UNIVERSIDADE DO VALE DO RIO DOS SINOS
CIÊNCIAS EXATAS E TECNOLÓGICAS.
PROGRAMA INTERDISCIPLINAR DE PÓS-GRADUAÇÃO EM
COMPUTAÇÃO APLICADA

Rodrigo Schramm

# Detecção de Faces e Rastreamento da Pose da Cabeça

São Leopoldo
2009

Rodrigo Schramm

# Detecção de Faces e Rastreamento da Pose da Cabeça

Dissertação apresentada à Universidade do Vale do Rio dos Sinos como requisito parcial para obtenção do título de Mestre em Computação Aplicada.

Orientador: Prof. Dr. Cláudio Rosito Jung

São Leopoldo

2009

S377d Schramm, Rodrigo

Detecção de faces e rastreamento da pose da cabeça / por Rodrigo Schramm. – 2009.

81 f. ; 30 cm.

Dissertação (mestrado) – Universidade do Vale do Rio dos Sinos, Programa Interdisciplinar de Pós-Graduação em Computação Aplicada, 2009.

"Orientação: Prof. Dr. Cláudio Rosito Jung, Ciências Exatas e Tecnológicas".

1.Rastreamento eletrônico – Computação. 2.Visão computacional. 3.Computação gráfica – Imagem. 4.Interface homem-máquina. 5.Interação homem-máquina. 6.Processamento – Imagem. 7.Percepção da face. 8.Biometria. I.Título.

CDU 004:621.398
004.92
004.932

Catalogação na publicação:
Bibliotecária Carla Maria Goulart de Moraes – CRB 10/1252

UNISINOS

UNIVERSIDADE DO VALE DO RIO DOS SINOS
Ciências Exatas e Tecnológicas
Programa Interdisciplinar de Pós-Graduação em Computação Aplicada – PIPCA

Errata

No documento folha de aprovação, referente ao título da dissertação do aluno Rodrigo Schramm, PPG em Computação Aplicada, há um erro de grafia na palavra FASES, onde deveria constar **FACES**. Assim, o título correto é: **"Detecção de Faces e Rastreamento da Pose da Cabeça".**

À disposição para maiores esclarecimentos.

Prof. Dr. Arthur Tórgo Gómez
Coordenador do PPG em Computação Aplicada.

Av. Unisinos, 950 Caixa Postal 275 CEP 93022-000 São Leopoldo Rio Grande do Sul Brasil
Fone: (51) 590-3333, Ramal 1701 Fax: (51) 590-8172 http://www.unisinos.br

UNISINOS

**UNIVERSIDADE DO VALE DO RIO DOS SINOS – UNISINOS**
**CIÊNCIAS EXATAS E TECNOLÓGICAS**
**PROGRAMA INTERDISCIPLINAR DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO APLICADA**

Aluno: **Rodrigo Schramm**

Título da Dissertação: "Detecção de Fases e Rastreamento da Pose da Cabeça"

Banca:

Dr. Cláudio Rosito Jung
Presidente/Orientador

Dr. Jacob Scharcanski
Membro da banca externo

Dra. Patrícia Augustin Jaques Maillard
Membro da banca interno

A banca examinadora da Dissertação, sob registro de Ata nº 08/2009 - PIPCA, em cumprimento ao Regimento do Programa Interdisciplinar de Pós-graduação em Computação Aplicada, julga esta Dissertação aprovada para o processo de obtenção do título de Mestre a Rodrigo Schramm.

São Leopoldo, 20 de março de 2009.

*Dedico este trabalho*

*à minha esposa, Helena.*

# Agradecimentos

À minha esposa, Helena, que me incentivou durante todo o período em que estive envolvido com o mestrado, dando-me carinho e todo o suporte de que precisei. Também a meus pais, Alexandre e Liane, que sempre estiveram ao meu lado, torcendo por mim, mesmo sem entender bem o motivo pelo qual eu dedicava tanto tempo a meus estudos.

A todos os meus professores, pelos ensinamentos e pelos desafios. Em especial ao meu orientador, Dr. Cláudio Rosito Jung, por estar sempre presente não apenas ensinando conteúdos acadêmicos, mas principalmente apontando a direção na qual eu deveria olhar.

À Unisinos, instituição que garantiu toda a infra-estrutura necessária para a realização das pesquisas envolvidas no meu trabalho, e à Hewlett-Packard, pelo auxílio financeiro.

A todos, minha mais sincera e profunda gratidão, porque uma vitória como esta não se alcança sozinho!

# Resumo

As câmeras de vídeo já fazem parte dos novos modelos de interação entre o homem e a máquina. Através destas, a face e a pose da cabeça podem ser detectadas promovendo novos recursos para o usuário. Entre o conjunto de aplicações que têm se beneficiado deste tipo de recurso estão a vídeo-conferência, os jogos educacionais e de entretenimento, o controle de atenção de motoristas e a medida de foco de atenção. Nesse contexto insere-se essa proposta de mestrado, a qual propõe um novo modelo para detectar e rastrear a pose da cabeça a partir de uma seqüência de vídeo obtida com uma câmera monocular. Para alcançar esse objetivo, duas etapas principais foram desenvolvidas: a detecção da face e o rastreamento da pose da cabeça. A primeira etapa serve de inicialização para o rastreamento da pose. Nessa etapa, a face é detectada em pose frontal utilizando-se um detector com *haar-like features*. Na segunda etapa do algoritmo, após a detecção da face em pose frontal, atributos específicos da mesma são rastreados para estimar a variação da pose da cabeça.

**Palavras-chave:** Visão Computacional, Detecção de Faces, Rastreamento da Pose da Cabeça, Interface Homem-Computador.

**TITLE:** "FACE DETECTION AND HEAD POSE ESTIMATION"

# Abstract

Video cameras are already part of the new man-machine interaction models. Through these, the face and pose of the head can be found, providing new resources for users. Among the applications that have benefited from this type of resource are video conference, educational and entertainment games, and measurement of attention focus. In this context, this Master's thesis proposes a new model to detect and track the pose of the head in a video sequence captured by a monocular camera. To achieve this goal, two main stages were developed: face detection and head pose tracking. The first stage is the starting point for tracking the pose. In this stage, the face is detected in frontal pose using a detector with Haar-like features. In the second step of the algorithm, after detecting the face in frontal pose, specific attributes of the head are tracked to estimate the change in the pose of the head.

**Keywords:** Computer Vision, Face Detection, Head Pose Estimation, Human-Computer Interaction.

# Sumário

# Lista de Abreviaturas

| | |
|---|---|
| **ACP** | Análise de Componentes Principais |
| **HSV** | *Hue, Saturation and Value* |
| **IRLS** | *Iteratively Reweighted Least Squares* |
| **KLT** | *Kanade-Lucas-Tomasi* |
| **MLESAC** | *Maximum Likelihood Estimation Sample Consensus* |
| **OpenCV** | *Open Computer Vision Library* |
| **POSIT** | *Pose from Orthography and Scaling with Iterations* |
| **RANSAC** | *Random Sample Consensus* |
| **RGB** | *Red, Green and Blue* |
| **RNA** | Rede Neural Artificial |
| **SIFT** | *Scale Invariant Feature Transform* |
| **SVM** | *Support Vector Machine* |
| **YCbCr** | *Luma, Blue Chroma and Red Chroma* |

# Lista de Figuras

# Lista de Tabelas

# Capítulo 1

# Introdução

A pesquisa por novas maneiras de interação entre o homem e a máquina aliada ao baixo custo dos dispositivos de hardware promoveram o desenvolvimento de novos mecanismos de interação. Neste contexto, um dispositivo muito utilizado é a câmera de vídeo, que vem sendo empregada em sistemas de visão computacional para detectar a face e analisar movimentos da cabeça a partir de seqüências de vídeo.

Diversas aplicações utilizam esse tipo de informação. Um dos exemplos são os sistemas de segurança, os quais podem utilizar a face como autenticação [1]. Outros estão na área de entretenimento, como é o caso de novos jogos de *videogame* [2] que utilizam o movimento da cabeça para controlar o jogo, melhorando a interatividade do mesmo. Na indústria automobilística, montadoras de automóveis investem em pesquisas e desenvolvimento de sistemas que utilizam a posição da cabeça como controle de atenção para os motoristas [3] [4]. Outra motivação para a realização deste trabalho se encontra nos sistemas de vídeo-conferência, pois, utilizando-se de técnicas de visão computacional, é possível localizar o locutor e também seu foco de atenção, identificando com quem ele está falando e possibilitando a geração de relatórios do fluxo de conversação [5] [6].

É nesse contexto que se insere essa proposta de dissertação, cujo objetivo é identificar faces e suas respectivas orientações em relação à câmera durante uma seqüência de vídeo. A detecção da face e da respectiva pose é uma tarefa complexa, pois além das variações de características faciais entre uma pessoa e outra, essas características são deformadas de acordo com o tipo e fonte de iluminação. Diversos algoritmos foram propostos para efetuar essa tarefa e, de uma forma geral, são

divididos em duas etapas. Na primeira, eles efetuam a classificação de uma subregião da imagem em face e não-face. Na segunda, a orientação da cabeça (pose) é detectada. Essa detecção da pose, por sua vez, pode acontecer por intermédio de um refinamento da classificação em poses discretas, ou pela estimativa da variação da posição de determinadas características quando a cabeça se movimenta.

## 1.1 Objetivo

O principal objetivo deste trabalho é desenvolver um sistema capaz de identificar a presença de faces a partir de uma seqüência de vídeo, estimando suas posições e respectivas orientações em relação a uma câmera monocular colorida (neste trabalho, serão *webcams*). A localização da cabeça será feita através da detecção de sua face na posição frontal em relação à câmera, utilizando um algoritmo de detecção de faces gerado a partir de uma base de treinamento. Para estimar a orientação da cabeça, um sistema de rastreamento fará o acompanhamento da face detectada previamente pelo detector de face. Juntamente com o detector de faces, visando diminuir o custo computacional, uma pré-classificação da imagem limitará a região de busca na mesma. A partir dessas considerações são definidos os objetivos específicos:

1. Construir um detector de faces frontais para seqüências de vídeo obtidas a partir de uma câmera monocular colorida.

2. Detectar a orientação 3D da cabeça por intermédio do acompanhamento de propriedades específicas selecionadas durante a fase de detecção da face.

3. Otimizar o custo computacional do sistema de forma que o mesmo possa ser utilizado simultaneamente com outros aplicativos em máquinas domésticas.

## 1.2 Estrutura do texto

O restante deste trabalho está estruturado da seguinte maneira. O capítulo 2 apresenta uma revisão bibliográfica sobre detecção de faces e estimativa da pose. O capítulo 3 apresenta a definição do modelo proposto nessa dissertação para estimar a pose da cabeça. Nesse capítulo, primeiramente é apresentado o detector de faces,

onde uma contribuição importante foi adicionada, qual seja, a redução do espaço de busca, proporcionando a diminuição do custo computacional e redução de falsos positivos. Na segunda parte desse mesmo capítulo, o algoritmo de rastreamento da pose é apresentado, incluindo outras contribuições significativas que visam melhorar a estimativa da pose. No capítulo 4 são apresentados os resultados obtidos ao aplicar a técnica descrita nesta dissertação. Esses resultados são ilustrados em gráficos, imagens e também tabelas comparativas. Essas últimas contém medidas do erro absoluto médio obtidas durante a aplicação do algoritmo em vídeo seqüências com *ground-truth*. Para finalizar, no capítulo 5 são feitas considerações sobre o trabalho desenvolvido e seus resultados.

# Capítulo 2

# Revisão Bibliográfica

A detecção da face e respectiva pose, cujos primeiros estudos localizados na revisão bibliográfica deste trabalho datam de 1973[1] [8], é tarefa complexa, ainda não totalmente resolvida. A grande variabilidade de tamanho, cor, posição, rotação e forma da face aumentam a complexidade do problema. Além disso, a face não é um objeto rígido nem estático, mas sim flexível e dinâmico, que varia sua forma em decorrência de fatores como expressão facial, iluminação, presença de artefatos e até mesmo a técnica de aquisição da imagem.

As primeiras técnicas de detecção de faces ainda se utilizavam de ambientes bastante controlados, onde a câmera era fixada num determinado ponto em relação ao objeto de interesse, sendo mantidas com posição e pose rígidas, além do controle total da iluminação. Porém, em aplicações reais, nem sempre é possível criar ambientes controlados. Por isso, a tarefa de detectar faces e suas respectivas orientações tem sido pesquisada e trabalhada há mais de trinta anos, em busca de soluções que visam tratar o problema da variabilidade de iluminação, posicionamento da câmera e do objeto-alvo, oclusões, presença de artefatos e variações nas expressões faciais.

Preocupados com tais evidências, Yang et al. [9] definiram fatores que precisam ser observados por algoritmos de detecção de faces, sobretudo aqueles que devem ser considerados para a obtenção de sua orientação. O primeiro é a pose da face. De fato, a cabeça possui três eixos de rotação (*Roll, Yaw e Pitch*), conforme ilustrado na Figura 2.1. A partir desses, pode-se gerar uma infinidade de projeções distintas na

[1] Tomou-se conhecimento do relatório técnico de Chan e Bledsoe [7], datado de 1965, contudo não se teve acesso ao respectivo texto.

aquisição pela câmera. Tornando a questão mais complexa, determinados atributos da face normalmente utilizados para detecção e rastreamento podem não ser visíveis em algumas dessas poses. Por exemplo, numa pose frontal, os olhos e boca são atributos importantes, que ajudam na identificação da face. Já num ângulo de 45 graus sobre o eixo *Yaw*, o nariz pode esconder um dos olhos, forçando a utilização de critérios diferentes no algoritmo para a detecção da mesma face.

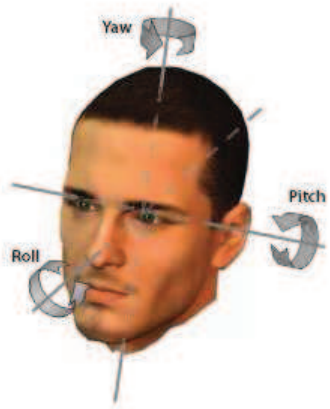


Figura 2.1 – Eixos de rotação da cabeça (em inglês, *Roll, Yaw e Pitch*).
Fonte: [10]

Outro fator corriqueiro e que interfere no desempenho dos algoritmos de detecção de faces é a presença de componentes estruturais, também chamados de artefatos, como óculos e barba. Um terceiro fator de influência é a expressão facial, pois a aparência da face é fortemente modificada, por exemplo, pela presença de um sorriso. Perpassando todos os anteriores, existe também o problema da oclusão, que ocorre nos instantes em que algumas partes da face podem ficar temporariamente obstruídas por algum objeto durante a captura da imagem. Até aqui, foram referidos aspectos que modificam o objeto cuja imagem está sendo captada, isto é, a face em si mesma. Contudo, e não menos relevante, é preciso levar em consideração um fator externo a ela, qual seja, o próprio instrumento de captura, que implica a qualidade da imagem. Durante o processo de captura da imagem, fatores como cor (imagem colorida ou em escala de cinza), resolução, iluminação, ruído e até mesmo deformação causada pelo tipo de lente influenciam no processo de detecção. Nesse item, além de questões pertinentes à qualidade do instrumento de captura, pode-se adicionar também a forma de aquisição, pois esta pode ser feita através de uma única câmera ou através de múltiplas câmeras.

A tarefa de detecção de faces está relacionada diretamente com o objetivo final de uso dessa informação. Como exemplo de uso estão: o reconhecimento ou autenticação de pessoas, a localização espacial de faces em imagens, a avaliação de expressões faciais, o rastreamento, e a estimativa da pose. Para atender todas essas necessidades, são utilizadas diferentes abordagens, as quais muitas vezes são específicas para uma determinada situação. Com o intuito de agrupar melhor esse conjunto grande e bastante diversificado de algoritmos, alguns artigos de *survey* [9, 10] desenvolveram taxonomias, sistematizando trabalhos produzidos nos últimos anos. Segundo esses, a maioria das abordagens são híbridas, ou seja, utilizam combinações de diferentes técnicas para melhorar a precisão da detecção da face e sua pose. A seguir, serão abordadas algumas técnicas existentes na literatura que possibilitam detecção da face e identificação da pose.

## 2.1 Delimitação do espaço de busca

Em diversas aplicações, os objetos de interesse a serem analisados representam apenas uma pequena região da imagem. A busca exaustiva por esses objetos, efetuada pelos algoritmos de detecção, implicam o aumento do custo computacional. Em imagens coloridas, a delimitação do espaço de busca utilizando a informação de cor pode ser utilizada para diminuir esse custo. Além disso, em casos onde a cor representa uma classe distinta de objetos, é possível utilizá-la para segmentar a imagem, efetuando a detecção desses objetos apenas com a informação de cor, sem precisar utilizar um detector baseado em informações geométricas [11]. Nesta seção serão descritas algumas técnicas que permitem classificar pixels em pele ou não pele, diminuindo o espaço de busca por faces na imagem.

A redução do espaço de busca visa contribuir para a diminuição do custo computacional e dos falsos positivos[2]. Porém, é importante frisar que, devido à sensibilidade às variações de iluminação, que interferem significamente na cor, a segmentação pode ter efeitos indesejáveis. De um modo geral, o uso desse tipo de abordagem não procura prioritariamente delimitar a face do restante da imagem, e sim, excluir pixels com alta probabilidade de não pertencerem a regiões da pele.

[2]Exemplos que não pertencem a uma determinada classe, mas que são erroneamente classificados como pertencentes a ela.

Existe um grande conjunto de espaços de cores que podem ser utilizados para representar a cor da pele, como por exemplo HSV, YCbCr e RGB [12]. Conforme o objetivo deste trabalho, no qual se pretende capturar imagens a partir de *webcams*, verifica-se que a maioria desses dispositivos capturam as imagens no formato RGB, que pode ser transformado em outros espaços de cor através de transformações lineares ou não-lineares.

Phung et al. [13] fazem um estudo comparativo entre algumas das principais técnicas de segmentação de cor de pele em diferentes modelos/espaços de cor. De um modo geral, as principais técnicas se baseiam em classificadores lineares, classificadores Bayesianos e classificadores não-lineares.

Os classificadores lineares separam pixels da pele e não-pele por intermédio de limiares (retas ou planos) que dividem o espaço de cor. Esses limiares são usualmente definidos através de *lookup tables*, geradas a partir de histogramas de um conjunto de pixels da pele segmentados manualmente em uma base de dados de treinamento.

Os classificadores Bayesianos definem uma razão probabilística entre os pixels da pele e os não-pele. Se a razão ultrapassar um determinado limiar, então um pixel é classificado como pele. A função de probabilidade condicional para a pele pode ser aproximada por funções Gaussianas paramétricas. Nesse caso, os classificadores Bayesianos também são chamados de classificadores Gaussianos.

Por último, classificadores não-lineares, implementados massivamente através de Redes Neurais Artificiais (RNAs), são capazes de produzir fronteiras de decisão mais complexas. Essas fronteiras são obtidas após um treinamento da rede utilizando um conjunto de imagens contendo pixels que representam pele e não-pele, previamente classificados de forma manual.

Um estudo comparativo entre cinco modelos de cores foi feito por Zarit et al. [14]. Neste estudo, os autores classificaram pixels em pele e não-pele com base em um classificador linear e um classifcador Bayesiano. Segundo seus resultados, o espaço de cor HSV obteve melhor resultado em comparação aos modelos YCrCb e RGB, porém a diferença não foi muito significativa.

O uso de classificadores lineares e Bayesianos são fáceis de implementar, mas falham em ambientes onde a iluminação não é uniforme. Um problema muito comum é evidenciado quando uma face recebe iluminação da luz do sol proveniente de um lado e iluminação artificial do outro, gerando uma diferença significativa entre

os pixels dos dois lados. Além disso, em seqüências de imagens, a variação de iluminação pode ocorrer entre os quadros, ou o objeto pode mudar sua posição em relação às fontes de luz. Nesses casos, a variação da cor da pele pode ser muito grande, dificultando a segmentação da imagem.

Para minimizar esse problema, alguns autores propuseram técnicas capazes de compensar a variação da cor da pele sob condições de variação de iluminação. Wong et al. [15], ao efetuarem uma análise da distribuição de cor da pele, concluíram que os componentes verde e azul são distribuídos linearmente em relação à intensidade da luminância. Observaram também que a intensidade do vermelho é saturada quando a intensidade da luminância de um pixel da cor de pele é maior que 175 (considerando a escala entre 0 e 255). Propõem então um re-mapeamento do canal vermelho, para que o mesmo fique entre 0 e 335. Após essa compensação, o classificador linear não precisa ser alterado e pode ser utilizado para segmentar a pele sob diferentes condições de iluminação.

