testes funcionais realizados para avaliar a eficiência da detecção de novos usuários e que simulam diferentes situações diárias mostrando como o sistema se comporta quando ocorre oclusões parciais e totais de usuários e quando os mesmos interagem entre si ou com objetos no ambiente.
- **Reconhecimento:** testes realizados para avaliar a acurácia na identificação do usuários perante uma base de dados.
- **Localização:** testes realizados para avaliar a precisão do sistema ao estimar a posição dos usuários no ambiente.
- **Integração com o middleware *UbiquitOS*:** uma aplicação foi desenvolvida para testar a integração do Sistema TRUE com o middleware *UbiquitOS*, testando os serviços disponíveis e os eventos gerados.

### 5.1 Rastreamento dos Usuários

O rastreamento é fundamental para o funcionamento correto do sistema uma vez que é responsável por rastrear os usuários no ambiente, determinar a sua localização física em relação ao sensor *Kinect* e gerenciar suas identidades. Portanto, foi realizado uma série de testes funcionais para determinar suas limitações.

Os primeiros testes realizados foram para testar a eficiência da detecção de novos usuários no ambiente. Os testes foram feitos simulando a entrada de um usuário na cena por diferentes ângulos e analisando o momento em que o mesmo era detectado. Em todos os testes o usuário era detectado antes mesmo de entrar na área de visão do sistema por completo, como mostrado na Figura 5.1.

Também foram realizados testes para tentar avaliar o impacto da oclusão no rastreamento. Em alguns testes um usuário se posicionava no ambiente com intuito de ocultar

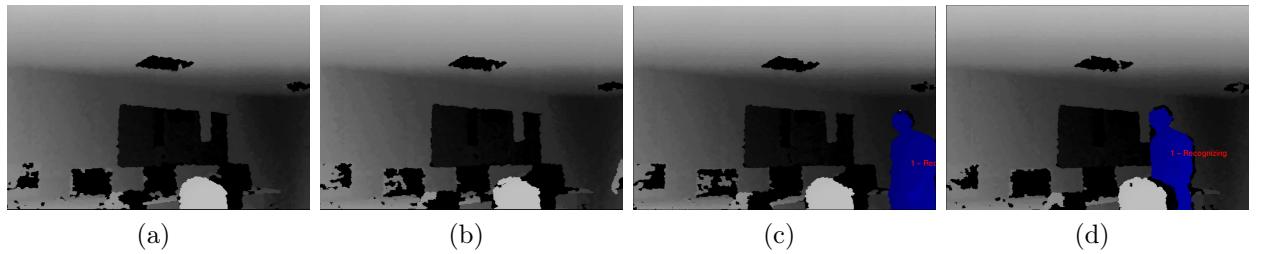


Figura 5.1: Momento em que um novo usuário foi detectado pelo Sistema TRUE.

totalmente outro usuário rastreado. Com isso, caso o usuário continuasse oculto, o sistema o dava como perdido. Porém, o sistema se mostrou robusto em casos que a oclusão era parcial, como mostrado na Figura 5.2. Em outros testes, foi simulada uma situação mais comum: quando um usuário, em movimento, ocultava ou era oculto, por um momento, por outro usuário. Neste caso, o sistema logo conseguia se recuperar e voltar a rastrear o usuário perdido, como mostrado na Figura 5.3. A oclusão era um problema esperado pois o Sistema TRUE utiliza somente um sensor *Kinect* como dispositivo de entrada.

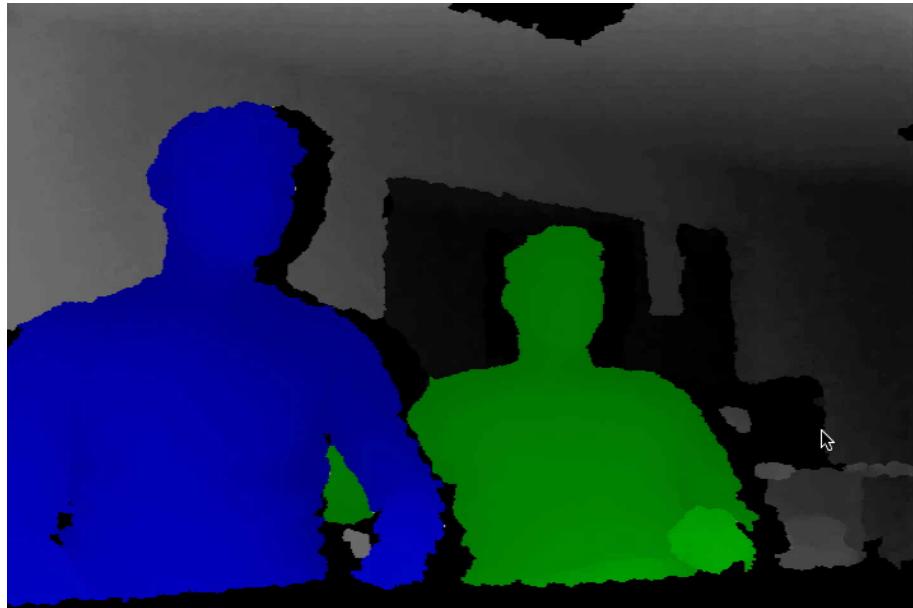


Figura 5.2: Oclusão parcial de dois usuários.

Outra característica importante que foi testada é a abrangência do campo de visão do Sistema TRUE. Como já mencionado o sistema utiliza o sensor *Kinect* que possui um campo de visão horizontal de 57°. Então, a uma distância de, aproximadamente, 4 metros do sensor, o número máximo de usuários que cabem no campo de visão sem que haja oclusão são 5 pessoas, como mostrado na Figura 5.4.

Durante os testes realizados com rastreamento foi observado alguns problemas que ocorriam quando o usuário rastreado interagia com objetos ou com outros usuários. Na grande parte das vezes que o usuário interagia com objetos, o Sistema TRUE considerava

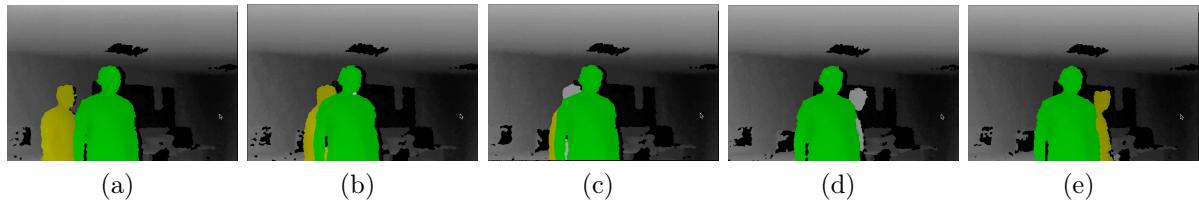


Figura 5.3: Oclusão de usuários.



Figura 5.4: Usuários posicionados lado a lado do sensor *Kinect* a uma distância de 4 metros.

o objeto como sendo parte do usuário como exemplificado na Figura 5.5, o que não prejudicou a eficiência do sistema. Já os problemas com interação entre usuários eram bem mais raros, porém o impacto era maior. Esses problemas consistem em algumas “interferências” que podiam acontecer quando havia contato entre dois ou mais usuários. A Figura 5.6 exemplifica melhor essas “interferências”.

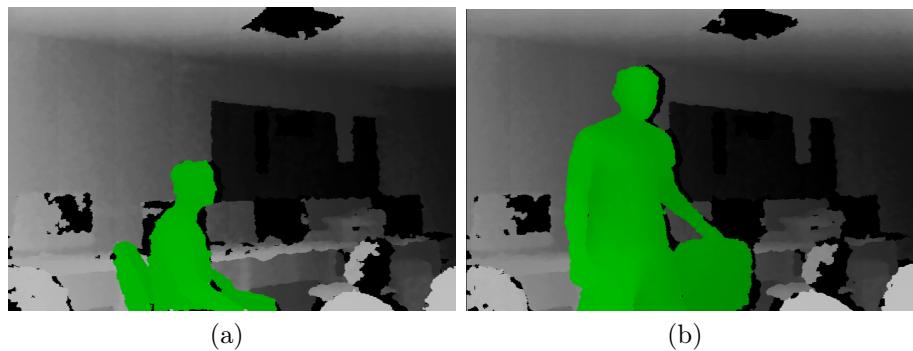


Figura 5.5: Usuários sendo rastreado conjuntamente com os objetos que interagem.

Apesar dos problemas relatados, o rastreamento conseguiu, na maioria dos testes, atender as necessidades rastreando os diversos usuários no ambiente em suas atividades diárias, como mostrado na Figura 5.7.

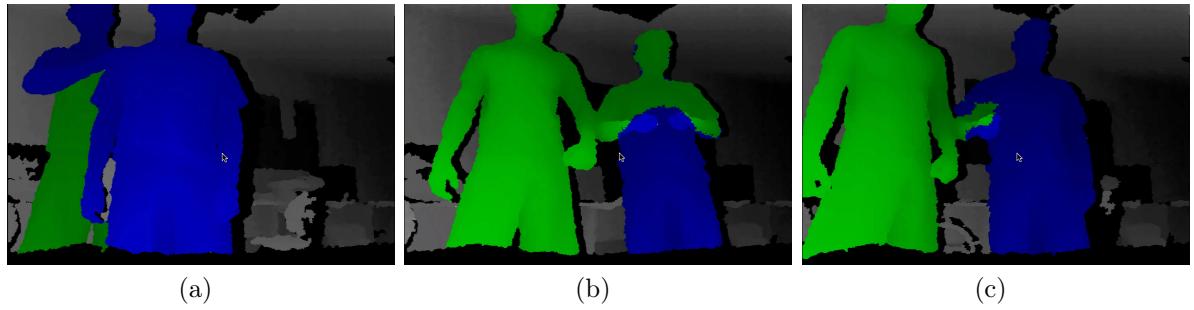


Figura 5.6: Usuários sofrendo interferência dos que estão ao seu redor.

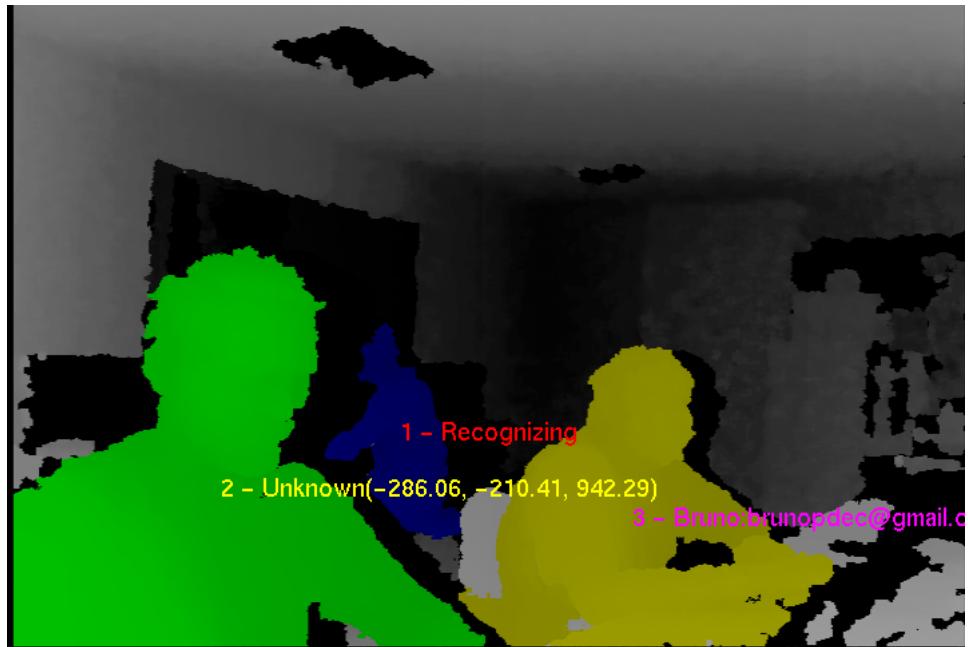


Figura 5.7: Usuários rastreados pelo Sistema TRUE.

## 5.2 Localização dos Usuários

O Sistema TRUE obtém a localização dos usuários no ambiente por meio de coordenadas dos mesmos em relação ao *Kinect*. Para saber o quanto essas coordenadas são confiáveis foram realizados alguns testes que resultaram em gráficos que compararam as coordenadas obtidas pelo sistema e as coordenadas reais.

As coordenadas  $(x, y, z)$  obtidas pelo sistema são coordenadas de um plano cartesiano de três dimensões em que o ponto  $(0, 0, 0)$  corresponde a posição do *Kinect*. A coordenada no eixo  $z$  mostra o quanto o usuário está distante, em termos de profundidade, em relação ao sensor, a coordenada no eixo  $x$  mostra o quanto o usuário está a direita ou a esquerda do sensor e a coordenada no eixo  $y$  mostra o quanto o centro de massa geométrico do usuário está acima ou abaixo do sensor.

Durante a estimativa da localização do usuário no ambiente inteligente, as coordenadas no eixo  $y$  são ignoradas por que em contextos ubíquos a localização espacial é baseada

em contextos de ambientes (sala de reunião 101, casa, patio) e não em relação a pontos cartesianos gerais. Portanto, os testes desenvolvidos aferiram somente os valores obtidos pelo Sistema TRUE nos eixos  $x$  e  $z$ .

### 5.2.1 Teste dos valores no eixo $z$

Os valores obtidos no eixo  $z$  correspondem aos valores de profundidade do usuário rastreado em relação ao *Kinect*. Portanto, para testar a precisão do Sistema TRUE foi realizado o seguinte teste: um objeto (uma caixa de papelão) foi colocada em frente ao sensor em diferentes distâncias do mesmo (1 metro, 2 metros, 3 metros, 4 metros, 4,057 metros), como mostrado na Figura 5.8. Para cada distância, foram coletados os dados informados pelo sistema (Tabela 5.1).