Outros autores [16], focando suas técnicas em seqüências de vídeo, efetuam a adaptação das regras de segmentação de cor, recalculando-as através da adição de novos pixels provenientes do histórico de imagens já processadas. Os pixels utilizados para a nova estimativa são segmentados através de critérios espaciais, baseando-se em uma região da imagem que é considerada pele. Soriano e colaboradores [17] salientam um problema nesse tipo de abordagem: se em algum momento o critério espacial falhar, pixels não pertencentes à pele influenciarão negativamente o modelo. Esses autores introduzem uma nova técnica, a qual seleciona pixels para adaptar o modelo baseada no conhecimento prévio da faixa de valores pertencentes aos pixels da pele.

Soriano et al. [17] fizeram experimentos para definir uma região de busca apropriada no espaço de cor RGB normalizado. Esse espaço é obtido através da normalização dos componentes R e G por I, onde $I = R + G + B$. Assim, $r = R/I$ e $g = G/I$. O componente $b$ é obtido implicitamente através de $b = 1 - r - g$. O espaço RGB normalizado forma um plano de cromaticidade, na qual a região de cor de pele representada é chamada de *skin locus* [18]. Para definir o *skin locus*, o grupo de Soriano utilizou duas funções quadráticas, uma para definir o limite superior e outra para definir o limite inferior no plano de cromaticidade. O *skin locus* precisa ser definido previamente para cada tipo e modelo de dispositivo (câmera). Os autores

definiram também um modelo de atualização, onde os limiares de classificação são atualizados a cada novo quadro, utilizando apenas pixels que pertencem ao *skin locus* do dispositivo.

## 2.2 Detecção da Face

Como já dito no início deste capítulo, existem diversas técnicas para detecção da face. Essas técnicas utilizam diferentes abordagens para classificar uma imagem em face ou não-face, a partir de diferentes atributos. Algumas delas utilizam a percepção humana sobre as características antropomórficas da face. Basicamente, elas levam em consideração a relação geométrica entre posição dos olhos, nariz e boca para para classificar as faces e suas respectivas orientações.

Nguyen e Huang [19] propuseram um algoritmo para detecção de faces em três estágios. No primeiro, a partir de uma imagem monocromática, é feita a detecção de bordas e segmentação da pele com vistas a extrair a silhueta da face. No segundo estágio, bordas são agrupadas de forma que representem atributos como olhos, nariz e boca. No último, tanto a imagem resultante do primeiro estágio quanto os atributos obtidos no segundo são utilizados para verificar as hipóteses da face baseadas na geometria facial. Existem duas desvantagens nessa técnica. Primeira, relações geométricas somente são evidentes em imagens onde as faces estão próximas da câmera; segunda, ela apresenta grande sensibilidade a oclusões parciais da face.

Wang e Sung [20] desenvolveram um algoritmo para detecção da face e extração de atributos como olhos, nariz e boca. Para detectar a face, primeiramente é feita a segmentação da imagem baseada na cor da pele. Os autores removem partes indesejadas do processo anterior através de operações morfológicas e, finalmente, estimam a posição da cabeça e de seus atributos internos por intermédio do casamento de uma elipse sobre o contorno obtido pelo processo anterior.

Através da relação geométrica do triângulo formado entre os olhos e a boca, Lin e Fan [21] desenvolveram uma técnica para detectar faces. Para estimar os vértices do triângulo, os autores inicialmente binarizam a imagem em escala de cinza através de um limiar pré-definido. Após, os centros dos componentes conexos resultantes são marcados na imagem binária e, por fim, triângulos são detectados através de conjuntos de três componentes que satisfazem algumas condições geométricas. Por

exemplo, a combinação dos olhos e da boca, em vista frontal, formam um triângulo isósceles, assim como a combinação entre um dos olhos, ouvido e a boca formam um triângulo denominado pelos autores de *right triangle*. Um conjunto de treinamento é avaliado pelas regras acima descritas. Os componentes que formam os triângulos são mantidos e utilizados para gerar uma máscara de pesos. Essa máscara penaliza regiões que não satisfazem os critérios já definidos e valoriza as regiões que os satisfazem. Uma vez definido o modelo, o processo de classificação se dá através da comparação entre esta máscara e regiões da imagem de entrada devidamente binarizada, ambas obtidas através do mesmo procedimento que foi utilizado para gerar as imagens de treinamento. Nessa comparação entre as duas imagens binárias, regiões em desacordo resultam valores negativos e regiões em acordo resultam valores positivos. Ao final, se o somatório dessas regiões for maior que um determinado limiar definido empiricamente, então o algoritmo conclui que existe uma face nesta região da imagem. Visando melhorar a segmentação inicial, os autores promoveram uma extensão da técnica, na qual utilizam também informação de cor [22] para fazer a segmentação inicial da imagem.

Papageorgiou et al. [23] desenvolveram um algoritmo para detecção de objetos em imagens estáticas. Nessa técnica, as classes de objetos são representadas a partir de um conjunto de funções wavelets básicas, as *haar-wavelets*. As haar-wavelets são um conjunto natural de funções-base que codificam diferenças entre as médias de intensidades em regiões vizinhas na imagem. Para escolher o conjunto que melhor representa uma classe (nesse caso, faces), os autores fizeram uma análise estatística das classes de objetos, utilizando-se de um conjunto de 2429 imagens em tons de cinza, com tamanho $19 \times 19$ pixels. Após definido o conjunto de coeficientes que melhor representam a classe, os autores utilizaram *Support Vector Machine* (SVMs) como algoritmo de classificação. Eles treinaram o sistema com uma base de dados de exemplos positivos obtidos em imagens de cenas internas e externas. Como exemplos negativos, foram utilizadas imagens de cenas naturais que não continham faces. Os autores atentam para o fato de que, enquanto a classe-alvo (faces) é bem definida, para a classe negativa não existem exemplos típicos, apenas imagens que não contém faces. Por causa disso, a classe negativa para treinamento precisa ser muito grande. Para minimizar esse problema, usaram a idéia de treinamento com *bootstrapping* [24]. Após o treinamento inicial, testaram o sistema sobre um conjunto de imagens que

não possuíam faces. Os falsos positivos obtidos nesse processo são adicionados ao conjunto de exemplos da classe negativa, e o classificador é re-treinado. Esse processo é repetido até que o número de falso negativos seja satisfatoriamente pequeno. Essa técnica obteve, levando em consideração a presença de um falso positivo a cada 15.000 testes, uma taxa de detecção de 70%.

Com o objetivo de minimizar problemas causados por oclusão, variação na aparência e pose dos objetos, Heisele et al. [25] propuseram um detector baseado em componentes. Ao invés de criar um classificador que leva em consideração as características globais da face, eles segmentaram a face em pequenos componentes considerados mais relevantes, os quais são combinados por um algoritmo construído em duas camadas. Na primeira, componentes da face são pré-selecionados e detectados por classificadores SVM (lineares). Um conjunto de 14 regiões consideradas mais significativas pelos autores para representar faces foi extraído de uma base de dados. Um classificador foi treinado para cada uma dessas regiões, servindo o resultado dessa camada como entrada para a segunda camada, onde um classificador SVM (polinomial de segunda ordem) determina a classe final do objeto.

A partir das idéias de Papageorgiou et al. [23], que utilizam *haar-wavelets*, Viola e Jones [26] [27] criaram um sistema detector de objetos visuais com alto desempenho (baixo custo computacional) e alta taxa de acertos na detecção. A idéia básica desse algoritmo é utilizar uma janela deslizante $W$, que varre a imagem de entrada monocromática exaustivamente em busca de uma face com dimensões similares às de $W$. Para tal, um conjunto de atributos denominados de *haar-like features* é empregado. Tais atributos codificam as diferenças médias de intensidade em regiões retangulares vizinhas na imagem. Para que esse cálculo seja rápido, foi utilizada a *imagem integral*, a qual substitui um grande número de cálculos por uma *lookup table*, contendo o somatório de qualquer região retangular da imagem original.

A vantagem da utilização desses atributos é que eles conseguem representar pequenas estruturas, sem se ater aos detalhes, promovendo melhor generalização. Segundo os autores, esse tipo de atributo é menos sensível ao ruído e tolera pequenas variações de iluminação, sendo utilizado como hipóteses fracas. Além disso, com o auxílio da imagem integral, esses atributos podem ser calculados rapidamente. Viola e Jones utilizaram um conjunto com três modelos primitivos de *haar-like features*,

conforme está ilustrado na Figura 2.2. Esses atributos, em formato retangular, avaliam as informações geométricas da janela nas direções vertical, horizontal e diagonal. O valor do atributo é dado pelo somatório dos pixels nas regiões marcadas em branco subtraído do somatório dos pixels nas regiões marcadas em preto. Os modelos primitivos das *haar-like-features* são utilizados com diferentes tamanhos e posições dentro da janela, gerando uma grande quantidade de atributos. A seleção dos atributos considerados adequados para a detecção de faces é feita com base em um conjunto de treinamento.

Para efetuar o treinamento do classificador, os autores utilizaram o algoritmo *AdaBoost* [24], utilizando o conjunto de *haar-like-features* como classificadores fracos (*weak classifiers*). O resultado do *AdaBoost* é um classificador forte (*strong classifier*), obtido através de uma combinação ponderada dos classificadores fracos seguida de um limiar para definir a classe de saída. No treinamento proposto em [26], todas as amostras positivas do treinamento (ou seja, faces) foram normalizadas para um tamanho fixo $20 \times 20$, de modo que o classificador foi treinado para identificar faces com esse mesmo tamanho.

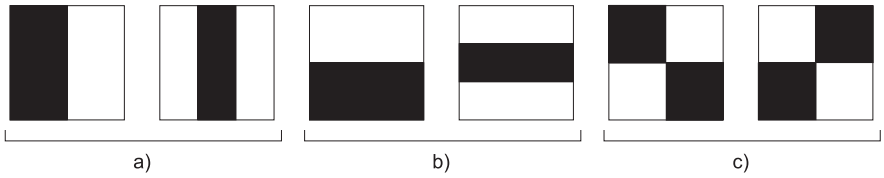


Figura 2.2 – Tipos de *haar-like features* utilizadas por Viola e Jones [26].
a) codifica informação na horizontal. b) codifica informação na vertical. c) codifica informação na diagonal.

Apesar de considerar o uso da imagem integral para executar o processo de classificação mais rapidamente, a soma do custo para testar cada hipótese fraca ainda se torna grande. Para amenizar esse problema, Viola e Jones utilizaram uma estrutura em cascata, onde a classificação acontece em estágios. A chave para essa idéia vem do fato que o processo de *boosting* pode ser construído rejeitando uma boa quantidade de exemplos negativos e aceitando praticamente todos os exemplos positivos com uma pequena quantidade de hipóteses fracas. Obviamente, durante a classificação, haverá muitos falsos positivos. Para diminuir o número de falsos positivos, novos classificadores (estágios) são treinados, reutilizando os exemplos que foram classificados erroneamente no estágio anterior. Dessa forma,

os estágios seguintes são capazes de efetuar classificações mais refinadas; por outro lado, entretanto, são mais custosos. Assim, classificadores mais simples são utilizados nas primeiras camadas da cascata, para rejeitar a maior parte de exemplos negativos antes que classificadores mais complexos, contendo uma quantidade maior de hipóteses fracas, sejam utilizadas nas camadas seguintes.

Ao detectar faces em uma imagem com a estrutura citada acima, grande parte dos candidatos será descartada na primeira camada da cascata e, dessa forma, apenas um pequeno conjunto de hipóteses será testado, consumindo pouco processamento. A partir da taxa de classificação e de falsos positivos desejada para o sistema como um todo, é possível ajustar as taxas para cada estágio individualmente. O classificador final (*strong classifier*) é discreto e binário, retornando 0 para padrões que não são classificados como face e +1 para padrões classificados como face. O classificador final em [26] utiliza aproximadamente 6000 atributos, distribuidos em 38 estágios, e é capaz de detectar faces com uma taxa de acerto de 88.4% com uma razão de falso positivos igual a 6.1%, conforme os resultados apresentados pelos autores.

Com o treinamento descrito acima, o algoritmo de Viola e Jones é capaz de identificar faces com tamanho aproximado de $20 \times 20$ pixels, deslizando uma janela do mesmo tamanho pela imagem e aplicando a cascata de classificadores em cada posição da janela. Janelas com tamanhos variados, obtidas re-escalando a janela inicial sucessivamente em fatores de 10% (os atributos são re-escalados proporcionalmente), também são utilizadas para permitir que faces com tamanhos variados sejam detectadas.

Um problema apontado por Hou et al. [28] é que esse tipo de classificador é treinado com o foco na minimização do erro de classificação, enquanto os limiares utilizados durante o período de treinamento são ajustados para se obter o menor número de falsos positivos e falsos negativos[3]. Como o objetivo final de minimização do erro é desviado, os atributos escolhidos pelo *AdaBoost* não são os que otimizam a classificação. Para resolver esse problema, os autores propuseram uma medida de erro de classificação assimétrica, onde um custo diferenciado é aplicado tanto para falsos positivos quanto para falsos negativos. Desta forma, é possível construir

[3]Exemplos que pertencem a uma determinada classe, mas que não são classificados como pertencentes a ela.

um detector de objetos com um número menor de classificadores mais simples e simultaneamente com performance melhor. Lienhart e Maydt [29] apresentaram um extensão das *haar-like features* propostas por Viola e Jones. Para isso, desenvolveram uma nova representação da imagem integral, que permite atributos rotacionadas em 45 graus. Utilizando esse novo conjunto de atributos, foi possível diminuir o número de falsos positivos em 10% em relação ao conjunto proposto em [26], mantendo a mesma taxa de detecção. Além disso, os autores reduziram o número de estágios para apenas treze, diminuindo o custo computacional. Extrapolando a idéia de Lienhart e Maydt, outros autores [30] adicionaram duas novas imagens integrais para calcular atributos rotacionados em 26,5 e 63,5 graus.

Ichikawa et al. [31] desenvolveram uma abordagem com sete classificadores: um classificador global (toda face), um para os lábios, dois para a boca (um para cada metade, esquerda e direita), um para cada olho (esquerdo e direito) e um para o nariz. Diferentemente de Heisele et al. [25], nessa proposta os autores utilizaram *AdaBoost* com *haar-like features*. A justificativa para o uso dessa técnica é sua superioridade relativa frente às técnicas anteriores, em desempenho e precisão. Segundo seus testes comparativos, o sistema utilizando *AdaBoost* foi 27 vezes mais rápido do que o mesmo utilizando SVM. Por fim, uma árvore de decisão foi construída para avaliar os casos com oclusão de partes da face. Nessas situações, onde alguns dos componentes não são detectados, a combinação de regras definidas na árvore ajudam na detecção de faces presentes.

Métodos não-lineares detectam faces e respectivas poses através da segmentação do espaço que representa a face por intermédio de funções não-lineares obtidas com aprendizado supervisionado. Os principais classificadores para essa categoria são implementados utilizando RNAs. Garcia e Delakis [32] utilizaram uma arquitetura com várias camadas de redes neurais. As primeiras quatro camadas são chamadas de mapas de atributos. Esses mapas são intercalados de forma a segmentar a imagem de entrada e reduzir sua resolução, aumentando a tolerância do sistema às variações de luminosidade, rotação, escala e posição. Ao final, duas camadas são adicionadas para efetuar a classificação, combinando atributos gerados nas camadas iniciais.

Feraud et al. [33] desenvolveram um sistema que utiliza RNAs, composto de quatro etapas. Na primeira, filtros que detectam o movimento de objetos a partir

da diferença entre quadros de uma seqüência de vídeo rejeitam 90% das hipóteses. Na segunda, filtros baseados na cor de pele rejeitam 60% das hipóteses restantes. A terceira etapa utiliza uma rede neural do tipo *multi-layer perceptron* sobre a imagem em escala de cinza, que elimina 93% das hipóteses restantes. Os casos restantes são processados pela última camada. Essa camada utiliza uma rede neural *Constrained Generative Model*, a qual aproxima a projeção do conjunto de dados sobre o sub-espaço que representa as faces, idéia semelhante à desenvolvida em [34]. Somente após passar por todas essas etapas, uma imagem é considerada face. Caso um etapa rejeite a imagem de entrada, o algoritmo é finalizado, reduzindo assim o custo computacional.

Um grande conjunto de técnicas busca projetar as informações que representam a face a partir de um espaço de alta dimensionalidade organizadas em sub-espaços de menor dimensionalidade. Uma dessas técnicas é a *eigenfaces* [35]. Nessa técnica, uma base de imagens é utilizada para gerar um modelo para a face que parte do conjunto de seus autovetores, derivados da matriz de covariância, por sua vez resultante da análise dos dados obtidos durante um período de treinamento. Nas *eigenfaces*, cada píxel da imagem é considerado um vetor e, por isso, a dimesionalidade do modelo é razoavelmente grande. Técnicas como Análise de Componentes Principais (ACP) costumam ser utilizadas para reduzir a dimensionalidade do problema, considerando apenas os autovetores mais significativos.

Um exemplo desse tipo de abordagem foi descrito por Sung e Poggio [34]. Para efetuar a classificação, é efetuada a projeção de um vetor (imagem de entrada) em um sub-espaço composto de aglomerados Gaussianos multi-dimensionais. Esses aglomerados são obtidos pelo treinamento prévio com imagens de exemplos positivos (faces) e negativos (imagens sem faces). Para obter cada um desses aglomerados, os autores utilizaram uma variação do algoritmo *K-means* [36] sobre uma base de dados contendo imagens normalizadas. Para eliminar falsos positivos utilizaram treinamento com *bootstrap*. Uma vez definidos os aglomerados, duas medidas de distância são obtidas para cada imagem de entrada. Uma delas é a distância de Mahalanobis entre o resultado da imagem projetada sobre o aglomerado e próprio centróide do aglomerado; a segunda é a distância Euclidiana entre a imagem de entrada e sua projeção no sub-espaço multidimensional. Essas duas distâncias

servem de entrada para uma RNA, que efetua a classificação final da imagem em face ou não-face.

O sistema descrito por Osuna et al. [37] também detecta faces através da busca exaustiva na imagem por faces em diferentes escalas. Para isso, o sistema utiliza um classificador SVM treinado com padrões de $19 \times 19$ pixels de face e não-face. Além disso, uma máscara é aplicada à imagem, valorizando a região central da face, assim como um algoritmo de correção de iluminação, para melhorar a representação da classe. A SVM utiliza como *kernel* para classificação uma função polinomial de segunda ordem. O sistema desenvolvido pelos autores consegue processar imagens rapidamente e com taxa de detecção de $97,1\%$. Outros autores também desenvolveram detectores utilizando SVM. Por exemplo, Terrilon et al. [38] utilizam *Orthogonal Fourier-Mellin moments* como atributos para treinar o SVM, e Bassiou et al. [39] fizeram um comparativo entre uma SVM e RNA. Segundo tais autores, ambas tiveram desempenho semelhantes, onde obtiveram uma taxa de detecção de aproximadamente $95,6\%$.

Utilizando uma abordagem não muito usual, Wong e Lam [40] criaram um detector com algoritmos genéticos. Por causa do alto custo computacional dos algoritmos genéticos, o espaço de busca foi reduzido apenas às regiões da imagem que possuem candidatos a olhos. Essas regiões são caracterizadas por vales, que podem ser obtidos através de operadores morfológicos, e são utilizadas para inicializar a população do algoritmo genético. A função de *fitness* é obtida pela projeção da região dos olhos sobre um auto-espaço previamente definido através de uma base de treinamento. A principal desvantagem dessa abordagem é o custo computacional, uma vez que é preciso esperar que a população do algoritmo genético evolua.

## 2.3 Identificação da Pose

Diversos trabalhos estenderam técnicas utilizadas na detecção de faces com vistas a resolver o problema de identificação da pose. Essas técnicas utilizam uma série de classificadores treinados, onde cada um é responsável por classificar uma determinada pose discreta. A vantagem desse tipo de abordagem é que cada detector não depende da detecção prévia para estimar a pose no futuro. Ou seja, caso um detector falhe em determinado momento, o sistema não precisará ser reiniciado. Uma

desvantagem dessa abordagem é que o custo computacional cresce linearmente de acordo o número de classificadores utilizados, ou seja, com a precisão da discretização das poses. Um conjunto muito grande de trabalhos foi desenvolvido a partir da idéia de Viola e Jones.

A proposta de Wu et al. [41] utiliza *Real AdaBoost*. Nesse algoritmo, a cada estágio um valor real representa a probabilidade do padrão avaliado pertencer a uma determinada classe (pose), diferentemente do *AdaBoost*, onde a saída é binária. A técnica propaga a probabilidade total de um estágio anterior para o seu próximo, influenciando o resultado dos classificadores das camadas sucessoras. Wang et al. [42] detectam a face e a variação da pose através de uma arquitetura em formato de pirâmide. No topo da pirâmide, candidatos a faces em qualquer orientação são aceitos. A cada novo estágio da pirâmide, a pose é refinada. Para mais bem representar as diferentes poses, os autores propuseram um novo conjunto de *haar-like features* assimétricas, uma vez que esses atributos conseguem representar melhor a variação da posição da face, em especial, posições mais próximas do perfil.