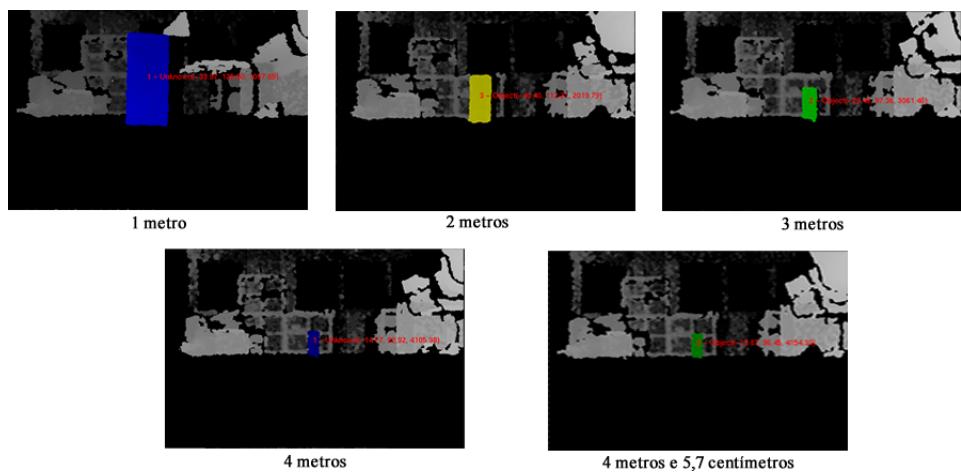


Figura 5.8: Fotos obtidas pelo Sistema TRUE no teste realizado para os valores do eixo  $z$ .

Tabela 5.1: Comparativo entre os valores reais e os valores obtidos pelo Sistema TRUE.

Distância real do objeto ao <i>Kinect</i> (mm)	Média dos valores obtidos pelo Sistema TRUE (mm)
1000,00	1006,24
2000,00	2020,62
3000,00	3059,35
4000,00	4112,99
4057,00	4166,34

Os valores obtidos no teste foram inseridos em um gráfico representado pela Figura 5.9. Como visto, a discrepância aumenta conforme o objeto se distancia do sensor. Neste teste o menor erro obtido foi de 3,21 milímetros e o maior de 11,175 centímetros.

Pelos resultados deste teste, pode-se concluir que o Sistema TRUE fornece informações sobre localização dos usuários bem precisas e confiáveis podendo ter erros de poucos centímetros que não prejudicam a integridade das informações.

Através deste teste, também foi possível obter a distância máxima e mínima que o usuário deve estar do *Kinect* para que o sistema consiga rastreá-lo e estimar sua localização. A distância mínima é de 48,3cm e a máxima de 4,057m.

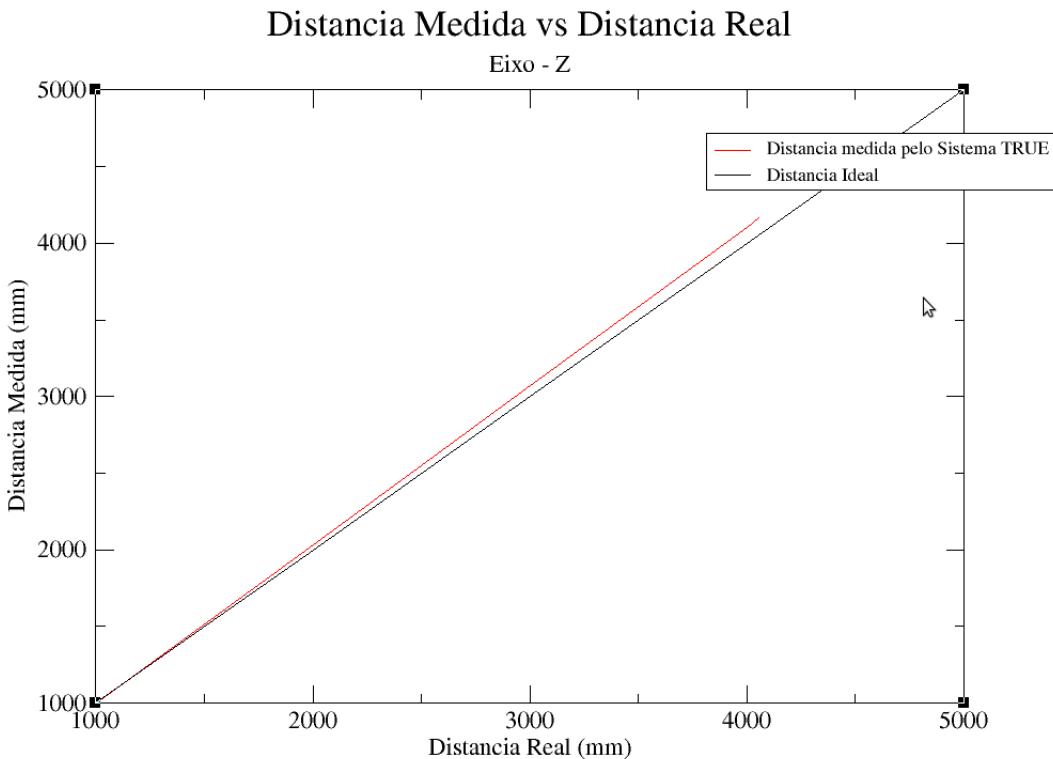


Figura 5.9: Gráfico comparativo entre os valores obtidos pelo Sistema TRUE e os valores reais.

### 5.2.2 Teste dos valores no eixo $x$

Os valores obtidos no eixo  $x$  mostram o quanto o usuário rastreado está a direita ou a esquerda em relação ao *Kinect*. Portanto, para testar a precisão do Sistema TRUE foi realizado o seguinte teste: um objeto (uma caixa de papelão) foi colocado a uma distância fixa de 3 metros do sensor e colocado em diferentes posições ao longo do eixo  $x$  ( $0, \pm 33\text{cm}, \pm 66\text{cm}, \pm 99\text{cm}$ ), como mostrado na Figura 5.10. Para cada posição, foram coletados os dados informados pelo sistema (Tabela 5.2).

Os valores obtidos no teste foram inseridos em um gráfico representado pela Figura 5.11. Como observado, existe uma diferença constante de poucos centímetros ao longo de toda a reta, ou seja, com o objeto fixo a uma distância de 3 metros do *Kinect*, o Sistema TRUE consegue fazer estimativas das coordenadas no  $x$  com erro de poucos centímetros, neste caso o menor e o maior erro obtidos foram de 27,19 mm e 79,29 mm respectivamente. Um erro tão pequeno que não prejudica a integridade dos valores obtidos pelo sistema.

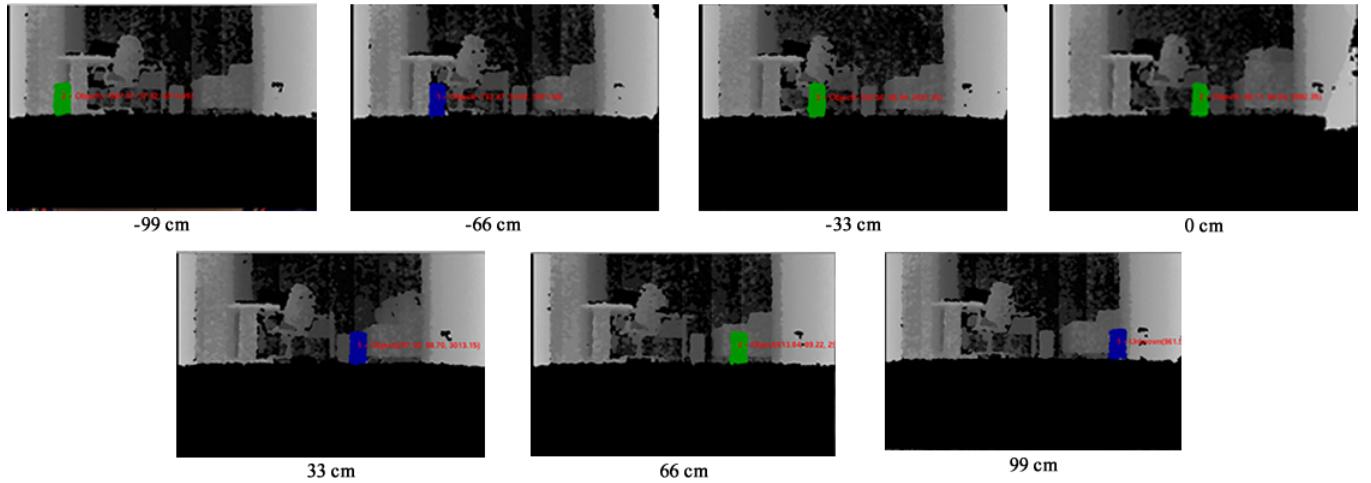


Figura 5.10: Fotos obtidas pelo Sistema TRUE no teste realizado para os valores do eixo  $x$ .