Classificadores não-lineares também são utilizados para detectar as diferentes poses. Através desses é possível, além de classificar discretamente, classificar também a pose de forma contínua, detectando a variação suave da pose facial. Uma implementação usando RNAs foi feita por Rae e Ritter [43]. Eles dividiram o sistema em três etapas, cada uma utilizando uma RNA específica. A primeira etapa efetua a segmentação da cor de pele, enquanto a segunda efetua apenas a detecção da face, indicando para a terceira etapa o posicionamento da mesma na imagem. A última etapa define, então, a orientação da cabeça.

Também foram desenvolvidas técnicas que utilizam redução de dimensionalidade para auxiliar na classificação das poses. Li et al. [44] desenvolveram uma proposta que utiliza *eigenfaces* e SVMs. Eles salientam que algoritmos que utilizam *eigenfaces* são mais rápidos que SVM, porém são menos acurados, pois as poses têm regiões de sobreposição no subespaço de faces. Para minimizar o problema do custo computacional e melhorar a precisão, eles primeiramente classificaram as poses "grosseiramente" com *eigenfaces* e depois refinaram a classificação das mesmas com SVM. A técnica proposta pelo grupo de McKenna [45] também usa *eigenfaces*. Porém, esses autores, além de gerar as *eigenfaces* a partir da imagem em escala de cinza, também as geram a partir da

imagem previamente transformada por *wavelets de Gabor*. Segundo os autores, a primeira mapeia melhor a localização da face e a segunda mapeia melhor a pose da mesma. Segundo suas análises, a componente principal, aquela com maior magnitude dentre as que geram o auto-espaço, divide a distribuição das poses em duas: perfil esquerdo até frontal, e frontal até perfil direito. O terceiro componente principal com maior magnitude representa a variação da iluminação, enquanto os demais componentes principais representam a variação da pose em cada um dos dois intervalos definidos pelo primeiro componente principal.

Um outro tipo de abordagem para detecção da pose utiliza modelos flexíveis. Nesses modelos, as faces são representadas por um conjunto de atributos locais (quinas, por exemplo). Esse conjunto é obtido pelo treinamento prévio, geralmente definido através de pontos marcados manualmente (*landmarks*) ou automaticamente empregando algoritmos de detecção de quinas, como por exemplo o detector de Harris [46]. O modelo gerado é então aplicado à nova imagem que se pretende avaliar. Um processo iterativo aproxima os pontos do modelo aos atributos da imagem a ser avaliada. Através da adaptação do modelo na imagem[4] é possível estimar a pose da face.

Krüger et al. [47] criaram grafos para representar cinco diferentes poses da face (frontal, perfil-esquerdo, central-esquerdo, perfil direito, central-direito). Cada nodo do grafo representa um conjunto de atributos locais, extraídos da imagem através da aplicação de *wavelets Gabor* de diferentes tamanhos e orientações. Nos nodos também são armazenadas suas distâncias em relação ao centróide do grafo, enquanto nas arestas são armazenadas as distâncias médias entre os respectivos nodos. Através de um processo iterativo, todos os nodos da imagem de entrada são comparados com os cinco modelos e adaptados de forma a obter o menor erro. Ao final do processo, o grafo que tiver maior similaridade com o modelo deformado define a pose.

Lanitis et al. [48] utilizaram 152 pontos para definir a figura da face. Para treinar seu modelo eles utilizaram uma base de dados com vinte indivíduos, cada um com oito imagens de exemplos. O modelo gerado pode aproximar qualquer face do conjunto de treinamento através do ajuste de dezesseis parâmetros. Esses parâmetros são os auto-valores mais significativos da matriz de covariância obtida no

[4]Esse modelo adaptativo é conhecido como *Elastic Bunch Graph* [10]

treinamento. Segundo os autores, os três primeiros parâmetros indicam a variação da pose da face. O quarto e o sexto decorrem da variação entre diferentes indivíduos utilizados no treinamento, enquanto o quinto é relativo às variações da expressão da face.

Através do rastreamento, algoritmos extraem a pose interpretando o movimento obtido pela comparação da posição de atributos, em quadros consecutivos de uma seqüência de vídeo [10] [49]. Entende-se como atributo (*feature*) qualquer elemento pertencente a um objeto na imagem capaz de ser identificado após alguma transformação na imagem. Alguns exemplos de atributos utilizados no rastreamento de objetos são bordas, quinas, e elementos como olhos, boca e nariz. Esses métodos apresentam os melhores resultados para estimativa contínua da pose; porém, precisam de um algoritmo secundário para detectar a posição inicial da face. Uma grande vantagem do rastreamento é que ele permite estimar um movimento bastante preciso e contínuo. Em contrapartida, a desvantagem é que, caso seja propagado o erro ao longo do tempo, é preciso reinicializar o sistema.

Alguns métodos utilizam toda a região da face como um atributo. Nesses métodos, a transformação é obtida a partir da comparação da região entre duas imagens. Por exemplo, Cascia et al. [50] desenvolveram um técnica onde a cabeça é mapeada em um modelo cilíndrico. Para isso, durante um período de treinamento offline, a textura da cabeça obtida a partir de imagens provenientes de diferentes poses são projetadas na superfície de um cilindro e posteriormente armazenadas em um mapa de textura. Após a construção do modelo, o rastreamento é feito por intermédio da minimização do erro quadrático da diferença entre uma nova imagem e o mapa de textura.

Uma outra técnica que utiliza um cilindro como modelo foi proposta por Xiao et al. [51]. Porém, diferentemente da técnica anterior, não há um treinamento prévio. Nessa técnica, após inicializado o sistema e detectada a face em posição frontal, a cabeça é rastreada explorando regiões vizinhas, na busca da minimização da diferença do gradiente entre o quadro atual e o quadro anterior. O cálculo da minimização é feito através de *Iteratively Reweighted Least Squares* (IRLS[5]), que inclui uma compensação para mudanças graduais na iluminação. Como o processo

[5]Técnica iterativa para o cálculo de mínimos quadrados utilizando pesos diferenciados para cada parâmetro.

é iterativo, os autores adicionaram ao sistema um conjunto de quadros referenciais, utilizados para estabilizar o sistema.

Conforme já citado anteriormente, existem trabalhos que, ao invés de utilizar a informação global da cabeça, inicializam o sistema com base na detecção de atributos esparsos. É evidente que um dos maiores problemas desse tipo de técnica é a seleção de atributos que sejam invariáveis diante da deformação e da iluminação. Nessas abordagens, como a pose é definida pela variação da posição dos atributos no espaço 3D, é preciso um modelo de transformação que estime a posição de um atributo no próximo quadro. Para isso, utiliza-se um modelo tridimensional da cabeça, onde cada atributo no plano da imagem é associado a um ponto 3D no modelo. Esses algoritmos, de uma forma geral, obtém a transformação através da minimização do erro quadrático da diferença de posição dos atributos obtidos entre dois quadros consecutivos.

Weissenfeld et al. [52] utilizam quinas extraídas com o detector de Harris [46], como atributos da face, e um modelo rígido para a face baseado no CANDIDE [53]. Esses atributos são acompanhados através da seqüência de imagens. Para escolher a melhor transformação que descreve o movimento dos atributos foi utilizado um otimizador evolucionário chamado de Evolução Diferencial[6]. Tordoff et al. [55] também utilizam o detector de Harris para extrair os atributos que serão acompanhados. Porém, nesse algoritmo, após um período de treinamento, são armazenados informações complementares para cada um dos atributos. Durante esse treinamento, à medida que diferentes poses são projetadas em um modelo 3D, vão sendo armazenadas, para cada atributo, regiões contendo informação de intensidade em escala de cinza. Além disso, em cada vértice do modelo 3D são também armazenados regiões contendo informação de cor no espaço YCrCb. Após finalizado o treinamento, a pose é estimada com base na comparação entre correlações das regiões ao redor dos atributos da imagem de entrada com os do modelo. O uso de um grande número de atributos torna o custo computacional muito elevado. Para resolver esse problema os autores utilizam o algoritmo *Maximum Likelihood Estimation Sample Consensus* (MLESAC)[7], amostrando um pequeno conjunto de

[6]Técnica de otimização global através de busca estocástica para resolução de problemas em espaços contínuos com rápida convergência [54].

[7]MLESAC é uma variação do RANSAC que é um método iterativo para estimar parâmetros de um modelo matemático a partir de um conjunto de observações que contém *outliers*. [56].

atributos e calculando a transformação correspondente.

Numa abordagem parecida, Choi et al. [57] utilizam *Scale Invariant Feature Transform* (SIFT) [58] para detectar pontos invariantes em relação à escala e à rotação. Para cada ponto encontrado com a utilização do algoritmo SIFT, existe um ponto no espaço associado ao modelo 3D que representa a face. A cada novo quadro de uma seqüência de vídeo, esses pontos são projetados na imagem e rastreados com o emprego da técnica Kanade-Lucas-Tomasi (KLT) [59]. A pose é estimada com base na variação da posição desses pontos, utilizando o algoritmo POSIT [60]. Os autores utilizam RANSAC para eliminar outliers que deturpam a estimativa.

Alguns trabalhos propuseram o acompanhamento da pose através de modelos não determinísticos, que analisam atributos faciais. Hannaksela et al. [61] utilizaram os olhos e a boca como atributos geométricos extraídos através de um pré-processamento na imagem. A técnica estima a pose da cabeça baseada na posição desses componentes e refina o resultado com a aplicação de um filtro de Kalman que faz a predição e correção da transformação obtida. Uma outra técnica [62] utiliza filtro de Kalman, para predizer a posição dos atributos 2D (sobrancelhas, boca, nariz e olhos) e estimar os parâmetros da transformação da pose da cabeça. Essas técnicas apontam para a vantagem de usar o filtro de Kalman pois o mesmo pode utilizar o histórico dos estados anteriores do sistema para melhor predizer os estados futuros. Com isso, através de uma forma não-determinística é possível prover uma maior estabilidade ao sistema, evitando que o rastreamento derive.

Rahimi et al. [63] desenvolveram uma técnica de acompanhamento diferencial onde, ao contrário da maioria das técnicas diferenciais que utilizam apenas a informação do último quadro para calcular o movimento e tendem a acumular erro, eles estimam as mudanças na pose considerando um conjunto de quadros anteriores empregando um framework probabilístico. Desta forma, cada novo quadro é ancorado a um conjunto de diversos quadros anteriores.

Vacchetti et al. [64] também utilizam a informação de quadros anteriores (quadros-chaves) para estabilizar o rastreamento, contudo, de forma determinística. Inicialmente, num treinamento *offline*, eles armazenam atributos capturados de diferentes poses do objeto. Durante o rastreamento, os atributos são projetados no modelo com base na pose estimada e simultaneamente, *templates* centrados nesses atributos pertencentes a base de dados criada *offline* são comparados com o

quadro atual para estimar a transformação que minimize o erro. Com o objetivo de tornar essa técnica mais robusta, novos quadros chaves podem ser adicionados. Esses quadros são inseridos sempre que a pose estimada for muito distante dos quadros chaves existentes na base de dados.

Extrapolando a idéia de estimativa da pose a partir de rastreamento de características faciais e da comparação com quadros-chaves, os trabalhos [65] [66] propuseram um novo tipo de abordagem, onde essas características não são rastreadas, mas sim detectadas a cada novo quadro. Essas abordagens são conhecidas como rastreamento por detecção (*tracking by detection*). Porém, para isso, é preciso efetuar um treinamento prévio, durante o qual diferentes poses do objeto-alvo são utilizadas para extrair características invariantes a pose, escala e iluminação de sua imagem. De um modo geral, as características mais utilizadas são bordas e quinas da imagem. Para cada uma dessas características, um descritor local é associado para auxiliar na comparação entre a imagem a ser classificada pelo modelo. Esse descritor geralmente codifica a variação de intensidade de uma determinada região da imagem (escala de cinza ou colorida) centrada na posição do atributo extraído. Após o treinamento, onde diferentes poses são armazenadas com suas características mais relevantes, uma imagem pode ser avaliada pelo modelo para estimar a pose do objeto comparando os aspectos da imagem de entrada com os aspectos do modelo treinado. Desta forma, a estimativa da pose torna-se um problema de classificação, na qual cada pose é representada por uma classe. Assim, inúmeros classificadores podem ser utilizados para distingui-las, dentre os quais podem ser citados *K-nearest Neighbor*, SVM, RNA.

O problema dessa abordagem é que, dependendo do número de características a serem consideradas (um número geralmente acima de 200), o custo computacional torna-se muito grande. O treinamento *offline* pode levar minutos, horas ou dias. Além disso, a classificação *online* também pode ser custosa, tornando a execução do algoritmo em tempo real impraticável. Visando diminuir o custo computacional e permitir a classificação em tempo real, Lepetit e Fua [67], motivados pelo trabalho de Amit e Geman [68], propuseram a utilização de *Randomized Trees*, para efetuar a classificação. Com o emprego desse classificador, uma espécie de árvore de decisão onde cada nodo da árvore efetua um simples teste que divide o espaço de dados a ser classificado, neste caso, o espaço é definido pelos *patches* da imagem que representam

uma característica da pose. Nessa estrutura, um atributo é classificado percorrendo a árvore a partir da raiz em direção das folhas. Quando esse atributo chega até a folha, uma probabilidade é associada ao mesmo. Ao final, utilizando algum tipo de combinação das probabilidades associadas a todos os atributos, é possível determinar a pose do objeto.

## 2.4 Considerações sobre a revisão bibliográfica

Como pode ser visto neste capítulo, existem diferentes abordagens para resolver o problema de detecção de face e pose. Na prática, essas abordagens não são implementadas individualmente, mas sim combinadas, formando sistemas híbridos, cujo objetivo é melhorar o resultado da estimativa da pose facial. A forma de aquisição da imagem e o custo computacional são dois aspectos que devem ser considerados na implementação de um algoritmo para a detecção da face e estimativa da pose. Em relação à forma de aquisição das imagens, existem diversos algoritmos robustos que utilizam câmeras estéreo [69] [70] [71] para estimar a pose de objetos 3D. Esses algoritmos possuem a informação de profundidade da cena que auxilia muito na resolução do problema. Considerando, entetanto, que o foco do trabalho aqui proposto é a utilização de *webcams*, essas abordagens não podem ser utilizadas.

O outro fator importante é o custo computacional. Para minimizar esse problema, algumas técnicas [65] [67] efetuam um treinamento *offline* prévio implicando em um tempo elevado para gerar o classificador, mas cujo rastreamento (*online*) é efetuado em tempo real. Essas últimas abordagens são as que têm recebido maior interesse por parte dos pesquisadores dedicados ao estudo de detecção de objetos 3D, especialmente depois que as máquinas domésticas ganharam alto poder de processamento. O inconveniente dessas propostas é que, geralmente, não são genéricas, mas sim específicas para um único objeto utilizado na fase de treinamento.

Durante a revisão, notou-se que os sistemas baseados na lógica do rastreamento têm recebido muita atenção nos últimos vinte anos [49]. Nesses sistemas, a inicialização e a estabilização do rastreamento são dois pontos cruciais. De uma forma geral, a inicialização quase sempre é feita por um detector de faces em posição frontal. Já a estabilização do rastreamento com o objetivo de evitar a propagação do erro acumulado, foi proposta por alguns autores [63] [64] mas continua sendo um

Tabela 2.1 – Técnicas utilizadas pelos autores citados na revisão bibliográfica

| Referência | Segmentação de cor (pele) | Detecção de face frontal | Estimativa da Pose | | Ano |
|---|---|---|---|---|---|
| | | | Discreta | Contínua | |
| [8] | | x | | | 1973 |
| [19] | | x | | | 1995 |
| [60] | | | | x | 1995 |
| [16] | | x | x | | 1996 |
| [37] | | x | | | 1997 |
| [47] | | | | x | 1997 |
| [34] | | x | | | 1998 |
| [43] | x | x | | x | 1998 |
| [14] | x | | | | 1999 |
| [40] | x | x | | | 1999 |
| [20] | x | x | | | 1999 |
| [50] | | x | | x | 2000 |
| [44] | x | x | x | | 2000 |
| [38] | x | x | | | 2001 |
| [63] | | x | | x | 2001 |
| [39] | | x | | | 2001 |
| [38] | x | x | | | 2001 |
| [33] | x | x | | | 2001 |
| [62] | | | | x | 2001 |
| [55] | | | x | | 2002 |
| [29] | | x | | | 2002 |
| [51] | | x | | x | 2002 |
| [18] | x | | | | 2002 |
| [15] | x | | | | 2003 |
| [17] | x | | | | 2003 |
| [21] | | x | x | | 2003 |
| [41] | | | x | | 2004 |
| [64] | | | | x | 2004 |
| [65] | | | | x | 2004 |
| [27] | | x | | | 2004 |
| [32] | | x | | | 2004 |
| [52] | | x | | x | 2006 |
| [31] | | x | | | 2006 |
| [25] | | x | | | 2007 |

problema em aberto.

A Tabela 2.1 faz um resumo das técnicas utilizadas pelos autores citados nesse capítulo. No próximo capítulo será descrito o modelo proposto nessa dissertação para efetuar a detecção e o rastreamento da pose da cabeça, devidamente seguido de sua discussão.

# Capítulo 3

# Método Proposto

De acordo com o objetivo desse trabalho, pretende-se rastrear a posição 3D da cabeça, a partir de uma seqüência de imagens de um indivíduo posicionado à frente de uma câmera de vídeo. Conforme visto na revisão bibliográfica, diferentes abordagens foram desenvolvidas para resolver o problema de detecção de faces e de estimativa da pose da cabeça. Neste capítulo, será descrito o modelo que se pretende adotar neste trabalho, baseado em um conjunto de técnicas para particionar o problema do rastreamento da posição 3D da cabeça em duas etapas principais.

A primeira etapa é formada pelo detector de faces, que serve para localizar a face na imagem assim que ela entre na região visível da câmera. Nessa etapa, a idéia é combinar o algoritmo de Viola e Jones [26], considerado um referencial na detecção de faces, com um pré-processamento baseado em detecção de cor de pele, para reduzir o custo computacional e diminuir o número de falsos positivos. Os resultados desses procedimentos servem de inicialização para a segunda etapa, onde um rastreador 3D da cabeça identifica a pose da mesma. O fluxograma do modelo proposto é ilustrado na Figura 3.1.

A seguir, essas etapas são descritas em duas seções distintas. A primeira descreve o algoritmo para detecção da face com a adição da segmentação da pele, e a segunda descreve o algoritmo para rastreamento da pose da face.