Tabela 5.2: Comparativo entre as posições reais e os valores obtidos pelo Sistema TRUE.

Posição do objeto ao longo do eixo $x$ (mm)	Média dos valores obtidos pelo Sistema TRUE (mm)
0	-32,80
0	-33,78
330	297,56
-330	-390,33
660	615,10
-660	-731,13
990	962,81
-990	-1069,29

### 5.3 Identificação dos Usuários

O reconhecimento facial do Sistema TRUE foi implementado utilizando imagens de cor como dados de entrada. Portanto, ele tem influência de diversos fatores como iluminação, ângulo, pose, expressões, cosméticos e acessórios. Portanto, para análise dos resultados foi utilizada como ferramenta uma matriz de confusão construída a partir dos dados do sistema.

Tal técnica consiste em uma matriz em que cada coluna representa as instâncias de uma classe prevista, enquanto que cada linha representa as instâncias de uma classe real. Como benefício da técnica temos a fácil visualização da confiança obtida no reconhecimento, isso pode ser observado através dos valores obtidos na diagonal principal que representa a porcentagem de acerto para cada usuário.

Nosso cenário foi construído com 11 usuários cadastrados préviamente além das 40 pessoas presentes no banco de faces da Universidade de Cambridge (1). Cada usuário se posicionava em frente ao *Kinect*, com o rosto posicionado de maneira frontal em relação

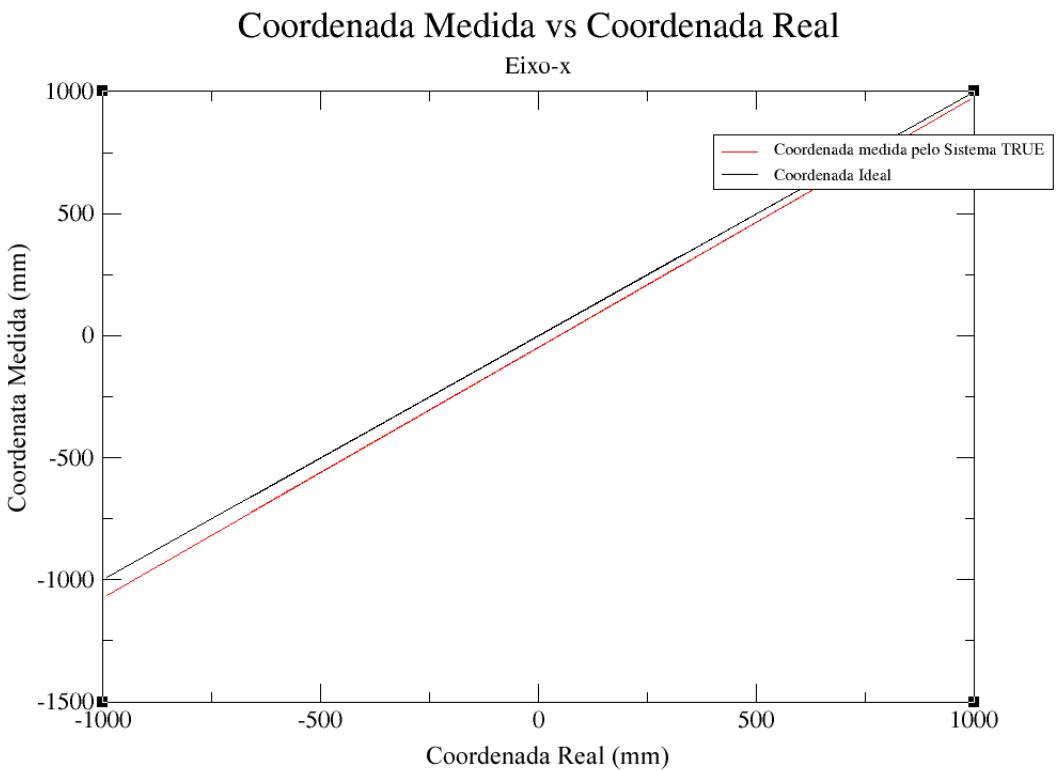


Figura 5.11: Gráfico comparativo entre os valores obtidos pelo Sistema TRUE e os valores reais.

ao mesmo, que realizava o processo de reconhecimento 20 vezes. As identidades obtidas pelo Sistema TRUE foram inseridas na matriz de confusão mostrada na Tabela 5.3.

Como visto na matriz, os maiores valores, como previsto, estão na diagonal principal. Alguns resultados obtidos foram satisfatórios já outros ficaram aquém do esperado. Estes últimos foram principalmente causados por problemas como pose e expressões faciais. A iluminação não foi um problema, pois no ambiente de teste não existe influência da iluminação externa e a interna é controlada. Para se ter uma melhor interpretação dos valores obtidos, foram retiradas três taxas desta matriz:

1. **Verdadeiro Positivo:** quando o sistema identifica o usuário de maneira correta.
2. **Verdadeiro Negativo:** quando o sistema identifica o usuário de maneira errada.
3. **Falso Negativo:** quando o sistema não identifica o usuário cadastrado.

Tais taxas foram inseridas na Tabela 5.4. A taxa de acerto (verdadeiro positivo) foi menor do que se esperava. Portanto, o teste foi realizado novamente utilizando menos usuários, 6 usuários além dos 40 presentes no banco de faces da Universidade de Cambridge (1). Porém o cadastro destes usuários foi feito de maneira diferente. Ao invés de se obter 10 fotos das faces posicionadas de maneira frontal em relação ao sensor, foram obtidas 100 imagens das faces em diferentes ângulos, posições e expressões faciais. Os resultados obtidos, foram inseridos em uma segunda matriz de confusão (Tabela 5.5).

Tabela 5.3: Matriz de confusão para apresentar os resultados obtidos.

	Tales	Danilo	Ana	Fabricio	Lucas	Bruno	Ricardo	Estevao	Rafael	Vinicio	Pedro	Desconhecido
Tales	95%											5%
Danilo		75%									25%	
Ana			95%									5%
Fabricio	20%			50%			10%			20%		
Lucas					55%						20%	25%
Bruno	5%					70%			10%	5%		10%
Ricardo			10%				85%					5%
Estevao								70%				30%
Rafael	10%	5%			10%				45%	20%		10%
Vinicio			5%						5%	70%	10%	10%
Pedro											100%	

Tabela 5.4: Taxas obtidas da Tabela 5.3.