## 3.1 Detecção da Face

O algoritmo utilizado nesse trabalho foi baseado no trabalho de Viola e Jones [26], cuja implementação está disponível na interface de programação da

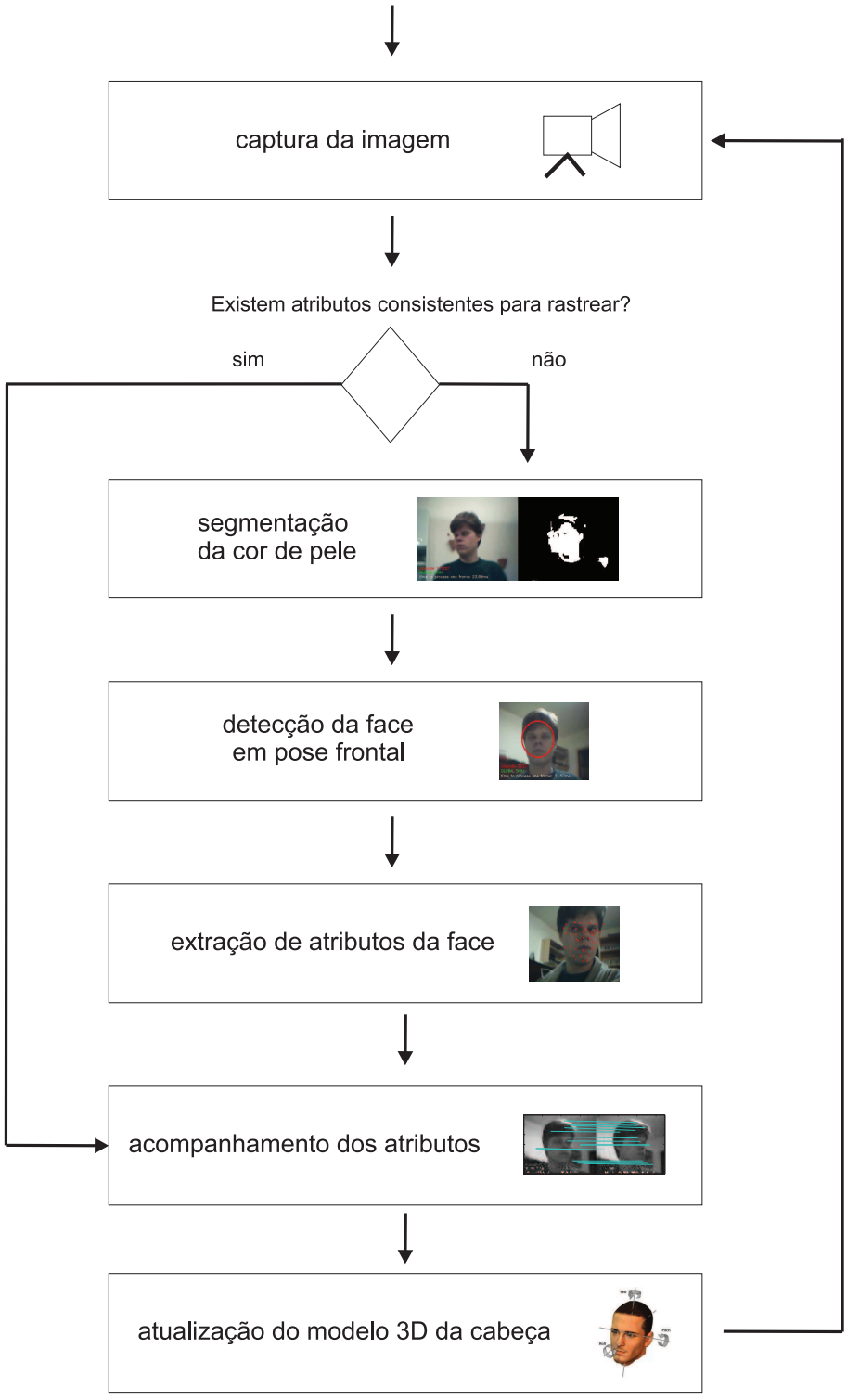


Figura 3.1 – Fluxograma do modelo proposto.

biblioteca para visão computacional OpenCV [72]. Ele tem uma precisão bastante alta e apresenta custo computacional relativamente baixo, permitindo execução em tempo real, nos computadores atuais. Entretanto, essa técnica consome muito processamento, limitando a utilização de outros aplicativos de forma simultânea

(o que é fundamental em aplicações de video-conferência, interfaces homem-computador, entre outras.).

Conforme citado no início desse capítulo, uma redução do espaço de busca foi desenvolvida para diminuir o processamento da etapa de detecção da face. Como conseqüência, falsos positivos que eram detectados em regiões não consideradas pele também serão removidos pela redução do espaço de busca, melhorando a qualidade do detector e reduzindo o custo computacional. A descrição desse processo será feita a seguir.

### 3.1.1 Detecção de Pele

A maioria das técnicas descritas na revisão bibliográfica efetuam a detecção de faces com classificadores aplicados a subregiões da imagem. Essas subregiões possuem um tamanho relativo à face a ser detectada e são verificadas através de uma janela deslizante, que percorre exaustivamente a imagem. Dessa foma, o custo computacional está intimamente ligado ao número de janelas necessárias para verificar toda a imagem. Por exemplo, para uma imagem de tamanho $320 \times 240$ pixels é possível posicionar uma janela de tamanho $20 \times 20$ pixels em 66521 posições diferentes, sendo o detector aplicado para cada uma dessas posições.

Uma forma de diminuir o número de testes é restringir a região de busca, descartando porções da imagem que não podem conter faces, de acordo com algum critério adequado. Um desses critérios, bastante utilizado e referenciado na revisão bibliográfica deste trabalho, é a cor de pele. Dessa forma, regiões que não são consideradas pele devem ser ignoradas pelo classificador.

A discriminação dos pixels em duas classes (pele e não-pele) utilizada neste trabalho é realizada aplicando uma regra simples e de rápida execução. A regra utilizada aqui é uma versão simplificada daquela proposta por Nusirwan et al. [73]. Esses autores propuseram um conjunto sofisticado de regras extraídas a partir da análise estatística de um conjunto de treinamento. Durante os testes realizados com essas regras em nosso sistema, verificou-se que, em condições de variação de iluminação, diversas regiões de pele foram mal classificadas. Para resolver esse problema de falsos negativos, algumas restrições foram relaxadas. A regra proposta nesta dissertação, diferentemente de outras propostas mais complexas [13], não visa detectar a pele com alto grau de precisão, mas sim descartar regiões cuja informação

cromática difere significativamente do modelo esperado para cor de pele.

Dada uma imagem de entrada colorida no formato $RGB$, $R$ representa a intensidade do canal vermelho, $G$ do canal verde, e $B$ do canal azul. Uma imagem binária $S$, que define as regiões consideradas candidatas à pele na imagem de entrada, é definida através de:

$$S = R > G \wedge R > B, \tag{3.1}$$

ou seja, pixels candidatos a pertencerem à pele são aqueles que apresentam o componente vermelho maior do que o verde e do que o azul. Salienta-se que essa regra pode gerar diversos falsos positivos (ou seja, pixels que não têm cor de pele e são classificados como pele). Mesmo assim, ela reduz consideravelmente o espaço de busca utilizado na etapa de detecção de faces, a qual usa um classificador que explora apenas características geométricas da mesma.

Um exemplo de detecção de pixels com tom de pele usando a Equação (3.1) é ilustrado na Figura 3.2. Como se pode perceber, pixels da pele (mais especificamente, da face) foram corretamente identificados, mas alguns objetos do fundo da cena também foram identificados como candidatos, sem contudo serem, de fato, pontos da pele.

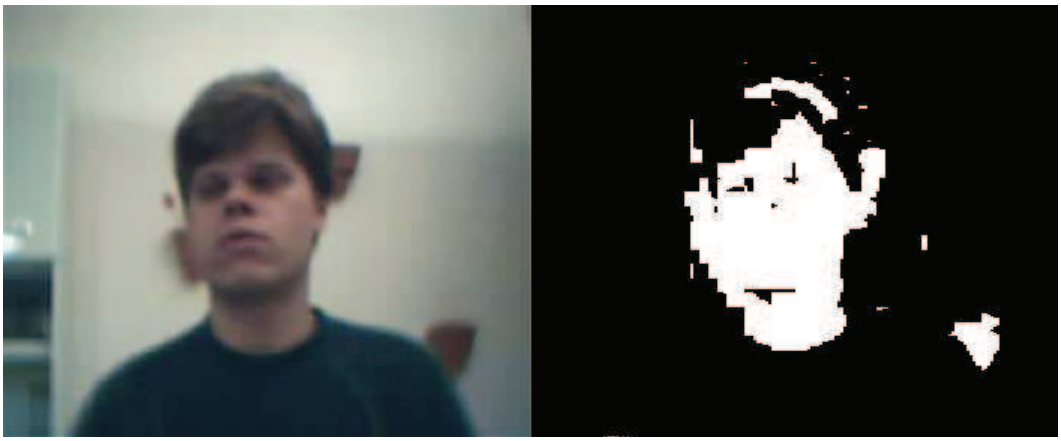


Figura 3.2 – a) imagem colorida. b) imagem binária obtida após a aplicação da regra de segmentação baseada na cor da pele.

### 3.1.2 Detecção de características espaciais da face

Conforme já foi dito, o método utilizado nesse trabalho para detectar faces usando características geométricas é baseado no trabalho de Viola e Jones [26], onde o classificador é construído com o algoritmo AdaBoost utilizando *haar-like features*. O algoritmo AdaBoost constrói um classificador não-linear complexo através da combinação linear de um conjunto de hipóteses fracas, também conhecidas como *weak classifiers*.

O detector de faces implementado utiliza a biblioteca OpenCV, que tem uma interface de programação para detecção de objetos empregando *haar-like features*. Para detectar faces em seqüências de vídeo, o detector percorre o quadro de entrada com uma janela de tamanho $20 \times 20$ pixels e aplica o detector descrito nessa seção. Para detectar faces de tamanhos variados, um fator de escala de 10% é aplicado sucessivamente às *haar-like features*, variando suas dimensões até o limite do tamanho da imagem.

Esse detector foi treinado por Lienhart et al. [74] a partir de um conjunto de 3000 imagens com exemplos negativos e 5000 imagens com exemplos positivos. Os exemplos negativos são imagens diversas que não contêm faces. Os exemplos positivos são derivados de um conjunto inicial de 1000 faces manualmente extraídas de um conjunto de imagens. A partir dessas 1000 faces originais, foram gerados os 5000 exemplos positivos, onde rotações aleatórias de $\pm 10°$, escalas aleatórias de $\pm 10\%$, espelhamento aleatório, e deslocamento aleatório de $\pm 1$ pixel foram aplicados aos exemplos originais.

Um total de vinte estágios foi configurado e treinado. Cada estágio foi treinado com o objetivo de descartar 50% dos exemplos não-face e, ao mesmo tempo, obter uma taxa de detecção de 99.9% (tais taxas são informadas ao algoritmo *AdaBoost* durante o treinamento). Dessa forma, ao final de vinte estágios, a taxa de detecção esperada é de aproximadamente $0.999^{20}$ e a taxa de falsos positivos aproximada de $0.5^{20}$. Maiores detalhes do algoritmo de treinamento pode ser visto na seção 2.2. Para melhorar a taxa de detecção, os autores incluiram um novo conjunto de atributos *haar-like* rotacionados. A Figura 3.3 ilustra os modelos primitivos desses atributos.

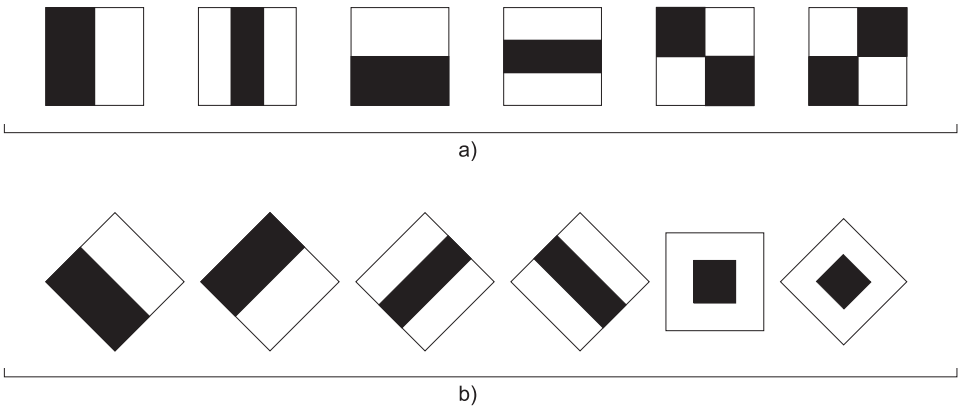


Figura 3.3 – Protótipos de atributos *Haar-like* utilizadas por Lienhart *et al.* [74] para o treinamento do detector. a) Conjunto básico de atributos utilizado por Viola e Jones [26]. b) Extensão proposta em [74]. A utilização de ambos os conjuntos melhora a taxa de detecção em 10%.

### 3.1.3 Integração dos modelos geométrico e de cor

Visando otimizar o processo de detecção de face descrito por Viola e Jones, utiliza-se a informação de cor de pele descrita na seção 3.1.1 para reduzir o espaço de busca onde será aplicada a detecção com base em atributos espaciais. A lógica do modelo de cor utilizado segue o princípio da cascata de classificadores proposto por Viola e Jones, segundo o qual, o teste de cor deve gerar poucos falsos negativos (ou seja, poucas faces são eliminadas neste estágio), mas pode gerar diversos falsos positivos (ou seja, deixa passar diversos objetos que não são faces). As regiões que passam no teste de cor são então processadas pelo algoritmo de Viola e Jones, que refina a detecção de faces com base nos atributos espaciais (*haar-like features).*

Dada uma imagem colorida $I$, o teste dado pela Equação (3.1) gera uma imagem binária $S$, contendo valor 1 nos pixels considerados como cor de pele, e 0 nos demais. Dada uma subregião retangular $J$, representando a caixa limitadora de uma face, espera-se que um percentual significativo de pixels em seu interior seja considerado como cor de pele. Por outro lado, como olhos, pelos e cabelo normalmente não são detectados como cor de pele, espera-se que esse percentual não seja muito alto também. Sendo $m \times n$ o tamanho da região $J$, ela é considerada candidata a face se

$$T_1 < \frac{1}{nm} \sum_{(x,y) \in R} S(x,y) < T_2, \tag{3.2}$$

onde $0 < T_1 < T_2 < 1$ são limiares que represetam os percentuais mínimo e

máximo de pixels de cor de pele exigidos para considerar a região $J$ candidata. Experimentalmente, definiu-se $T_1 = 0.5$, $T_2 = 0.95$.

O somatório da Equação (3.2) pode ser calculado eficientemente usando imagens integrais. A imagem integral é uma representação intermediária, na qual cada pixel $iS(x, y)$ contém a soma dos pixels acima, à esquerda, e da própria posição $(x, y)$ da imagem original $S$. Uma vez criada a imagem integral, cada somatório pode ser calculado por meio de quatro consultas à imagem $iS$.

Formalmente a imagem integral é dada por

$$iS(x, y) = \sum_{x' \leq x, y' \leq y} S(x', y'), \tag{3.3}$$

onde $iS(x, y)$ é o valor da imagem integral no ponto $(x, y)$, e $S(x, y)$ é o valor da imagem original no mesmo ponto. É possível gerar a imagem integral, percorrendo uma única vez a imagem original e usando o seguinte par de recorrências:

$$\begin{aligned} s(x, y) &= s(x, y - 1) + S(x, y) \\ iS(x, y) &= iS(x - 1, y) + s(x, y) \end{aligned}, \tag{3.4}$$

onde $s$ é a soma cumulativa da linha, $s(x, -1) = 0$ e $iS(-1, y) = 0$.

Como não se sabe onde candidatos a face podem estar em uma dada imagem, é necessário fazer uma varredura de todos os pixels por intermédio de uma janela deslizante $J$, e aplicar o teste descrito em (3.2) para cada janela. Salienta-se também que o teste para cor de pele dado pela condição (3.1) é extremamente rápido, pois envolve uma comparação direta entre canais de cor. Assim, todo pré-processamento baseado em cor de pele apresenta um custo computacional $\mathcal{O}(N \times M)$, onde $N \times M$ é a dimensão total da imagem que está sendo analisada.

O algoritmo final, combinando informação de cor com atributos geométricos, consiste em varrer a imagem original $I$ através de janelas deslizantes retangulares $J$, conforme indicado na seção 3.1.2. Entretanto, os atributos baseados nas *haar-like features* no interior de $J$ são calculados apenas quando o teste (3.2) for satisfeito, o que tende a reduzir significativamente a quantidade de cálculos dos atributos espaciais.

A Figura 3.4 ilustra o resultado do algoritmo completo para detecção de faces

frontais, combinando o pré-processamento de cores com a validação por atributos geométricos. Conforme ilustra a figura, falsos positivos são removidos quando a região de busca é restringida pelo filtro baseado na cor de pele.

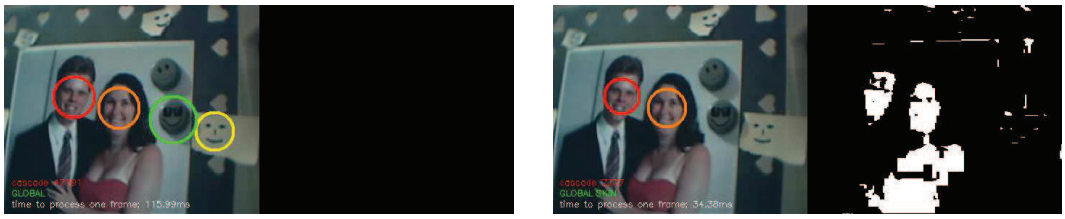


Figura 3.4 – a) detecção de faces sem restrição do espaço de busca. b) detecção de faces com restrição do espaço de busca.

Com o emprego do processo descrito nessa seção, é possível detectar faces em seqüências de vídeo a uma taxa de 25 quadros por segundo, utilizando um computador Pentium 4 com processador 3GHz. A posição de cada face detectada nesse processo serve como inicialização para a próxima etapa do algoritmo. Nessa próxima etapa, descrita na seção seguinte, a pose da cabeça é obtida através do rastreamento de atributos da cabeça no espaço 3D.

## 3.2 Rastreamento da pose

Conforme discutido anteriormente, a pose da cabeça é uma informação muito importante, que pode ser utilizada em diversas aplicações, como sistemas de segurança, *videogames*, video-conferências, etc. Uma revisão bibliográfica das abordagens existentes para efetuar a estimativa da pose da cabeça foi explicitada no capítulo 2 e, a partir dessas, foi escolhida uma abordagem na qual o rastreamento de atributos da face no plano da imagem é utilizado para descrever o movimento da cabeça no espaço 3D. A escolha dessa modalidade foi influenciada pela capacidade de a mesma permitir estimar a pose da cabeça de forma contínua e com boa precisão. Segundo [10], essa abordagem tem a desvantagem de necessitar a inicialização da posição da cabeça com uma pose previamente conhecida. Entretanto, esse problema é resolvido à medida que acontece a detecção da cabeça em pose frontal, pela utilização do detector de faces descrito na seção anterior.

O algoritmo para estimar a pose, em linhas gerais, é inicializado com a detecção

da face em pose frontal. Após a detecção, um modelo rígido 3D que representa a cabeça é posicionado centrado na face. Com o modelo posicionado, inicializa-se um conjunto inicial de atributos 2D na imagem da face. Esses atributos, por sua vez, são rastreados quadro a quadro durante a seqüência de vídeo. Partindo do deslocamento desses atributos é possível estimar uma transformação do modelo da cabeça que melhor descreve as translações 2D locais de cada um dos atributos. Ao longo dessa secção, serão descritas com mais detalhes cada uma das etapas do procedimento para estimar a pose.

Primeiramente, é necessário definir um mapeamento do espaço 3D no plano da imagem. Qualquer ponto no espaço é representado no plano da imagem por meio de um tipo de projeção. No trabalho aqui desenvolvido foi utilizado o modelo de câmera com projeção perspectiva. Nesse modelo de câmera [75], um ponto no espaço é representado por $\boldsymbol{M} = [x, y, z]^T$ e definido num sistema de coordenadas euclidiano. O ponto correspondente a $\boldsymbol{M}$ projetado no plano da imagem é representado por $\boldsymbol{m} = [u, v]^T$. Assumindo que a matriz de projeção da câmera depende apenas da distância focal[1] $\phi$,temos o mapeamento dos pontos $\boldsymbol{M}$ em $\boldsymbol{m}$ na forma

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{\phi}{z} \begin{bmatrix} x \\ y \end{bmatrix}, \tag{3.5}$$

assumindo que a origem do sistemas cartesianos de $\boldsymbol{m}$ e $\boldsymbol{M}$ estão no centro da imagem. Além disso, assume-se que o componente $z$ de $\boldsymbol{M}$ está apontando para dentro da imagem, ou seja, quanto mais longe estiver a cabeça rastreada pela câmera, maior será o $z$. A Figura 3.5 ilustra essa relação.

### 3.2.1 Modelo rígido 3D

Para que se possa estimar a pose da cabeça com base na translação dos pontos $\boldsymbol{m}$, é preciso associar esses com os pontos $\boldsymbol{M}$ no espaço e, estes últimos, por sua vez, devem ser relacionados com a geometria da superfície de um modelo 3D da cabeça. Dessa forma, baseado na alteração da posição do conjunto de pontos $\boldsymbol{M}$, ou seja, na alteração do modelo 3D, é possível estimar a transformação sofrida pela cabeça. A Figura 3.6 ilustra essa relação, onde o ponto $\boldsymbol{M}_i$ no quadro $t$, em conseqüência

[1] Algumas câmeras possuem uma distância focal diferente para cada um dos eixos $u$ e $v$. Para essas câmeras é preciso utilizar uma distância focal individual ($\phi_u$ e $\phi_v$) para cada eixo.