<b>Verdadeiro Positivo</b>	73,63%
<b>Verdadeiro Negativo</b>	17,27%
<b>Falso Negativo</b>	9,10%

Ao analisar os dados da Tabela 5.5 é fácil observar que os resultados melhoraram significamente. Essa melhora é confirmada pelas taxas retiradas desta tabela e inserida na Tabela 5.6. Houve um bom aumento na taxa de Verdadeiro Positivo de 73,63% para 95,83% e redução na taxa de Verdadeiro Negativo de 17,27% para 0%. Tal melhora aconteceu devido a mudança na base de dados tornando o Sistema TRUE mais robusto com as variações de poses, ângulos e expressões faciais.

## 5.4 Integração com middleware *UbiquitOS*

Com intuito de exemplificar a utilização do *UserDriver* e testar a integração do Sistema TRUE com o middleware *UbiquitOS* foi desenvolvido uma aplicação para o middleware chamada *UserApp*. Esta aplicação registra um *listener* para “escutar” os eventos do *UserDriver* chamado *UserListener*.

Como aplicação o *UserApp* apenas se inscreve para receber os eventos gerados pelo *UserDriver*. Já o *UserListener*, como *listener*, espera os eventos serem gerados e realiza duas análises: analisa o evento cru (novo usuário detectado, usuário perdido, ou reconhecimento realizado) e analisa quanto tempo cada usuário está parado no mesmo lugar.

Tabela 5.5: Matriz de confusão para apresentar os resultados obtidos.

	Tales	Danilo	Ana	Fabricio	Lucas	Bruno	Carla	Marcela	Caio	Marcelo	Desconhecido
Tales	100%										
Danilo		100%									
Ana											
Fabricio											
Lucas				85%							15%
Bruno					75%			20%	5%		
Carla											
Marcela							65%	10%		25%	
Caio				5%	15%			80%			
Marcelo									90%	10%	

Tabela 5.6: Taxas obtidas da Tabela 5.5.

<b>Verdadeiro Positivo</b>	95,83%
<b>Verdadeiro Negativo</b>	0%
<b>Falso Negativo</b>	4,17%

Basicamente, quando o *UserListener* obtém os eventos do *UserDriver* envia mensagens pelo Twitter para os usuários no ambiente, conforme o evento recebido. A Figura 5.12 mostra o fluxo básico de execução do *listener* e as mensagens padrões para cada tipo de evento recebido. Para enviar as mensagens pelo Twitter foi utilizado a biblioteca *twitter4j*.

Testes funcionais foram feitos com a aplicação mostrando que o driver consegue obter os dados íntegros do Sistema TRUE e gerar os eventos de maneira quase instantânea. Algumas vezes as mensagens demoravam a chegar ao Twitter, geralmente nos horários de “pico” quando o Twitter operava próximo ao limite da sua capacidade. A Figura 5.13 mostra as mensagens geradas pela aplicação em um teste funcional, onde Danilo, um usuário cadastrado, entra no ambiente senta em uma mesa com seu notebook permanecendo no mesmo lugar por mais de uma hora, e logo depois deixa o ambiente.

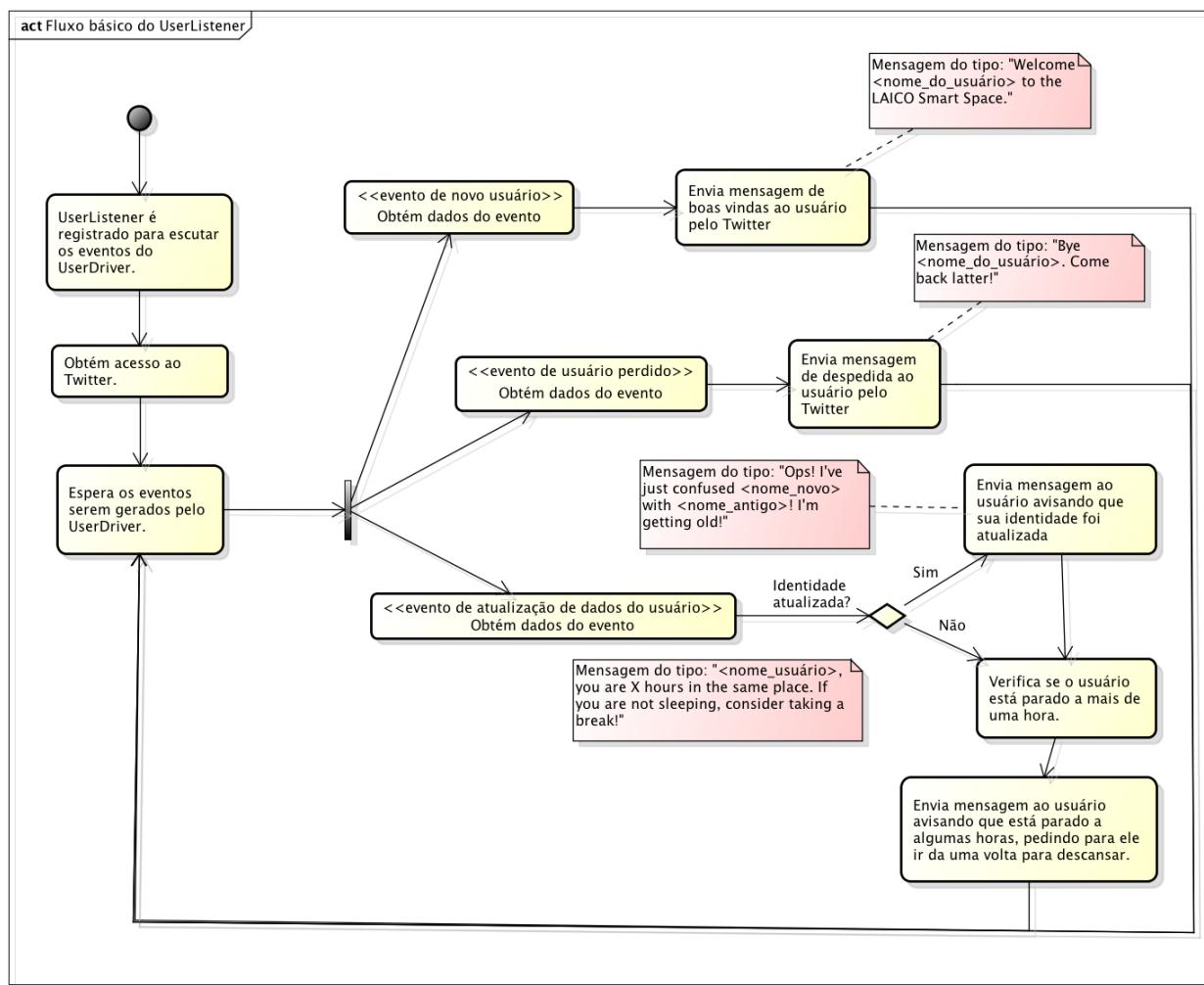


Figura 5.12: Fluxo básico de execução do *listener UserListener*.



Figura 5.13: Exemplo das mensagens enviadas pelo Twitter aos usuários no ambiente.