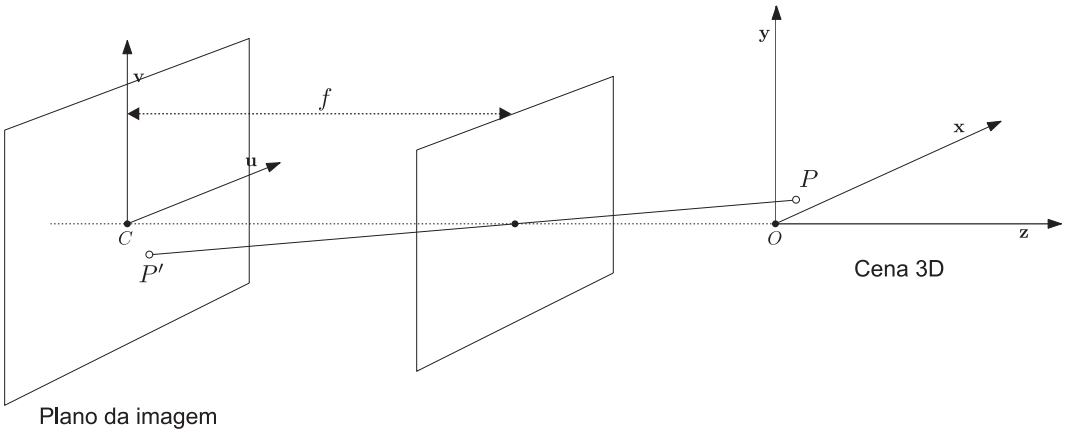


Figura 3.5 – Relação geométrica entre o sistema de coordenadas do plano da imagem e o sistema de coordenadas da cena 3D. O ponto $P'$ é a projeção na imagem do ponto $P$ da cena 3D dada a distância focal $f$.

do movimento da cabeça, é movido na cena 3D para $\boldsymbol{M_i'}$ no quadro $t+1$. Após o movimento da cabeça, $\boldsymbol{m_i}$, que é a projeção no plano da imagem de $\boldsymbol{M_i}$, sofre uma translação sobre esse mesmo plano, parando em $\boldsymbol{m_i'}$ no quadro $t+1$.

Essa abordagem utiliza um modelo rígido conhecido *a priori* (*Model-Based* [49]). A utilização do modelo rígido torna mais fácil a modelagem da transformação sofrida pela cabeça, ao contrário dos modelos flexíveis, que precisam de um conjunto maior de parâmetros para recuperar a transformação entre um quadro e outro. A desvantagem dos modelos rígidos é que eles não são ideais para capturar deformações do próprio objeto, como por exemplo, expressões faciais.

Existem diversas propostas de modelos para representar a face. Xiao et al. [51] utilizaram um modelo em formato de cilindro, enquanto que Hager e Belhumeur [76] e Liang et al. [77] utilizam um plano alinhado com os olhos e boca para representar a cabeça no espaço 3D. Um modelo bastante conhecido é o Candide [53], que representa a face por meio de 75 vértices e 100 triângulos. Uma desvantagem do Candide é conter polígonos somente na região da face, de tal forma que as laterais da cabeça não são representadas pelo modelo. Assim sendo, para a técnica proposta nessa dissertação, foi realizada uma pequena modificação no Candide, a qual adicionou novos polígonos nas regiões laterais da cabeça. A Figura 3.7 ilustra a máscara original e a modificada. O tamanho 3D da máscara é fixo (assume tamanho médio padrão da face/cabeça).

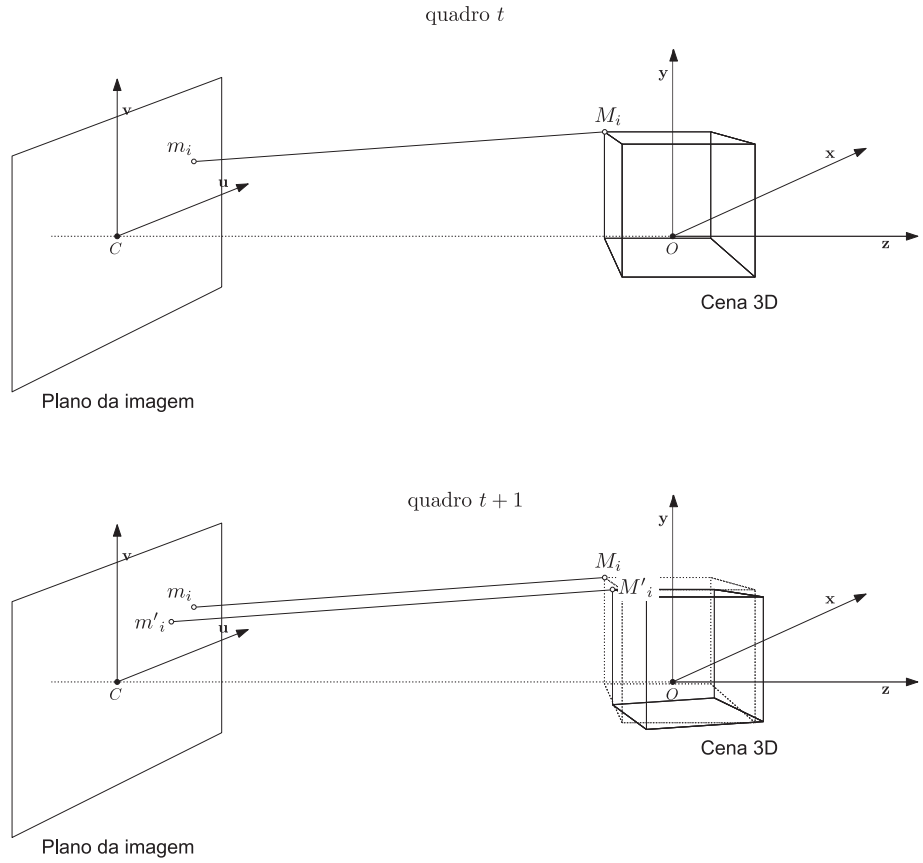


Figura 3.6 – Para cada atributo rastreado no plano da imagem, existe um ponto correlacionado no modelo rígido (neste gráfico, para facilitar a visualização, está sendo utilizado um cubo ao invés da cabeça). A variação da posição de $\boldsymbol{M}_i$ na cena 3D implica em uma translação de $\boldsymbol{m}_i$ no plano da imagem.

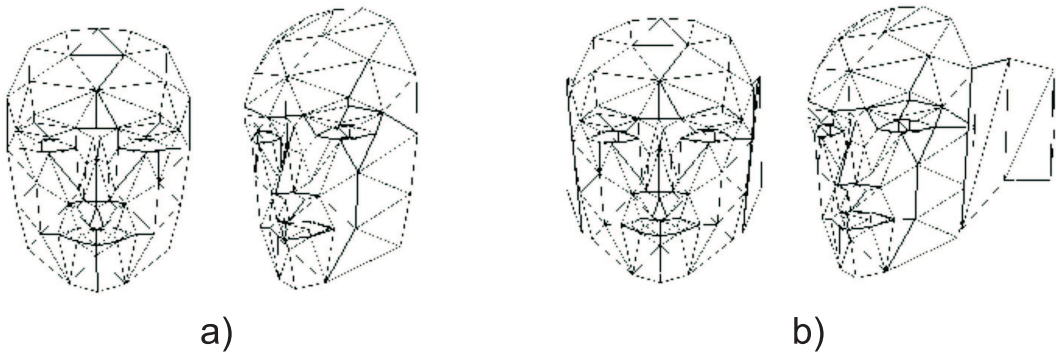


Figura 3.7 – a) modelo Candide original. b) modelo Candide com novos polígonos na lateral da cabeça.

### 3.2.2 Cálculo da transformação 3D

Como dito anteriormente, a partir da variação da posição dos pontos representado por $\boldsymbol{M}$, visualizados e representados na imagem por intermédio de $\boldsymbol{m}$,

é possível estimar a variação da pose da cabeça. Matematicamente falando, dada a posição da cabeça no espaço 3D no quadro $t$, representada por $\boldsymbol{M}$, queremos saber qual é a nova posição da mesma no quadro $t+1$, representada por $\boldsymbol{M'}$, onde $\boldsymbol{M} = [x, y, z]^T$ e $\boldsymbol{M'} = [x', y', z']^T$. Em notação de coordenadas homogêneas [75] temos $\boldsymbol{M_h} = [x, y, z, 1]^T$, e a nova posição $\boldsymbol{M'_h}$ é obtida utilizando uma transformação rígida 3D, que pode ser modelada através de

$$\boldsymbol{M'_h} = \boldsymbol{G}\boldsymbol{M_h}, \tag{3.6}$$

onde $\boldsymbol{G}$ é uma matriz de transformação na forma

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{T} \\ 0 & 1 \end{bmatrix}, \tag{3.7}$$

que inclui tanto rotação como translação. $\boldsymbol{R}_{3x3}$ é a matriz de rotação e $\boldsymbol{T}_{3x1}$ é o vetor de translação 3D, ambos com três graus de liberdade.

Se $\boldsymbol{G}$ contiver uma matriz de rotação tradicional, contendo senos e cossenos, será preciso resolver um sistema não-linear para achar os parâmetros que definem os ângulos nos eixos $x$,$y$,$z$ respectivamente. Isso é ruim, pois aumenta a complexidade do problema. Porém, existe uma forma de representar a matriz $\boldsymbol{G}$ de modo que não envolva senos e cossenos. Essa representação é conhecida como *twist* [51], na qual $\boldsymbol{G}$ é aproximada através da série infinita

$$\boldsymbol{G} = e^{\boldsymbol{\xi}} = \boldsymbol{I} + \boldsymbol{\xi} + \frac{1}{2!}\boldsymbol{\xi}^2 + \frac{1}{3!}\boldsymbol{\xi}^3 + ... \tag{3.8}$$

onde $\boldsymbol{\xi}$ é a matriz

$$\xi = \begin{bmatrix} 0 & -\omega_z & \omega_y & d_x \\ \omega_z & 0 & -\omega_x & d_y \\ -\omega_y & \omega_x & 0 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.9}$$

Como o objetivo é manter o sistema linear, a série é truncada e são utilizados apenas os termos de primeira ordem. Os termos de segunda ordem poderiam ser mantidos, porém, isso causaria um aumento da complexidade do sistema final e,

possivelmente, o sistema tornaria-se mais sensível a presença de *outliers*[2] causados pelo erro de rastreamento dos atributos 2D no plano da imagem. Por isso, optou-se por utilizar apenas os termos de primeira ordem e aplicar um refinamento posterior. O processo de refinamento é descrito na seção 3.2.5.

A matriz de transformação é então aproximada como $\boldsymbol{G} \approx \boldsymbol{I} + \boldsymbol{\xi}$, ou seja, será

$$\boldsymbol{G} \approx \begin{bmatrix} 1 & -\omega_z & \omega_y & d_x \\ \omega_z & 1 & -\omega_x & d_y \\ -\omega_y & \omega_x & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.10}$$

Uma vez definida a transformação, o problema passa a ser a obtenção dos parâmetros da matriz $\boldsymbol{G}$, representados em notação vetorial como $\boldsymbol{\alpha} = [\omega_x, \omega_y, \omega_z, d_x, d_y, d_z]^T$.

É nesse momento que se utiliza a informação obtida no rastreamento dos atributos $\boldsymbol{m}$ no plano da imagem, sendo necessário achar $\boldsymbol{\alpha}$ tal que

$$\boldsymbol{m'} = f(\boldsymbol{\alpha}, \boldsymbol{M}), \tag{3.11}$$

onde $f$ é a função que combina a transformação rígida $\boldsymbol{G}$ de $\boldsymbol{M}$ em $\boldsymbol{M'}$ com a projeção de $\boldsymbol{M'}$ em coordenadas de imagem $\boldsymbol{m'}$. Ou seja, se deseja encontrar $\boldsymbol{\alpha}$ de forma que, após a transformação dos pontos $\boldsymbol{M}$ em $\boldsymbol{M'}$, a projeção na imagem desses últimos seja a mais próxima possível de $\boldsymbol{m'}$. Efetuando o movimento rígido e a projeção, obtém-se

$$f(\boldsymbol{\alpha}, \boldsymbol{M}) = \frac{\phi}{-x\omega_y + y\omega_x + z + d_z} \begin{bmatrix} x - y\omega_z + z\omega_y + d_x \\ x\omega_z + y - z\omega_x + d_y \end{bmatrix}. \tag{3.12}$$

Na prática, teremos um conjunto de $N$ pontos $\boldsymbol{M}_i$, $i = 1, .., N$, que gerará $2N$ restrições conforme a equação (3.11), onde $\boldsymbol{\alpha}$ é a incógnita. De fato, o sistema de equações gerado usando os $N$ atributos é linear em relação a $\boldsymbol{\alpha}$, possuindo $2N$

[2] *Outlier* é o ponto que não está de acordo com um critério pré-determinado/desejado. *Inlier*, conceito oposto de *outlier*, é o ponto que está de acordo com um critério pré-determinado ou esperado.

equações e 6 incógnitas, e pode ser escrito na forma

$$\boldsymbol{A}\boldsymbol{\alpha} = \boldsymbol{b}, \tag{3.13}$$

onde $\boldsymbol{A}$ e $\boldsymbol{b}$ são obtidos depois de várias operações algébricas, isolando as incógnitas da equação (3.12), gerando

$$\begin{bmatrix} -u_1y_1 & \phi z_1 + u_1x_1 & -\phi y_1 & \phi & 0 & -u_1 \\ -\phi z_1 - v_1y_1 & v_1x_1 & \phi x_1 & 0 & \phi & -v_1 \\ -u_2y_2 & \phi z_2 + u_2x_2 & -\phi y_2 & \phi & 0 & -u_2 \\ -\phi z_2 - v_2y_2 & v_2x_2 & \phi x_2 & 0 & \phi & -v_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -u_Ny_N & \phi z_N + u_Nx_N & -\phi y_N & \phi & 0 & -u_N \\ -\phi z_N - v_Ny_N & v_Nx_N & \phi x_N & 0 & \phi & -v_N \end{bmatrix} \begin{bmatrix} \omega_x, \\ \omega_y, \\ \omega_z, \\ d_x, \\ d_y, \\ d_z \end{bmatrix} = \begin{bmatrix} u_1z_1 - \phi\, x_1 \\ v_1z_1 - \phi\, y_1 \\ u_2z_2 - \phi\, x_2 \\ v_2z_2 - \phi\, y_2 \\ \vdots \\ u_Nz_N - \phi\, x_N \\ v_Nz_N - \phi\, y_N \end{bmatrix} \tag{3.14}$$

Na teoria, o rastreamento de três atributos seria suficiente para a determinação de $\boldsymbol{\alpha}$, mas os erros inerentes ao processo de rastreamento tendem a deturpar a estimativa dos parâmetros da pose. Assim, utiliza-se uma quantidade maior de atributos, resolvendo o sistema (3.13) de forma aproximada. Uma forma de resolver esse sistema sobredeterminado é através de mínimos quadrados, porém essa opção tem o problema de ser muito influenciada por *outliers*. Uma outra técnica para solução de sistemas sobredeterminados é o algoritmo RANSAC [78], o qual calcula a solução descartando os *outliers*. O algoritmo RANSAC foi a opção escolhida para resolver o sistema (3.13) e será descrito brevemente na próxima seção deste capítulo.

### 3.2.3 Eliminação de *outliers* utilizando RANSAC

A solução tradicional por mínimos quadrados é simples de implementar, mas é bastante afetada por *outliers*, que são pontos mal rastreados e ocorrem com freqüência por causa de oclusões e variação de iluminação. Assim, visando uma solução mais robusta, o algoritmo RANSAC [78] foi empregado para extrair os *outliers*. Esse algoritmo verifica, de forma iterativa, a presença de atributos muito distantes da solução ótima, e descarta-os para que sua influência não seja

significativa. As etapas do algoritmo RANSAC são sumarizadas no Algoritmo 1.

**Algoritmo 1**: RANSAC utilizando 3 pontos

Dado um conjunto de pontos $X = \{M_1, M_2, ..., M_N\}$:
1: Selecione aleatoriamente um conjunto de pontos $X_s = \{M_i, M_j, M_k\}$, necessários para calcular a transformação.
2: Resolva o sistema utilizando $X_s$ para achar os parâmetros da transformação.
3: Determine $Q$ como o subconjunto de pontos pertencentes a $X$ e que estão de acordo com a transformação obtida no passo anterior, dada uma tolerância $\psi$.
4: Se a fração do número de *inliers* (elementos de $Q$) por $N$ (total de pontos do conjunto $X$) for maior que um limiar pré-definido $\lambda$, estime a transformação (via mínimos quadrados) utilizando todos os elementos de $Q$ e termine o algoritmo.
5: Caso contrário, repita os passos 1 até 4 (máximo de $K$ vezes).

Nesse algoritmo é necessário definir alguns parâmetros. O primeiro deles é o número mínimo de pontos para calcular a transformação. Como dito anteriormente, são necessárias seis equações para resolver o sistema (3.13) de forma exata. Cada ponto gera duas equações, portanto são necessários no mínimo três pontos. No algoritmo RANSAC existem dois parâmetros que definem o critério de parada. Um dos critérios de parada é o número de *inliers* desejados. Quanto mais próximo for o número de *inliers* do número total de pontos, melhor será o resultado da transformação. Porém, já é sabido que existem pontos mal estimados e precisam ser descartados, sendo esta a utilidade do RANSAC. Não é possível definir um valor teórico para o número mínimo de *inliers*, e o limiar utilizado nesta dissertação foi estimado a partir de testes com seqüências de vídeo. Após uma análise empírica dos resultados, esse limiar, $\lambda$, foi fixado em 70% de *inliers* sobre o total de pontos. O outro critério de parada é o número limite de iterações $K$.

É possível estimar $K$, de acordo com uma probabilidade desejada [78], de forma a obter o número mínimo de iterações necessárias que sorteie pelo menos um subconjunto com todos os elementos sendo *inliers*. Porém, essa estimativa depende do número de *inliers* esperados na distribuição de dados, e, como já dito anteriormente, essa informação não é conhecida em nosso sistema. Então, foi utilizada uma estimativa baseada na análise do erro obtido após aplicar a transformação aos pontos em cada iteração do RANSAC. A partir da análise do erro, feita durante a realização de testes, foi definido $K = 46$. Uma desvantagem do uso do RANSAC é que, sendo não-determinístico, não é possível afirmar que ao final de $K$ iterações a solução será ótima. Para os casos onde, mesmo após $K$ iterações,

não foi possível encontrar uma solução que satisfizesse o critério de número mínimo de *inliers*, são utilizados os valores de $\boldsymbol{\alpha}$ obtidos na iteração que alcançou o maior número de *inliers* para calcular a solução de mínimos quadrados.

Ainda é preciso definir o limiar $\psi$ que classifica um ponto como *inlier* a cada iteração do RANSAC. O critério de erro $E_i$ para cada ponto $\boldsymbol{m}_i$ é a distância Euclidiana entre a posição do ponto projetado, após a transformação com o conjunto mínimo de pontos selecionados aleatoriamente, e a posição do ponto correspondente rastreado na imagem. O limiar de erro $\psi$ ficou definido em quatro pixels e foi estipulado empiricamente durante a realização de testes com diferentes valores.

### 3.2.4 Rastreamento e atualização de atributos no plano da imagem

Para inicializar o rastreamento da pose, primeiramente é preciso posicionar o modelo rígido 3D de acordo com a posição da face detectada pelo emprego do algoritmo descrito no início deste capítulo. O detector de faces informa o centro da face em relação ao centro da imagem e o raio da circunferência que circunscreve a face. A posição e o tamanho da face informada é relativa a posição e ao tamanho da janela deslizate utilizada pelo algoritmo de detecção de faces. O centro de rotação do modelo 3D é transladado para a posição determinada pelo vetor $P(x_f, y_f, z_f)$. Primeiramente, $z_f$ é obtido pela relação $z_f = \frac{L\phi}{d}$, onde $L$ é uma constante que define a largura do modelo 3D e $d$ é o diâmetro da circunferência que circunscreve a face. Uma vez definido $z_f$, obtém-se $x_f = \frac{u_0 z_f}{\phi}$ e $y_f = \frac{v_0 z_f}{\phi}$, onde $(u_0, v_0)$ representam o centro da face em coordenadas de imagem.

Para estimar a pose do objeto 3D (cabeça) a partir da Equação (3.11), é necessário encontrar os parâmetros $\boldsymbol{\alpha}$ de acordo com (3.13). Mas, para isso, é preciso encontrar $\boldsymbol{m}'$ no quadro $t+1$ a partir de $\boldsymbol{m}$ no quadro $t$. No primeiro quadro, $t = 0$, após a detecção da face e a translação do modelo para $P$, o sistema inicializa o conjunto de atributos na região da imagem compreendida pela face. Esses atributos serão rastreados a cada novo quadro, utilizando a técnica de rastreamento 2D desenvolvida por Kanade, Lucas e Tomasi [59], popularmente conhecida como KLT, a qual tende a rastrear melhor os atributos próximos a quinas na imagem.