# Capítulo 6

## Conclusão

# Referências

- [1] How face detection works. *SERVO Magazine*, fevereiro 2007. vii, 17, 18, 19, 20
- [2] A. Abad, C. Canton-Ferrer, C. Segura, J. Landabaso, Luis, Macho, Dušan, Casas, J. Ramon, Hernando, Javier, Pardàs, Montse, and Climent Nadeu. Upc audio, video and multimodal person tracking systems in the clear evaluation campaign. In *Proceedings of the 1st international evaluation conference on Classification of events, activities and relationships*, CLEAR'06, pages 93–104, Berlin, Heidelberg, 2007. Springer-Verlag. 27
- [3] G. Abowd, C. Atkeson, and I. Essa. Ubiquitous smart spaces. *Georgia Institute of Technology, College of Computing*, 1998. 1
- [4] K. Bernardin, T. Gehrig, and R. Stiefelhagen. Multimodal technologies for perception of humans. chapter Multi-level Particle Filter Fusion of Features and Cues for Audio-Visual Person Tracking, pages 70–81. Springer-Verlag, Berlin, Heidelberg, 2008. 27
- [5] Â. R. Bianchini. Arquitetura de redes neurais para o reconhecimento facial baseado no neocognitron. Master's thesis, Universidade Federal de São Carlos, 2001. 1, 2, 10, 11, 12, 14
- [6] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 17, 31, 38
- [7] F. N. Buzeto. Um conjunto de soluções para a construção de aplicativos de computação ubíqua. Master's thesis, Departamento de Ciência da Computação, Universidade de Brasília, <http://monografias.cic.unb.br/dspace/handle/123456789/257>, 2010. 1, 36
- [8] Cambridge. The database of faces. <http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html>, 2011. viii, 39, 40, 56, 57
- [9] C.J.Veenman, M.J.T. Reinders, and E. Backer. Resolving Motion Correspondence for Densely Moving Points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(1):54–72, jan 2001. 5
- [10] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-Based Object Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–575, may 2003. 6
- [11] J. Daugman. Face and gesture recognition: Overview. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):675–676, 1997. 1

- [12] T. S. Körting e N. L. D. Filho. Utilizando eigenfaces para reconhecimento de imagens. *Fundação Universidade Federal do Rio Grande*. 21, 39
- [13] H. K. Ekenel, Q. Jin, M. Fischer, and R. Stiefelhagen. Multimodal technologies for perception of humans. chapter ISL Person Identification Systems in the CLEAR 2007 Evaluations, pages 256–265. Springer-Verlag, Berlin, Heidelberg, 2008. 29
- [14] H. K. Ekenel and A. Pnevmatikakis. Video-based face recognition evaluation in the chil project - run 1. In *Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition*, FGR '06, pages 85–90, Washington, DC, USA, 2006. IEEE Computer Society. 29
- [15] H. K. Ekenel and R. Stiefelhagen. Analysis of local appearance-based face recognition: Effects of feature selection and feature normalization. In *Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop*, CVPRW '06, pages 34–, Washington, DC, USA, 2006. IEEE Computer Society. 29
- [16] S. Emami. Introduction to face detection and face recognition. <http://www.shervinemami.co.cc/faceRecognition.html>, 12 2010. viii, 38, 39, 41
- [17] H. Fairhead. All about kinect. <http://www.i-programmer.info/babbages-bag/2003-kinect-the-technology-.html>, 2011. viii, 67, 68
- [18] F. Fleuret, J. Berclaz, R. Lengagne, and P. Fua. Multicamera people tracking with a probabilistic occupancy map. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30:267–282, February 2008. 30
- [19] A. R. Gomes. Ubiquitos – uma proposta de arquitetura de middleware para a adaptabilidade de serviços em sistemas de computação ubíqua. Master's thesis, Departamento de Ciência da Computação, Universidade de Brasília, <http://monografias.cic.unb.br/dspace/handle/123456789/110>, 2007. 2
- [20] N. Grammalidis, G. Goussis, G. Troufakos, and M. G. Strintzis. 3-d human body tracking from depth images using analysis by synthesis. *Department of Electrical and Computer Engineering University of Thessaloniki*, 2001. 4
- [21] R. Hewitt. Face recognition with eigenface. *SERVO Magazine*, 2007. vii, 21, 22, 23, 24, 25
- [22] J. Hightower and G. Borriello. Location sensing techniques. *University of Washington, Computer Science and Engineering*, agosto 2001. vii, 9, 10
- [23] B. Horn and B. Schunk. Determining optical flow. *Artificial Intelligence Laboratory*, 17:185–203, 1981. 6
- [24] W. Hu, T. Tan, Fellow, IEEE, L. Wang, and S. Maybank. A survey on visual surveillance of a survey on visual surveillance of object motion and behaviors. *IEEE Transactions On Systems, Man, and Cybernetics—part C: Applications and Reviews*, 34(3):334–352, augusto 2004. 7

- [25] R. Jain, R. Kasturi, and B. G. Schunck. *Machine vision*. McGraw-Hill, Inc., New York, NY, USA, 1995. vii, 8, 9, 10
- [26] S. A. D. Junior. Reconhecimento facial 3d utilizando o simulated annealing com as medidas surface interpenetration measure e m-estimator sample consensus. Master's thesis, Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná, 2007. vii, 11, 12
- [27] Hong L. and Jain A. Integrating faces and fingerprints for personal identification. *IEEE Transactions on Pattern and Machine Intelligence*, 20(12):1295–1307, dezembro 1998. 11, 12, 13, 14, 15
- [28] O. Lanz, P. Chippendale, and R. Brunelli. Multimodal technologies for perception of humans. chapter An Appearance-Based Particle Filter for Visual Tracking in Smart Rooms, pages 57–69. Springer-Verlag, Berlin, Heidelberg, 2008. 27
- [29] E . C. Lopes. Detecção de faces e características faciais. Technical report, Pontifícia Universidade Católica do Rio Grande do Sul. 17
- [30] A. P. Pentland M. A. Turk. Face recognition using eigenfaces. *IEEE Computer Society Confer*, 1991. 23
- [31] J. H. Saito M. Arantes, A. N. Ide. A system for fingerprint minutiae classification and recognition. In *Proceedings of the 9th International Conference on Neural Information Processing(ICONIP'02)*, volume 5, pages 2474 – 2478. ix, 12, 13
- [32] A. Pentland M. Turk. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991. 21, 24
- [33] N. Ahuja M. Yang, D. J. Kriegman. Detecting faces in images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):34–58, janeiro 2002. 16, 17
- [34] Edson Lek Hong Ma. Avaliação de características haar em um modelo de detecção de face. *Universidade de Brasília, Instituto de Ciências Exatas, Departamento de Ciência da Computação*, 2007. vii, 17, 18
- [35] T. B. Moeslund and E. Granum. A survey of computer vision-based human motion capture. *Comput. Vis. Image Underst.*, 81:231–268, March 2001. 5, 6
- [36] D. R. Oliveira. Reconhecimento de faces usando redes neurais e biometria. Master's thesis, São José dos Campos: Instituto Nacional de Pesquisas Espaciais (INPE), setembro 2003. 14, 15, 16, 21
- [37] D. J. Kriegman P. N. Belhumeur, J. P. Hespanha. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *European Conference on Computer Vision*, 1996. vii, 14, 15, 21
- [38] M. Jones P. Viola. Robust real-time object detection. *Second International Workshop on Statistical and Computational Theories of Vision – Modeling, Learning, Computing, and Sampling*, julho 2001. vii, 17, 18, 19, 20, 38