Uma grande vantagem do KLT é que o mesmo tem baixo custo computacional, permitindo calcular um conjunto com grande número de atributos. Uma desvantagem, porém, é que essa técnica é bastante sensível à variação de iluminação

e às oclusões. Essas vulnerabilidades podem comprometer o sistema, uma vez que a face real observada a partir da seqüência de imagens não é estática, pois sua forma é modificada por causa das expressões faciais e da iluminação. Além disso, a variação da pose pode fazer com que algumas regiões da face obstruam outras regiões contendo atributos importantes. Por exemplo, quando a cabeça está em posição de perfil, a visão de um dos olhos fica obstruída. Por esse motivo, o rastreamento de diversos atributos da cabeça pode falhar ao longo de uma seqüência de vídeo.

Pensou-se então que uma forma de diminuir este problema poderia ser a substituição no modelo 3D dos pontos correspondentes a atributos mal rastreados, por novos pontos correspondentes a atributos existentes nas novas regiões da imagem da cabeça captada pela câmera. Para testar essa hipótese foram definidos dois novos critérios: o critério de remoção de pontos expúrios e o critério de adição de novos pontos ao modelo.

Inicialmente, aborda-se o critério para inserção de um novo ponto no modelo. Parte-se do princípio que, na inicialização do sistema, a cabeça esteja em posição frontal, ou seja, com a face aparecendo de forma predominante. Considera-se também que os pontos, correspondentes aos novos atributos captados, precisam ser adicionados ao algoritmo quando a cabeça estiver em poses diferentes, como por exemplo, em posição de perfil.

Caso não sejam adicionados pontos nas laterais do modelo e a cabeça esteja em posição de perfil, grande parte dos atributos da face poderão ser perdidos por causa da oclusão, ocasionando falha na estimativa da pose. Então, a cada novo quadro, o modelo 3D é projetado na imagem, delimitando sua área visível. Utilizando o algoritmo [3] [79], atributos (quinas) são detectados e utilizados para inicializar novos pontos. Para validar esses pontos com o modelo 3D, os mesmos são projetados na máscara Candide. Um ponto só é válido se o mesmo interseccionar pelo menos um polígono da máscara, garantindo que o mesmo esteja associado a um ponto no espaço. Caso o ponto seja válido, o mesmo é adicionado em $\boldsymbol{m}$ e o respectivo ponto de interseção com o polígono da máscara é adicionado em $\boldsymbol{M}$.

Com o critério de inserção de novos pontos definido, passa-se a discutir o critério referente à remoção de pontos. Os atributos que não são bem rastreados

[3] Esse método, proposto por Shi e Tomasi, é implementado pela biblioteca OpenCV através da função *cvGoodFeaturesToTrack*.

tendem sempre a deturpar a estimativa da pose. Entendendo que à medida que o tempo fosse passando, pontos mal rastreados tentederiam a se somarem ao novos *outliers* que surgem a cada novo quadro, dominando e acabando por prejudicar a estimativa da pose, decidiu-se verificar a hipóte-se de ser necessário removê-los.

Como critério para remoção de pontos inadequados é utilizada então a avaliação do RANSAC. Resumindo, para remover a influência dos atributos mal rastreados, a cada novo quadro, os *outliers* encontrados durante a estimativa da pose com RANSAC são removidos do modelo.

### 3.2.5 Refinamento dos parâmetros da transformação 3D

O cálculo da transformação 3D da cabeça possui um erro inerente, uma vez que contém uma aproximação de primeira ordem. Sendo assim, tendo como meta diminuir uma provável intensificação na propagação desse erro a cada novo quadro, é proposto um refinamento no cálculo da transformação.

Na seção ( 3.2.2) foi exposto o uso da representação da matriz de transformação através da exponencial matricial, cuja a aproximação de primeira ordem é dada através de $\boldsymbol{G} \approx \boldsymbol{I}+\boldsymbol{\xi}$. Observa-se que, apesar da representação *twist* ser desenvolvida para a matriz de rotação, será utilizado $\boldsymbol{G}$ conforme a Equação (3.10), pois os componentes de translação também são lineares.

Para encontrar os parâmetros de $\boldsymbol{G}$, resolvemos o sistema $(\boldsymbol{I}+\boldsymbol{\xi})\boldsymbol{M} = \boldsymbol{M'}$ na forma $\boldsymbol{A\alpha} = \boldsymbol{b}$, onde $\boldsymbol{\alpha}$ contém os coeficientes de $\boldsymbol{G}$ e, por conseqüência, de $\boldsymbol{\xi}$, identificando-se essa primeira solução como $\boldsymbol{\alpha}_0$ e a respectiva matriz de transformação de $\boldsymbol{\xi}_0$.

Na verdade, o sistema exato deveria ser $(e^{\boldsymbol{\xi}})\boldsymbol{M} = \boldsymbol{M'}$; porém, como já dito anteriormente, queremos manter o sistema linear e por isso truncamos a série, mantendo apenas os elementos de primeira ordem. Assim sendo, temos um erro associado à solução que é da ordem de $\boldsymbol{\xi}^2$. Visando diminuir esse erro, refinamos a estimativa da solução utilizando como partida a solução $\boldsymbol{\xi}_0$. Para tanto, foi adicionado um termo de retificação $\boldsymbol{\xi}_1$, de modo que $\boldsymbol{\xi} = \boldsymbol{\xi}_0 + \boldsymbol{\xi}_1$. Como $\boldsymbol{\xi}_0$ já foi obtido na solução anterior, o problema passa a ser encontrar $\boldsymbol{\xi}_1$.

No refinamento $\boldsymbol{\xi} = \boldsymbol{\xi}_0 + \boldsymbol{\xi}_1$, uma nova série é gerada

$$e^{\boldsymbol{\xi}} = \boldsymbol{I} + (\boldsymbol{\xi}_0 + \boldsymbol{\xi}_1) + \frac{1}{2!}(\boldsymbol{\xi}_0 + \boldsymbol{\xi}_1)^2 + ... = \boldsymbol{I} + \boldsymbol{\xi}_0 + \boldsymbol{\xi}_1 + \frac{\boldsymbol{\xi}_0\boldsymbol{\xi}_1}{2} + \frac{\boldsymbol{\xi}_1\boldsymbol{\xi}_0}{2} + \frac{(\boldsymbol{\xi}_0)^2}{2} + \frac{(\boldsymbol{\xi}_1)^2}{2} + ... \tag{3.15}$$

Montamos o sistema $\boldsymbol{CM} = \boldsymbol{M'}$, onde $\boldsymbol{C} = \boldsymbol{I} + \boldsymbol{\xi}_0 + \boldsymbol{\xi}_1 + \frac{\boldsymbol{\xi}_0\boldsymbol{\xi}_1}{2} + \frac{\boldsymbol{\xi}_1\boldsymbol{\xi}_0}{2} + \frac{(\boldsymbol{\xi}_0)^2}{2}$ é a série truncada no termo de segunda ordem (observe que o elemento $\frac{(\boldsymbol{\xi}_1)^2}{2}$ foi descartado para manter o sistema linear), e o reescrevemos na forma $\boldsymbol{A'\alpha'} = \boldsymbol{b}$, onde $\boldsymbol{\alpha'}$ contém os coeficientes de $\boldsymbol{\xi}_1$. Uma vez resolvido esse sistema, remontamos $\boldsymbol{\xi}_1$ com os coeficientes $\boldsymbol{\alpha'}$ e obtemos $\boldsymbol{\xi} \approx \boldsymbol{\xi}_0 + \boldsymbol{\xi}_1$.

### 3.2.6 Predição

Mesmo que os pontos que representam os atributos da cabeça no plano da imagem sejam bem rastreados pelo KLT, caso haja uma oclusão muito grande, o conjunto de *inliers* pode ser muito pequeno para calcular uma boa estimativa da pose. Nesses casos, a estimativa da pose tende a falhar, fazendo com que o sistema derive. Para evitar esse problema, um modelo preditivo é inserido.

Neste trabalho, optou-se pela predição usando um Filtro Exponencial Duplo [80] que, segundo os autores, apresenta desempenho similar ao filtro de Kalman, porém é mais simples de implementar e possui um custo computacional muito menor. Dada uma série temporal dos parâmetros $\boldsymbol{\alpha}_t$, a predição no tempo $t + \tau$ é obtida de forma recursiva através de

$$\boldsymbol{\alpha}_{t+\tau} = \left(2 + \frac{\beta\tau}{1-\beta}\right)\boldsymbol{\nu}_t - \left(1 + \frac{\beta\tau}{1-\beta}\right)\boldsymbol{\nu}_t^{[2]}, \tag{3.16}$$

onde $\boldsymbol{S\alpha}_t$ e $\boldsymbol{S\alpha}_t^{[2]}$ são variáveis auxiliares computadas através de

$$\boldsymbol{\nu}_t = \beta\boldsymbol{\alpha}_t + (1-\beta)\boldsymbol{\nu}_{t-1}, \tag{3.17}$$

$$\boldsymbol{\nu}_t^{[2]} = \beta\boldsymbol{\nu}_t + (1-\beta)\boldsymbol{\nu}_{t-1}^{[2]}. \tag{3.18}$$

Aqui, $\beta$ é o fator de decaimento exponencial, onde valores pequenos para $\beta$ produzem predições atenuadas. Na abordagem proposta, foi utilizado $\beta = 0.1$ para obter predição suavizada e $\tau = 1$, o qual atualiza a predição a cada novo quadro, para gerar os parâmetros preditos $\boldsymbol{\alpha}_p = \boldsymbol{\alpha}_{t+1}$. A cada novo quadro, se a mediana do

erro $E_i$ obtido durante o cálculo da transformação for pequeno, então os parâmetros $\boldsymbol{\alpha}$ da transformação podem ser utilizados para atualizar a predição. Por outro lado, se a mediana do erro $E_i$ for muito grande, os parâmetros da transformação são ignorados e substituídos pelos parâmetros obtidos utilizando a predição baseada no histórico.

Resumindo, o modelo proposto é caracterizado em duas etapas principais. A primeira é responsável pela detecção da face em pose frontal e a segunda é responsável pela estimativa da pose. Na primeira etapa, foi adicionada a delimitação do espaço de busca baseada na cor de pele a fim de reduzir o custo computacional. A utilização de uma regra simples para a segmentação da cor de pele juntamente com a avaliação da quantidade de pixels de pele por intermédio da imagem integral foram as primeiras contríbuições desta dissertação. Na segunda etapa, para verificar supostas melhorias na estimativa da pose, foram propostos o refinamento, a atualização de pontos e o sistema preditivo. Através do refinamento, outra contribuição desta dissertação, é possível reduzir o erro de aproximação de primeira ordem, obtido no cálculo da transformação da pose. O modelo apresentado nesta dissertação efetua o rastreamento de atributos no plano da imagem para estimar a pose da cabeça no espaço, semelhante às técnicas [52] [55]. Porém, neste modelo, há uma inovação, onde os pontos mal rastreados são atualizados a cada novo quadro, permitindo uma melhor representação do modelo 3D da cabeça. Testes foram realizados para avaliar o modelo e o conjunto de melhorias descritos. Os resultados destes testes são apresentados e analisados no próximo capítulo.

# Capítulo 4

# Resultados

Este capítulo descreve os resultados obtidos usando as técnicas descritas nesta dissertação e está organizado em duas seções. Na primeira, apresentam-se gráficos, indicando o custo computacional, e uma tabela, com a taxa de detecção e a proporção de falsos positivos. Esses dados se referem à detecção de faces, pelo emprego do método de Viola e Jones, o qual utiliza atributos geométricos para identificação de objetos, combinado com a informação de cor de pele, pertinentes à técnica desenvolvida no âmbito deste estudo. Na segunda seção, é avaliado o algoritmo proposto neste trabalho para rastreamento da pose, levando em consideração fatores como a atualização de atributos rastreados, o refinamento, oclusão e a predição, conforme modelo já detalhado no capítulo 3.

## 4.1 Detecção da Face

Conforme explicado no capítulo 3, para a detecção de face foi utilizada a técnica proposta por Viola e Jones. Contudo, considerando-se o alto custo computacional desta técnica, foi proposta uma redução do espaço de busca, com base apenas na seleção de áreas correpondentes à cor de pele.

Nesse processo, foi utilizada uma base de dados de imagens com grande variedade de fotografias, contendo faces humanas em diferentes ambientes. Essa base de dados, construída originalmente para treinar e testar algoritmos de reconhecimento facial, foi escolhida porque está disponível na internet e porque contém imagens coloridas, particularmente necessárias aos testes em questão, que aplicam filtro de cor de pele. Tais imagens estão disponíveis em

`http://vis-www.cs.umass.edu/lfw/`, em formato JPG com dimensões de 250 × 250 pixels. Foram utilizadas 300 imagens selecionadas aleatoriamente, com um total de 325 faces. Além dessas, foram utilizados também imagens obtidas a partir de seqüências de vídeo gravadas com uma *webcam*, para auxiliar na verificação da qualidade da segmentação baseada na cor de pele sob condições de iluminação adversas.

A regra de classificação de pele descrita na seção 3.1.1 foi aplicada ao conjunto de imagens de teste. Como modelo de referência, foi utilizada a técnica descrita por Nusirwan *et al.* [73] para efetuar comparações quantitativas. Tanto essas regras quanto a regra utilizada nesta dissertação tiveram bons resultados para classificar os pixels com cor de pele. Porém, constatou-se que [73], junto com sua alta precisão, tem grande sensibilidade a variações de iluminação. Em contrapartida, a técnica proposta por esta dissertação, a qual flexibiliza os limiares de [73], tem menor precisão, mas também menor sensibilidade à variação de luminosidade.

A Figura 4.1 apresenta algumas imagens escolhidas para ilustrar o resultado do filtro proposto nesta dissertação. Fica evidente o número maior de falsos positivos, quando utilizada a regra proposta neste trabalho, em relação à regra proposta em [73]. Fica evidente também que, em nossa proposta, dificilmente pixels referentes à pele não foram detectados, em contraponto à técnica utilizada como referência. Optou-se, então pelo relaxamento dos limiares, com o objetivo de privilegiar uma ampla classificação do tipo "cor de pele", mesmo que, eventualmente, sejam classificadas áreas de "não pele". Essa abordagem garante que grande parte dos pixels com cor de pele seja, de fato, identificada.

A etapa anterior alimentou os testes que se seguiram, os quais foram focados na detecção de faces, utilizando o detector treinado disponível na biblioteca OpenCV. De acordo com o descrito na seção 3.1.2, o detector foi treinado por Lienhart e possui vinte estágios com uma taxa de detecção de $0.999^{20}$ e uma taxa de falsos positivos de $0.5^{20}$. Esta etapa corresponde especificamente à detecção da face e considerou resultados de testes realizados sobre a mesma base de imagens utilizada para avaliar a segmentação da pele. Esses testes tiveram como objetivo avaliar o ganho obtido em relação ao custo computacional e ao número de falsos positivos, quando adicionada à redução do espaço de busca baseada na cor de pele.

Para medir o custo computacional, foram avaliados o tempo de processamento

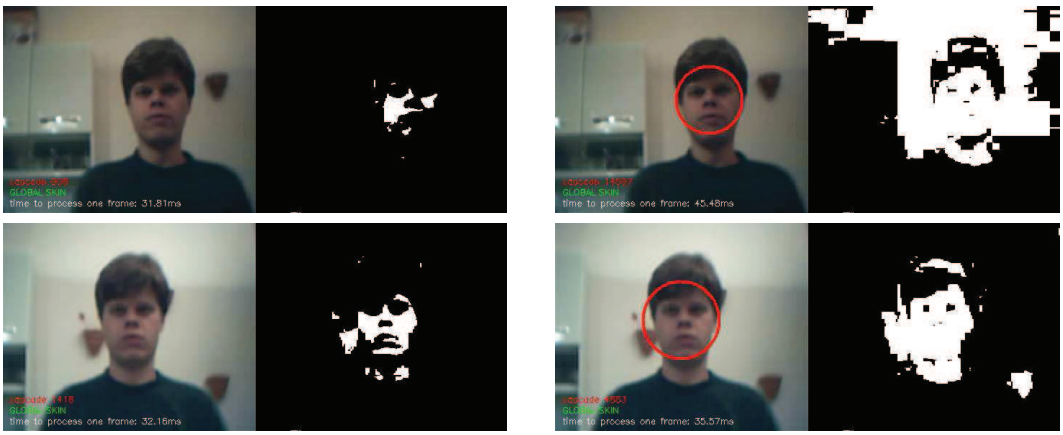


Figura 4.1 – Comparativo entre as regras de segmentação de cor de pele. a) regra proposta em [73]. b) regra proposta nessa dissertação.

e o número de *haar-like features* utilizadas a cada imagem computada. O gráfico da Figura 4.2 apresenta os resultados obtidos. Nesse gráfico é possível perceber que o número de *haar-like features* verificadas e o tempo de processamento são menores quando a redução do espaço de busca é empregado. Cabe lembrar que existe a possibilidade de, em alguns casos onde praticamente toda imagem passe no teste de cor de pele, o custo do detector com pré-processamento de cor ser maior do que quando apenas utilizado isoladamente; contudo, sendo tais casos esporádicos, afirma-se ser compensador utilizar o pré-processamento.

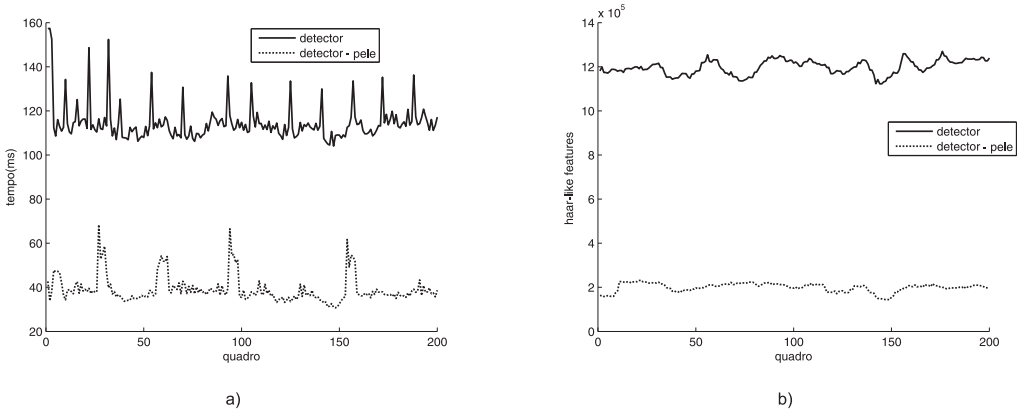


Figura 4.2 – Comparativo de custo computacional aplicando a redução do espaço de busca. a) tempo gasto para processar cada quadro; b) número de *haar-like features* utilizadas em cada quadro.

A Tabela 4.1 compara o número de falsos positivos pela utilização da segmentação do espaço de busca com cor de pele e o número de falsos positivos pelo emprego da técnica original de detecção de faces baseada apenas em características

geométricas. Com base nos dados desta tabela é possível verificar que, com a redução do espaço de busca, há uma diminuição dos falsos positivos. Isso é previsível, uma vez que muitos falsos positivos acontecem em regiões que não pertencem à cor de pele. A taxa de detecção também diminui, porém, em quantidade não significativa. Essa diminuição acontece porque o detector não consegue identificar as faces em imagens onde a iluminação modifica fortemente a tonalidade da cor de pele ou quando a face sofre oclusão parcial por algum objeto com cor diferente da pele. A Figura 4.3 ilustra alguns destes casos com exemplos.

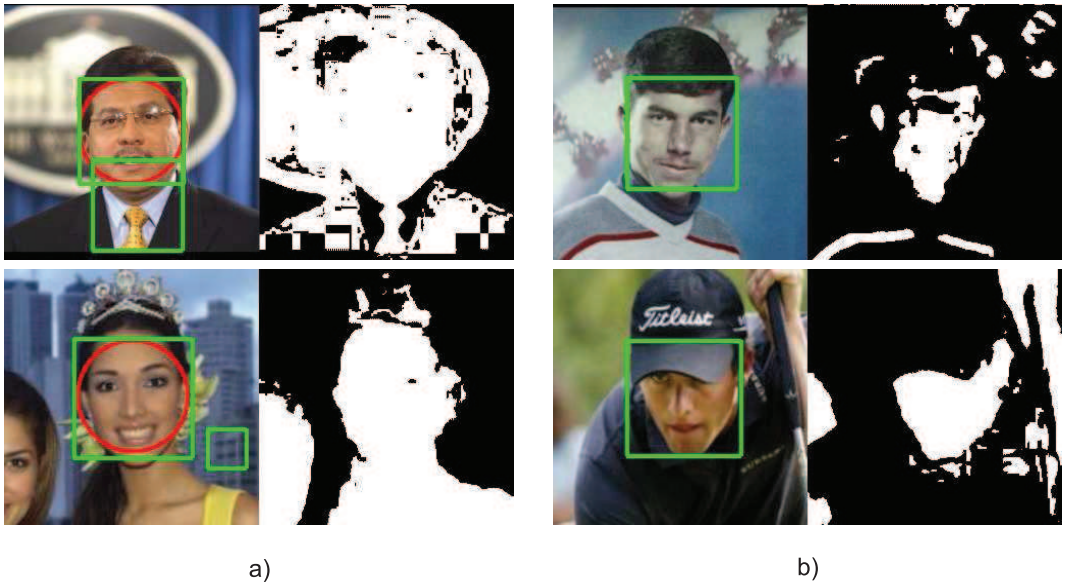


Figura 4.3 – Falsos positivos são removidos enquanto falsos negativos são adicionados com a redução do espaço de busca. O círculo identifica o detector com segmentação da pele e o quadrado identifica o detector original baseado apenas em informação geométrica.

Tabela 4.1 – Precisão do detector de face

| | Viola & Jones (somente) | Viola & Jones (com pré-processamento) |
|---|---|---|
| taxa de detecção | 97% | 96% |
| taxa de falsos positivos | 3% | 2% |

## 4.2 Rastreamento da Pose

Uma vez identificada a posição inicial da face na seqüência considerada, a segunda parte deste capítulo é dedicada aos testes e à análise dos resultados obtidos

com a técnica de rastreamento da pose. Para fazer a análise desses resultados, foram observados os critérios de precisão do ângulo e da posição da cabeça, com uma taxa média de processamento de 15 quadros por segundo. Os testes também foram realizados com o objetivo de analisar o ganho correspondente às etapas de refinamento e de atualização dos pontos. Durante os testes, foi utilizada uma base de dados com seqüências de imagens (resolução $320 \times 240$ pixels) de indivíduos variando a pose da cabeça. Essa base de dados foi desenvolvida na *Bostom University* e pode ser obtida através de `http://www.cs.bu.edu/fac/sclaroff/ivc/HeadTracking/`. Cada seqüência de vídeo contém um *ground-truth*[1] extraído por intermédio de um sensor magnético, possibilitando medir o erro que se evidencia com a aplicação da técnica proposta nesta dissertação.

Essa base de dados não contém exemplos com oclusões, o que ocasionou a necessidade de gravar algumas seqüências com uma *webcam* para simular tais situações. Com o objetivo de avaliar a sensibilidade da técnica em relação a esses fatores, foram simuladas cenas onde ocorrem oclusão parcial e total da cabeça, bem como mudanças bruscas na iluminação. A Figura 4.4 mostra algumas imagens extraídas das seqüências de vídeo utilizadas nos experimentos.

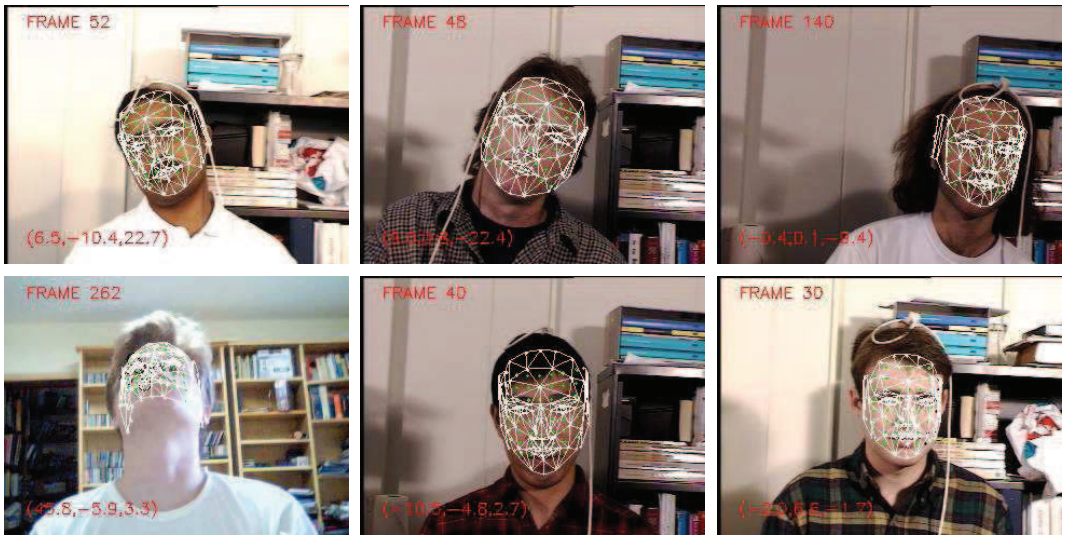


Figura 4.4 – Quadros extraídos de seqüências de vídeo processadas durante a realização dos testes com o algoritmo proposto.

[1]No contexto da estimativa da pose, *ground-truth* é um conjunto de informações contendo a posição e rotação da cabeça a cada quadro da seqüência de vídeo, extraídas através de um sensor confiável, representando um conjunto verdade. Esse conjunto verdade é utilizado para avaliar algoritmos de estimativa da pose por intermédio da comparação dos valores do mesmo com os valores obtidos pelo algoritmo avaliado.

Cada seqüência de vídeo da base de dados foi processada com o algoritmo proposto por este estudo. Para cada vídeo processado, foram armazenados a posição da cabeça no espaço, representados como $X$,$Y$,$Z$ e também os ângulos dos eixos *Yaw*, *Roll* e *Pitch*. Uma avaliação quantitativa foi feita, comparando os dados conseguidos com aqueles disponíveis na base de dados com *ground-truth*.

Utilizando a configuração do sistema conforme descrito no modelo, foram processados vídeos da base de dados, organizados em três focos distintos, quais sejam, a avaliação do refinamento, a avaliação da atualização de pontos, e a avaliação da sensibilidade do sistema a oclusões.

A partir deste ponto do texto, portanto, apresenta-se os resultados obtidos durantes os testes do sistema considerando esses três fatores. Na análise dos resultados obtidos durante os testes realizados, foram avaliados a variação da posição da cabeça em $X$,$Y$,$Z$ e a variação angular nos eixos *Yaw*, *Roll* e *Pitch*.

### 4.2.1 Avaliação do Refinamento

A meta da realização dos testes foi verificar se existe melhora de resultados ao empregar a técnica de refinamento. Inicialmente, durante testes sintéticos do modelo realizados em MATLAB, pode ser observada uma melhora da estimativa da pose quando aplicado o refinamento. Nesses testes, rotações com variação angular crescente entre 0 e 20 graus foram aplicadas aos eixos *Roll*, *Pitch* e *Yaw* do modelo 3D. Para cada uma das configurações angulares foi medido o erro absoluto obtido no cálculo da transformação utilizando três pontos pertencentes ao modelo 3D e escolhidos aleatoriamente, sendo cada configuração repetida 1000 vezes. A Figura 4.5 ilustra o resultado do experimento com uma dessas configurações, onde a rotação no eixo *Roll* ficou fixada em 3°, no eixo *Pitch* em 3° e no eixo *Yaw* variou de 0° a 20°. Nesse gráfico, a linha pontilhada representa o algoritmo sem refinamento e a linha contínua representa o algoritmo com refinamento. É possível observar que, à medida que a diferença angular cresce no eixo *Yaw*, maior é o erro na transformação; porém, este é atenuado no algoritmo que utiliza o refinamento.

Acreditando-se na redução da propagação do erro quando utilizado o refinamento, foram feitos testes para avaliar o algoritmo em situações reais. Para isso, foram utilizadas as seqüências de vídeo da base de dados da *Boston University*. A Figura 4.6 ilustra o resultado da aplicação do algoritmo desenvolvido com e

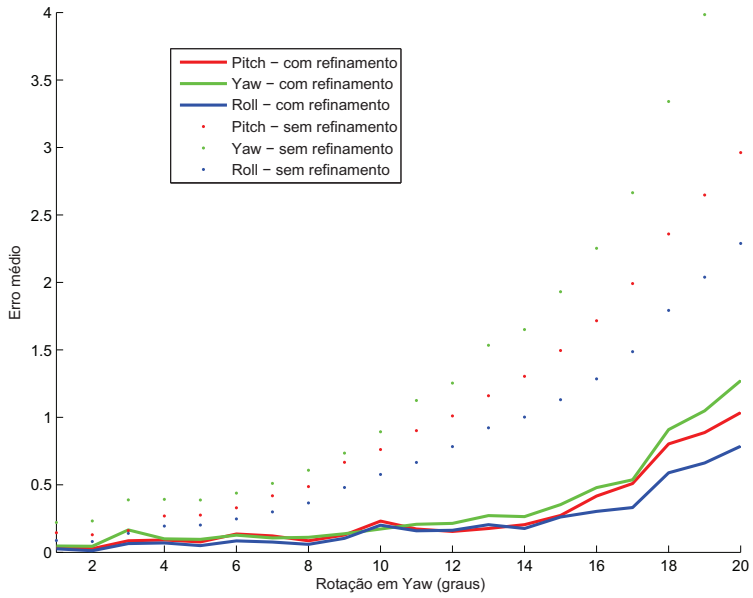


Figura 4.5 – Avaliação do erro no cálculo da transformação 3D. Linha contínua representa o algoritmo com refinamento. Linha pontilhada representa o algoritmo sem refinamento. A rotação no eixo *Roll* ficou fixada em 3°, no eixo *Pitch* ficou fixada em 3° e no eixo *Yaw* variou de 0° a 20°.

sem refinamento. Nessa figura, em forma de grade, cada célula contém os valores correspondentes ao rastreamento para cada um dos casos (com refinamento, sem refinamento e *ground-truth*) representados por cores distintas. A linha vermelha corresponde ao *ground-truth*; a verde, ao algoritmo proposto sem o refinamento; a azul, ao mesmo algoritmo, porém, utilizando o refinamento. Em cada segmento horizontal dessa grade, está representado a variação de um dos graus de liberdade analisados ($X$,$Y$,$Z$, *Yaw*, *Roll* e *Pitch*) e, por fim, cada coluna dessa grade representa o processamento de um vídeo distinto.

Na coluna da esquerda (vídeo *jim4*), a maior variação do movimento acontece no eixo *Pitch*; na coluna central (vídeo *ssm8*), a maior variação do movimento acontece no eixo *Yaw*; e na coluna da direita (vídeo *jam1*), a maior variação do movimento acontece no eixo *Roll*. Esses três vídeos representam bem a base de dados e, por intermédio deles, é possível observar que tanto o algoritmo sem refinamento quanto o algoritmo com o refinamento tiverem resultados semelhantes.

A Tabela 4.2 apresenta o resultado do erro absoluto médio obtido após o processamento de todos os vídeos da base dados. Essa, de acordo com o visualizado nos gráficos da Figura 4.6, mostra que não houve diferenças significativas entre o algoritmo com refinamento e o mesmo sem refinamento. Acredita-se que o ganho

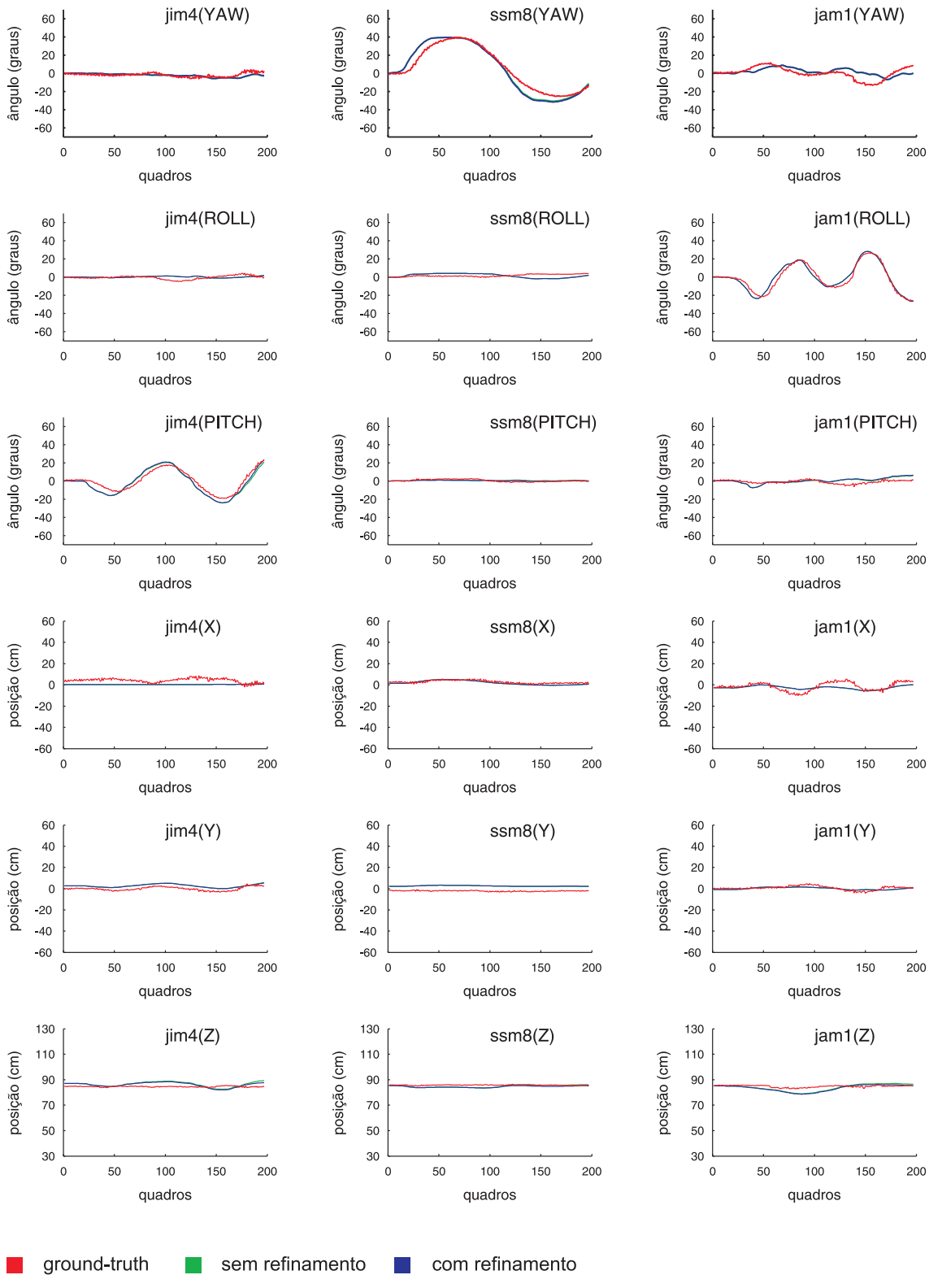


Figura 4.6 – Avaliação da técnica proposta considerando o refinamento.

obtido no refinamento é perdido devido ao erro no rastreamento dos pontos na imagem pelo KLT (nos testes sintéticos, a posição exata dos pontos rastreados foi utilizada, o que não ocorre quando algoritmos de rastreamento são utilizados.).

Tabela 4.2 – Avaliação do refinamento: comparativo do erro absoluto médio (em graus) da estimativa da pose utilizando a base de dados da *Boston University*.

| Algoritmo | *Pitch* | *Yaw* | *Roll* |
|---|---|---|---|
| sem refinamento | 3.33 | 4.64 | 2.43 |
| com refinamento | 3.39 | 4.63 | 2.41 |

Desta forma, o refinamento deve contribuir apenas quando utilizado em conjunto com um algoritmo de rastreamento mais robusto.

### 4.2.2 Avaliação da Atualização de Pontos

Finalizados os testes com o refinamento, passa-se a considerar como foco principal a avaliação da atualização dos pontos. Foram realizados testes para avaliar a qualidade da estimativa da pose em relação à atualização dos pontos rastreados na imagem, repetindo-se a seqüência de procedimentos e a lógica de estruturação anteriormente empregada.

Como dito na descrição do modelo, a cada novo quadro da imagem, pontos descartados pelo RANSAC são eliminados e novos pontos são adicionados. A Figura 4.8 apresenta os resultados obtidos efetuando novo teste em três seqüências distintas. Nessa nova figura, a linha vermelha continua correspondendo ao *ground-truth*; a azul, ao algoritmo com atualização dos pontos; a verde, ao algoritmo sem atualização de pontos.


Figura 4.7 – Seqüência (*jim7*) com variação angular no eixo *Yaw*. a) técnica sem atualização de pontos. b) técnica com atualização de pontos.

Na coluna da esquerda (*jam7*), a pose é estimada com boa precisão mesmo sem a atualização de pontos, onde o erro obtido nos eixos *Roll*, *Pitch* e *Yaw* é menor que três graus. O mesmo não ocorre na coluna do centro e da direita (vídeo *jim7*

e *jim9*). Na coluna do centro, a partir do quadro 80, o algoritmo sem atualização de pontos diverge no eixo *Yaw*. A Figura 4.7 ilustra, com imagens retiradas da seqüência de vídeo *jim7*, o erro no rastreamento.

Ao analisar o erro absoluto médio na avaliação do experimento com atualização de pontos, verificou-se que a diferença foi muito pequena entre o algoritmo com atualização de pontos e sem atualização de pontos, conforme mostra a Tabela 4.3. Porém, nos casos onde há grandes rotações no eixo *Yaw*, a adição de pontos na lateral da cabeça auxiliou o sistema na estimativa da pose.

Tabela 4.3 – Avaliação da atualização de pontos: comparativo do erro absoluto médio (em graus) da estimativa da pose utilizando a base de dados da *Boston University*.

| Algoritmo | *Pitch* | *Yaw* | *Roll* |
|---|---|---|---|
| sem atualização | 3.33 | 4.64 | 2.43 |
| com atualização | 3.84 | 4.66 | 2.30 |

Os casos ilustrados pelos gráficos das figuras 4.6 e 4.8 representam bem as classes de resultados observadas durante os testes com a base de dados da universidade de Boston. Nas figuras em questão foram escolhidos apenas alguns exemplos mais significativos. Na tabela 4.4 estão disponibilizados a média e o desvio padrão do erro obtido nos demais testes realizados para avaliar o algoritmo completo, com refinamento e atualização de pontos.

Os resultados obtidos utilizando todos os 45 vídeos da base de dados foi comparado com o resultado publicado por Ho An e Chung [1]. A tabela 4.5 apresenta os resultados da técnica proposta nesta dissertação, utilizando o modelo Candide modificado, com os resultados de [1] utilizando um plano, um cilindro e uma elipsóide como modelos para a cabeça. É possível ver que a técnica proposta nesta dissertação teve resultados inferiores apenas em relação ao modelo elipsóide de [1], e, mesmo assim, a diferença foi bem pequena.