- [39] Vytautas Perlibakas. Distance measures for pca-based face recognition. *Pattern Recogn. Lett.*, 25:711–724, abril 2004. 22
- [40] Caroline Rougier, Edouard Auvinet, Jacqueline Rousseau, Max Mignotte, and Jean Meunier. Fall detection from depth map video sequences. In Bessam Abdulrazak, Sylvain Giroux, Bruno Bouchard, Hélène Pigot, and Mounir Mokhtari, editors, *ICOST*, volume 6719 of *Lecture Notes in Computer Science*, pages 121–128. Springer, 2011. 8, 9
- [41] A. Jain S. Pankanti, R. M. Bolle. Guest editors’ introduction: Biometrics—the future of identification. *Computer*, 33:46–49, 2000. 1, 11, 12
- [42] A. Ali Salah, R. Morros, J. Luque, C. Segura, J. Hernando, O. Ambekar, B. Schouten, and E. Pauwels. Multimodal identification and localization of users in a smart environment. *Journal on Multimodal User Interfaces*, 2(2):75–91, setembro 2008. vii, 29, 30, 31, 32
- [43] D. Serby, E. Koller-Meier, and L. Van Gool. Probabilistic Object Tracking Using Multiple Features. *17th International Conference on Pattern Recognition (ICPR)*, 2:184–187, 2004. 5
- [44] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000. 7
- [45] A. Stergiou, A. Pnevmatikakis, and L. Polymenakos. A decision fusion system across time and classifiers for audio-visual person identification. In *Proceedings of the 1st international evaluation conference on Classification of events, activities and relationships*, CLEAR’06, pages 223–232, Berlin, Heidelberg, 2007. Springer-Verlag. 29
- [46] Becky Stern. How-to: Modded camera looks at kinect infrared output. <http://blog.makezine.com/archive/2010/11/how-to-modded-camera-looks-at-kinec.html>, 2011. vii, 10
- [47] R. Stiefelhagen, K. Bernardin, H. Kemal Ekenel, and M. Voit. Tracking identities and attention in smart environments - contributions and progress in the chil project. In *FG*, pages 1–8, 2008. vii, 26, 28
- [48] Inc. Sun Microsystems. The java native interface - programmer’s guide and specification. 72
- [49] M. M. Trivedi, K. S. Huang, and I. Mikic. Dynamic context capture and distributed video arrays for intelligent spaces. *Ieee Transactions On Systems, Man, and Cybernetics—part A: Systems and Humans*, 35(1):145 – 163, janeiro 2005. vii, 32, 33, 34
- [50] A. Tyagi, G. Potamianos, J. W. Davis, and S. M. Chu. Fusion of multiple camera views for kernel-based 3d tracking. In *Proceedings of the IEEE Workshop on Motion and Video Computing*, pages 1–, Washington, DC, USA, 2007. IEEE Computer Society. 27

- [51] A. Waibel, R. Stiefelhagen, R. Carlson, J. R. Casas, J. Kleindienst, L. Lamel, O. Lanz, D. Mostefa, M. Omologo, F. Pianesi, L. Polymenakos, G. Potamianos, J. Soldatos, G. Sutschet, and J. Terken. Computers in the human interaction loop. In *Handbook of Ambient Intelligence and Smart Environments*, pages 1071–1116. 2010. 26
- [52] M. Weiser. The computer for the 21st century. *Scientific American*, 1991. 1
- [53] M. Weiser. The world is not a desktop. *ACM Interactions*, 1:7–8, 1993. 1
- [54] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4):13+, December 2006. vii, 3, 4, 5, 6, 7, 8
- [55] A. Yilmaz, X. Li, and M. Shah. Contour-Based Object Tracking with Occlusion Handling in Video Acquired Using Mobile Cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1531–1536, nov 2004. 6
- [56] Z. Zhang, G. Potamianos, A. W. Senior, and T. S. Huang. Joint face and head tracking inside multi-camera smart rooms. In *Signal, Image and Video Processing*, pages 163–178, 2007. 27

# Apêndice A

## Kinect

O Kinect, mostrado na Figura A.1, é o nome de um projeto da Microsoft para seu console de videogame Xbox 360, que tem ainda como colaboradora a empresa Prime Sense. O projeto visa criar uma nova tecnologia capaz de permitir aos jogadores interagir com os jogos eletrônicos sem a necessidade de ter em mãos um controle(*joystick*), inovando no campo da jogabilidade.



Figura A.1: Sensor Kinect da Microsoft.

A Microsoft define o Kinect como “jogos sem necessidade de controle e experiência de entretenimento”. Porém, vê-lo como um novo jeito de jogar é subestimar sua significância (17).

O Kinect possui as seguintes especificações técnicas:

- Sensor
  - Lentes com detecção de cores e profundidade

- Microfone de voz
- Motor de inclinação para ajuste do sensor
- Campo de visão
  - Campo de visão horizontal: 57 graus
  - Campo de visão vertical: 43 graus
  - Alcance físico da inclinação: (+/-) 27 graus
  - Um alcance máximo de aproximadamente 4.5 metros para câmera de profundidade.
- Fluxo de Dados
  - 320x240 16-bit depth a 30FPS
  - 640x480 32-bit color a 30FPS
  - 16-bit áudio a 16 kHz

Basicamente, é um hardware composto por câmeras que obtém imagens de cor, som e que utiliza iluminação infra-vermelha (IR) para obter imagens de profundidade (17). A Figura A.2 mostra a organização interna do Kinect em alto nível.

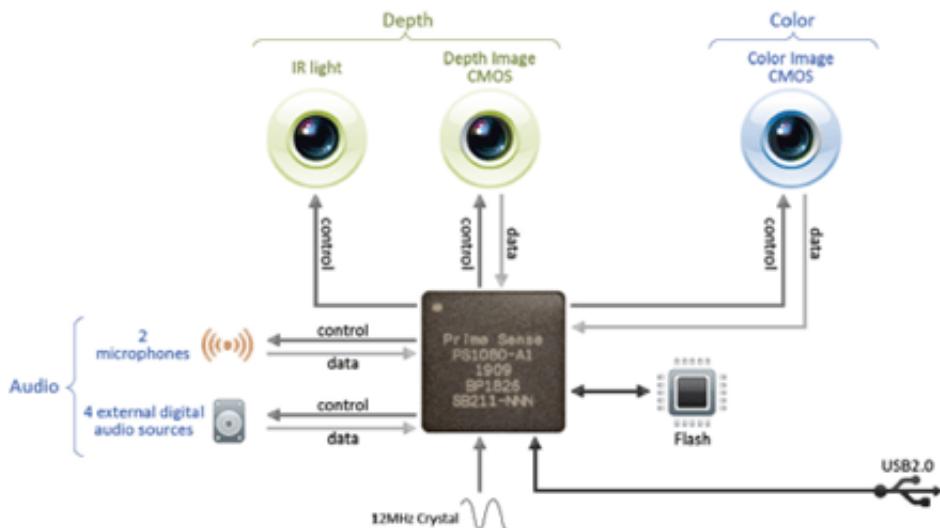


Figura A.2: Organização interna do Kinect (17).

Um chip personalizado processa os dados providos da câmera de profundidade que está correlacionado com as imagens de cor. Com isso, o software que utiliza o Kinect pode combinar cada pixel com sua profundidade. Os dados processados, são enviados para a máquina por meio de uma interface USB na forma de mapas de profundidades e imagens de cor (17).

Com os dados provados pelo sensor (mapas de profundidades e imagens de cor) e pelo preço acessível, ele se torna um dispositivo passível de ser usado em tarefas como rastreamento, localização e reconhecimento facial.

# Apêndice B

## Processamento da imagem

As etapas de processamento da imagem utilizadas nesse trabalho consistem na conversão em escala de cinza, corte, redimensionamento e equalização da mesma. Neste apêndice são mostrado trechos de códigos que implementam tais etapas utilizando a biblioteca *OpenCV*.

### B.1 Escala de Cinza

```
1 IplImage* imageToGreyscale( const IplImage *imageSrc ) {
2     IplImage *imageGrey;
3
4     // cria uma nova imagem do mesmo tamanho que a imagem passada como
5     // parametro e de 1 canal.
6     imageGrey = cvCreateImage( cvGetSize(imageSrc) , IPL_DEPTH_8U, 1 );
7
8     // converte o espaço de cor de imageSrc para o espaço de cor (cinza)
9     // da imageGrey
10    cvCvtColor( imageSrc , imageGrey , CV_BGR2GRAY );
11
12    return imageGrey;
13 }
```

Listing B.1: Conversão de uma imagem para escala de cinza.

### B.2 Corte da Imagem

```
1 IplImage* crop( const IplImage *img , const CvRect region ) {
2     IplImage *imageTmp;
3     IplImage *imageRGB;
4     CvSize size ;
5     size .height = img->height ;
6     size .width = img->width ;
7
8     if (img->depth != IPL_DEPTH_8U) {
9         printf("ERROR: img->depth de %d desconhecido ao invés de 8 bits por
10            pixel passado a crop().\n" , img->depth );
11         exit(1);
```

```

11 }
12
13 // cria uma nova imagem (colorida ou em cinza) e copia os conteudos
14 // da imagem nela.
14 imageTmp = cvCreateImage( size , IPL_DEPTH_8U, img->nChannels);
15 cvCopy( img , imageTmp , NULL);
16
17 // cria uma nova imagem da regiao detectada
18 // seta a regiao de interesse que ao redor da face
19 cvSetImageROI(imageTmp, region);
20
21 // copia a imagem de interesse na nova iplImage (imageRGB) e a retorna
22 size.width = region.width;
23 size.height = region.height;
24 imageRGB = cvCreateImage( size , IPL_DEPTH_8U, img->nChannels);
25 cvCopy( imageTmp , imageRGB , NULL); // copia somente a regiao
26
27 cvReleaseImage(&imageTmp);
28 return imageRGB;
29 }
```

Listing B.2: Corte de uma imagem.

### B.3 Redimensionamento

```

1 IplImage* resize( const IplImage *origImg , int newWidth, int newHeight)
2 {
3     IplImage *outImg = 0;
4     int origWidth , origHeight ;
5
6     if ( origImg ) {
7         origWidth = origImg->width;
8         origHeight = origImg->height ;
9     }
10
11    if ( newWidth <= 0 || newHeight <= 0 || origImg == 0 || origWidth <= 0
12        || origHeight <= 0) {
13        printf("ERROR: Tamanho %dx%d desejado para imagem invalido\n",
14              newWidth, newHeight);
15        exit(1);
16    }
17
18    // modifica as dimensoes da imagem, mesmo se a proporcao mude
19    outImg = cvCreateImage( cvSize(newWidth, newHeight), origImg->depth ,
20                          origImg->nChannels);
21    cvResetImageROI((IplImage*) origImg );
22    if (newWidth > origImg->width && newHeight > origImg->height) { // aumentar a imagem
23        //CV_INTER_LINEAR muito boa para aumentar a imagem
24        cvResize(origImg , outImg , CV_INTER_LINEAR);
25
26    } else { // diminuir a imagem
27        //CV_INTER_AREA muito boa para diminuir a imagem, porem pessima
28        // para aumenta-la
29 }
```

```
24         cvResize( origImg , outImg , CV_INTER_AREA ) ;
25     }
26
27     return outImg ;
28 }
```

Listing B.3: Redimensionamento de uma imagem.

## B.4 Equalização

```
1 // cria uma imagem limpa na escala de cinza
2 equalizedImg = cvCreateImage( cvGetSize( sizedImg ) , 8 , 1 ) ;
3
4 // metodo que realiza "equaliza o de histograma"
5 // normalizando o brilho e aumentando o contraste
6 cvEqualizeHist( sizedImg , equalizedImg ) ;
```

Listing B.4: Equalização de uma imagem.

# Apêndice C

## JNI

JNI (*Java Native Interface*) é uma *framework* que permite que aplicações em Java possam chamar e serem chamadas por aplicações nativas e bibliotecas escritas em outras linguagem tal como C, C++ e Assembly. O propósito dessa abordagem é oferecer a possibilidade de que programadores possam implementar funcionalidades não disponibilizadas pela API padrão do Java ou até mesmo melhorar as já implementadas seja por questão de desempenho, segurança ou outros.

Para implementar os metodos nativos basta criar uma função com a estrutura C.1. Quando a JVM executar o metodo ela invocará a função definida e passará os parametros conforme o esperado (48).

```
1  JNIEXPORT void JNICALL Java_ClassName_MethodName(JNIEnv *env, jobject
2      obj) {
3      /*Implement Native Method Here*/
4  }
```

Listing C.1: HelloWorld.java.

### C.1 HelloWorld

Para criar o primeiro programa utilizando a JNI basta criar os arquivos C.2, C.3, C.4 e C.5 e no console executar (48):

```
chmod +x make.sh
./make.sh
```

```
1  class HelloWorld {
2      static {
3          System.loadLibrary("HelloWorld");
4      }
5
6      private native void print();
7
8      public static void main(String[] args) {
9          new HelloWorld().print();
10     }
11 }
```

Listing C.2: HelloWorld.java.

```

1  /* DO NOT EDIT THIS FILE – it is machine generated */
2  #include <jni.h>
3  /* Header for class HelloWorld */
4
5  #ifndef _Included_HelloWorld
6  #define _Included_HelloWorld
7  #ifdef __cplusplus
8  extern "C" {
9  #endif
10 #endif
11 /*
12  * Class:      HelloWorld
13  * Method:    print
14  * Signature: ()V
15  */
16 JNIEXPORT void JNICALL Java_HelloWorld_print(JNIEnv *, jobject);
17
18 #ifdef __cplusplus
19 }
20 #endif
21 #endif

```

Listing C.3: HelloWorld.h.

```

1  #include "jni.h"
2  #include <stdio.h>
3  #include "HelloWorld.h"
4
5  JNIEXPORT void JNICALL Java_HelloWorld_print(JNIEnv *env, jobject obj)
6  {
7      printf("Hello World!\n");
8      return;
}

```

Listing C.4: HelloWorld.c.

```

1 #!/bin/sh
2  export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:.
3  javah HelloWorld
4  gcc -shared -Wl,-soname,HelloWorld -o libHelloWorld.so HelloWorld.c \
5      -I/usr/lib/jvm/java-6-openjdk/include \
6      -I/usr/lib/jvm/java-6-openjdk/include/linux
7  javac HelloWorld.java
8  java HelloWorld

```

Listing C.5: make.sh.