### 4.2.3 Avaliação da sensibilidade à oclusão

Por fim, considerando que não existem exemplos no *ground-truth* com oclusão, para avaliar a sensibilidade do sistema em relação a oclusões parciais, foram realizados testes com as seqüências próprias, gravadas com uma webcam. Nessas seqüências, um objeto é anteposto entre a câmera e a cabeça de forma a obstruir

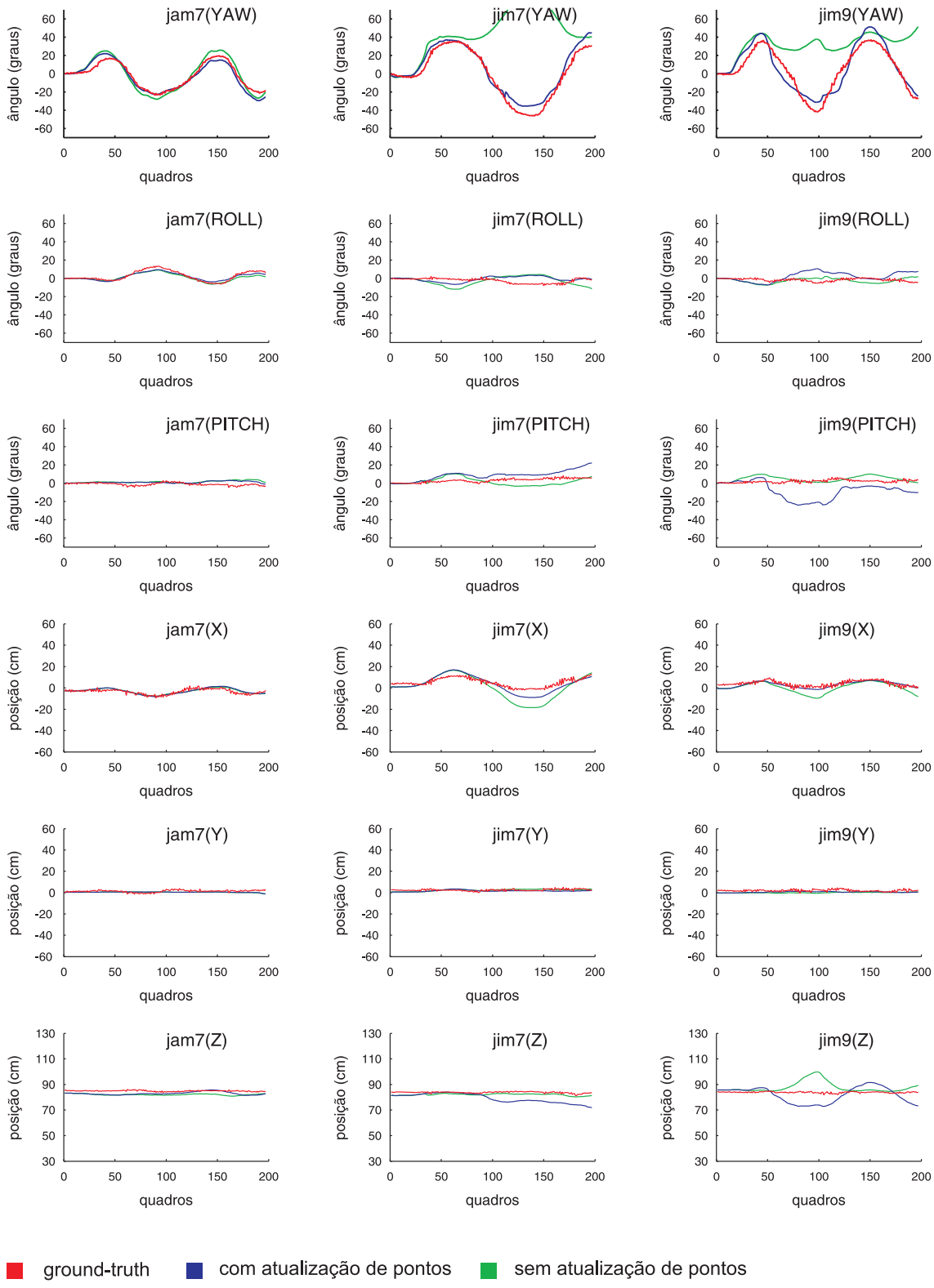


Figura 4.8 – Avaliação da técnica proposta considerando a atualização dos atributos 2D no plano da imagem.

a visão de parte da mesma, pela câmera. A avaliação dos resultados foi realizada de forma visual. A Figura 4.9 apresenta alguns quadros de uma das seqüências de vídeo utilizadas durante os testes. Essa figura foi organizada com alguns quadros

onde há oclusão e dividida em duas partes. Na parte (a), a estimativa da pose é deturpada pela oclusão. Analisando o quadro 353 desta figura, é possível notar que os *inliers* (pontos em verde), que deveriam estar sob a região da face, estão situados sobre a capa do livro. Nesse caso, os pontos sob a capa do livro, apesar de serem em menor quantidade, formam o maior grupo de pontos que efetuam uma transformação coerente entre si. Por causa disso, a pose é estimada baseada no movimento do livro, e não da cabeça. Na parte (b) é apresentado um caso onde o sistema consegue estimar a pose mesmo com grande oclusão da cabeça. No quadro 750, os pontos circulados em azul representam os *outliers* gerados pela oclusão feita pelo livro. Neste caso, os pontos sobre a face formam o maior conjunto que possui um movimento coerente entre si, e por isso, os *outliers* não influenciam a transformação. No quadro 780, por causa da oclusão muito grande, não foi possível detectar um número de *inliers* suficientes para estimar a transformação. Nesse caso, a predição foi utilizada para estimar a pose. A partir do quadro 785, os pontos sobre a face já foram atualizados e o rastreamento segue normalmente.

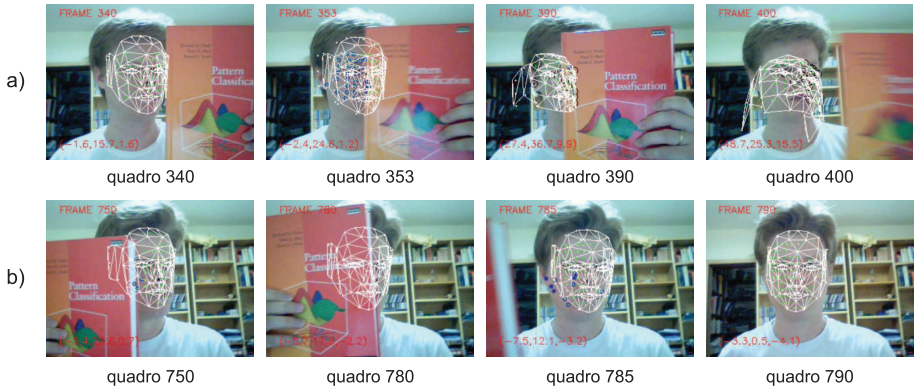


Figura 4.9 – Seqüência de vídeo contendo oclusão parcial da cabeça. a) os atributos rastreados sobre a região do livro formam o maior conjunto coerente com a transformação final obtida. b) os atributos rastreados sobre a região da cabeça formam o maior conjunto coerente com a transformação final obtida.

Na seção 3.2.6 foi descrito o modelo preditivo. Esse modelo auxilia o sistema nos casos onde a estimativa da pose não pode ser encontrada devido à oclusão de maioria de pontos ou devido ao erro de rastreamento em uma grande quantidade de atributos. Caso o número de *inliers* seja muito pequeno (menor que um limiar mínimo de aceitação), o sistema ignora a transformação encontrada e utiliza a predição para atualizar o modelo. Um exemplo dessa situação foi ilustrada na Figura 4.9(b).

Tabela 4.4 – Avaliação quantitativa da estimativa da pose.

| Vídeo | *Pitch* | | *Yaw* | | *Roll* | | *X* | | *Y* | | *Z* | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ |
| jam1 | 2.26 | 1.85 | 4.08 | 3.43 | 2.82 | 2.27 | 2.67 | 2.05 | 1.23 | 0.90 | 2.73 | 2.66 |
| jam2 | 1.55 | 1.06 | 4.24 | 3.00 | 1.28 | 1.00 | 2.34 | 1.59 | 1.52 | 1.11 | 1.39 | 1.48 |
| jam3 | 1.19 | 0.97 | 1.91 | 1.77 | 1.36 | 1.28 | 2.41 | 2.43 | 2.68 | 2.54 | 4.15 | 2.87 |
| jam4 | 2.01 | 1.44 | 2.46 | 1.60 | 1.37 | 0.84 | 2.80 | 1.88 | 2.99 | 2.75 | 4.20 | 3.46 |
| jam5 | 1.52 | 1.44 | 4.86 | 3.66 | 1.29 | 0.93 | 1.91 | 1.49 | 1.12 | 0.79 | 2.85 | 1.45 |
| jam6 | 12.09 | 9.38 | 3.13 | 2.61 | 3.13 | 2.21 | 3.23 | 2.31 | 2.26 | 1.31 | 4.00 | 3.06 |
| jam7 | 1.99 | 1.29 | 3.38 | 3.13 | 1.89 | 1.44 | 1.47 | 1.11 | 1.27 | 0.72 | 4.59 | 2.32 |
| jam8 | 3.36 | 4.75 | 3.56 | 3.54 | 2.94 | 2.00 | 3.16 | 2.34 | 2.37 | 1.85 | 3.54 | 2.80 |
| jam9 | 1.90 | 1.41 | 6.53 | 3.69 | 2.80 | 2.11 | 2.25 | 1.42 | 1.44 | 1.18 | 1.00 | 0.61 |
| jim1 | 1.21 | 0.93 | 4.27 | 3.59 | 0.73 | 0.67 | 3.40 | 1.78 | 2.88 | 0.91 | 3.03 | 2.16 |
| jim2 | 0.88 | 0.71 | 5.86 | 5.16 | 0.95 | 0.66 | 3.54 | 1.86 | 1.83 | 0.64 | 2.55 | 1.47 |
| jim3 | 2.96 | 2.01 | 0.87 | 0.67 | 0.98 | 0.79 | 4.39 | 1.23 | 2.17 | 0.86 | 1.60 | 1.12 |
| jim4 | 5.03 | 3.69 | 1.71 | 1.21 | 1.67 | 1.53 | 4.04 | 1.78 | 2.77 | 1.04 | 2.50 | 1.36 |
| jim5 | 2.66 | 1.83 | 2.56 | 1.87 | 1.51 | 0.92 | 3.28 | 2.09 | 1.91 | 0.98 | 2.54 | 1.13 |
| jim6 | 4.08 | 3.04 | 2.96 | 2.31 | 5.07 | 3.51 | 2.54 | 1.65 | 1.76 | 1.24 | 1.83 | 0.94 |
| jim7 | 2.46 | 2.01 | 12.83 | 10.45 | 4.37 | 2.94 | 4.25 | 2.73 | 1.05 | 0.70 | 8.52 | 4.54 |
| jim8 | 6.61 | 4.31 | 3.28 | 2.64 | 2.70 | 2.03 | 2.93 | 1.75 | 3.21 | 1.16 | 4.97 | 3.72 |
| jim9 | 1.48 | 1.22 | 11.57 | 13.08 | 1.33 | 1.03 | 2.86 | 2.15 | 1.78 | 1.06 | 5.23 | 4.47 |
| llm1 | 2.12 | 1.83 | 8.16 | 5.64 | 2.07 | 1.65 | 3.33 | 2.17 | 1.53 | 1.16 | 4.42 | 1.73 |
| llm2 | 6.46 | 5.52 | 4.73 | 5.24 | 1.77 | 2.05 | 2.50 | 1.45 | 1.89 | 1.98 | 4.83 | 2.84 |
| llm3 | 2.28 | 1.62 | 7.52 | 5.10 | 2.76 | 1.72 | 3.28 | 1.78 | 4.66 | 1.43 | 1.33 | 1.33 |
| llm4 | 9.75 | 8.97 | 10.14 | 9.44 | 2.65 | 1.72 | 2.89 | 1.97 | 4.52 | 2.95 | 4.12 | 4.13 |
| llm5 | 4.65 | 3.90 | 4.10 | 4.14 | 4.31 | 4.73 | 2.40 | 1.61 | 3.93 | 1.79 | 2.44 | 1.85 |
| llm6 | 8.15 | 6.25 | 1.97 | 1.71 | 3.21 | 1.68 | 2.18 | 1.71 | 3.67 | 1.40 | 4.40 | 2.64 |
| llm7 | 2.64 | 2.36 | 4.18 | 3.02 | 0.98 | 0.53 | 1.61 | 0.91 | 4.60 | 1.17 | 1.29 | 1.24 |
| llm8 | 2.62 | 1.58 | 7.95 | 6.01 | 1.76 | 1.57 | 2.11 | 1.45 | 4.50 | 1.29 | 2.91 | 2.29 |
| llm9 | 6.29 | 4.11 | 9.82 | 7.24 | 2.33 | 2.13 | 1.78 | 1.33 | 2.88 | 1.68 | 2.71 | 1.62 |
| ssm1 | 1.10 | 0.82 | 5.18 | 3.28 | 2.13 | 1.51 | 3.70 | 2.55 | 5.40 | 1.34 | 1.67 | 1.33 |
| ssm2 | 2.63 | 2.61 | 2.00 | 1.29 | 0.53 | 0.39 | 2.35 | 1.77 | 4.53 | 0.86 | 5.54 | 3.59 |
| ssm3 | 0.55 | 0.35 | 5.59 | 4.00 | 1.14 | 0.98 | 1.95 | 0.95 | 4.46 | 0.52 | 2.13 | 1.93 |
| ssm4 | 4.23 | 3.84 | 1.85 | 1.40 | 3.46 | 2.42 | 2.06 | 0.76 | 5.09 | 1.31 | 0.92 | 0.79 |
| ssm5 | 7.39 | 5.09 | 1.03 | 0.87 | 2.08 | 1.86 | 1.44 | 0.81 | 5.78 | 1.70 | 4.34 | 2.43 |
| ssm6 | 8.94 | 6.31 | 1.36 | 1.19 | 2.93 | 2.25 | 1.26 | 0.73 | 4.38 | 1.19 | 6.88 | 3.13 |
| ssm7 | 2.55 | 2.13 | 2.95 | 2.33 | 3.33 | 2.50 | 2.88 | 2.39 | 5.52 | 1.64 | 2.61 | 2.35 |
| ssm8 | 1.58 | 0.86 | 11.89 | 7.39 | 2.32 | 2.05 | 1.93 | 1.11 | 4.86 | 0.49 | 3.82 | 3.37 |
| ssm9 | 1.10 | 0.75 | 11.05 | 6.92 | 2.39 | 1.52 | 3.81 | 2.22 | 4.35 | 1.74 | 3.34 | 2.52 |
| vam1 | 1.54 | 1.05 | 2.12 | 1.17 | 1.35 | 1.02 | 3.00 | 1.87 | 3.22 | 1.64 | 1.22 | 0.78 |
| vam2 | 4.23 | 3.28 | 2.16 | 1.71 | 1.42 | 1.61 | 2.01 | 1.64 | 5.52 | 3.14 | 5.08 | 4.22 |
| vam3 | 6.50 | 4.47 | 2.19 | 1.81 | 3.32 | 2.89 | 1.82 | 1.46 | 5.50 | 2.23 | 4.77 | 3.13 |
| vam4 | 4.54 | 3.73 | 4.75 | 4.93 | 2.27 | 1.97 | 2.97 | 2.62 | 4.14 | 1.78 | 1.41 | 0.95 |
| vam5 | 7.53 | 8.23 | 3.52 | 3.15 | 4.00 | 3.18 | 3.41 | 3.02 | 6.91 | 3.07 | 5.00 | 3.33 |
| vam6 | 3.97 | 3.55 | 2.73 | 2.20 | 3.33 | 3.02 | 2.86 | 2.23 | 5.75 | 3.48 | 4.11 | 2.84 |
| vam7 | 4.03 | 3.76 | 5.00 | 4.49 | 3.94 | 3.10 | 2.87 | 2.58 | 4.84 | 2.43 | 4.54 | 4.06 |
| vam8 | 6.07 | 3.84 | 3.40 | 2.99 | 1.36 | 1.34 | 2.57 | 2.07 | 3.64 | 2.23 | 3.23 | 2.56 |
| vam9 | 4.57 | 3.83 | 2.37 | 2.20 | 2.52 | 2.38 | 2.29 | 1.67 | 4.26 | 2.29 | 3.46 | 2.17 |

Tabela 4.5 – Comparativo do erro médio (em graus) da estimativa da rotação utilizando a base de dados da *Boston University*.

| Eixo | Técnica proposta | Técnica proposta em [1] | | |
|---|---|---|---|---|
| | Candide adaptado | Elipsóide | Cilindro | Plano |
| *Roll* | 2.30 | 2.83 | 3.22 | 2.99 |
| *Yaw* | 4.66 | 2.95 | 5.33 | 18.08 |
| *Pitch* | 3.84 | 3.96 | 7.22 | 7.33 |

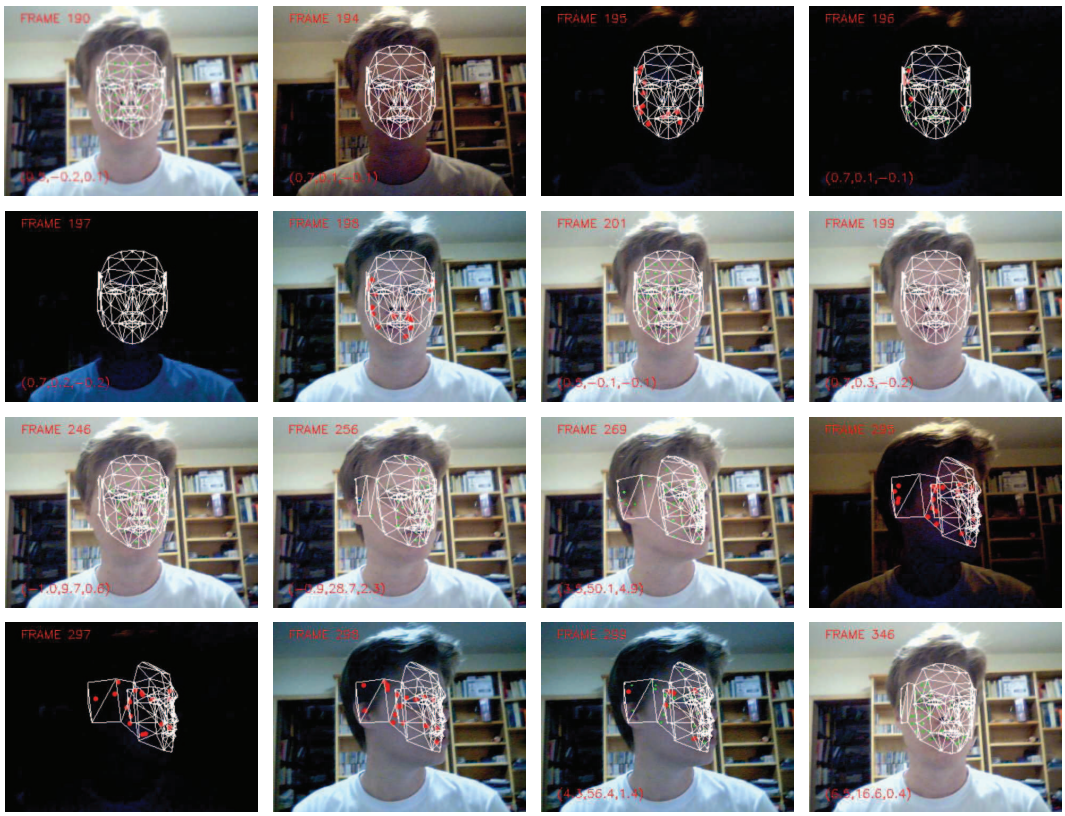


Figura 4.10 – Ao aplicar a predição, a cabeça continua sendo rastreada corretamente, mesmo com mudanças bruscas de iluminação.

A predição é útil também em casos onde a imagem sofre mudanças bruscas de iluminação. A figura 4.10 demonstra o uso da predição através de uma seqüência de vídeo que contém uma variação repentina dessa. Nessa figura, a cabeça está em pose frontal entre os quadros 190 e 201. Neste intervalo, a iluminação da cena é desligada e ligada novamente em seguida. Como existe uma mudança muito grande na intensidade da imagem, o rastreamento dos atributos é ruim e gera um pequeno número de *inliers*, e, por isso, o sistema utiliza a predição para estimar a pose. Outra mudança brusca na iluminação acontece entre os quadros 295 e 299, onde a cabeça está em posição de perfil. Novamente, a pose é estimada corretamente através da predição.

# Capítulo 5

# Considerações Finais

Este trabalho apresentou um conjunto de técnicas que constituem um sistema capaz de estimar a pose da cabeça em seqüências de vídeo captadas por uma câmera monocular colorida. O desenvolvimento desse sistema foi baseado no estudo prévio de trabalhos relacionados com detecção de faces e estimativa da pose.

Durante a revisão desses trabalhos, foi observado que, no que diz respeito à estimativa da pose, existem duas abordagens principais: estimativa discreta e estimativa contínua. A discreta é feita através de classificadores, e seu inconveniente é a impossibilidade de se obter uma estimativa fina da pose. Por esse motivo, optou-se por empregá-la apenas para detectar a face em pose frontal. A partir daí, desenvolveu-se uma técnica de estimativa contínua da pose. Assim sendo, o modelo proposto foi divido em duas etapas, onde a primeira é responsável por detectar a face em pose frontal e a segunda é responsável por estimar a pose de forma contínua.

Na primeira etapa, que utiliza a técnica de Viola e Jones [26] para detectar faces, foi inserido um algoritmo de redução do espaço de busca. Tal redução é feita por intermédio da segmentação de cor de pele, a qual, como conseqüência, proporcionou a redução do custo computacional e a diminuição dos falsos positivos. Ambos representando uma contribuição relevante desta dissertação.

O resultado da primeira etapa serviu para alimentar a segunda, responsável por estimar a pose da cabeça. Nessa etapa, a estimativa da pose é obtida por intermédio de atributos 2D no plano da imagem. Partindo do deslocamento desses, foi possível estimar a transformação 3D sofrida pela cabeça que melhor descreve as translações 2D locais de cada um dos atributos.

Durante a realização dos testes descritos no capítulo 4, pode ser observado que o refinamento fornece estimativas melhores da pose quando o casamento dos pontos é perfeito (teste sintético). Entretanto, o refinamento não reproduz seu desempenho em vídeos reais, nos quais o rastreamento usando KLT apresenta erros. Acredita-se que o refinamento deve gerar resultados melhores se um rastreador mais eficiente for utilizado no lugar do KLT. A atualização de pontos contribuiu para melhorar a estimativa da pose em alguns casos particulares, nos quais havia grandes variações no eixo *Yaw*. Ao avaliar o conjunto total de 45 vídeos da base de dados *Boston University*, contudo, verificou-se que a diferença entre o algoritmo com atualização de pontos e o mesmo sem a atualização de pontos é muito pequena.

Para essa base de dados, o sistema proposto obteve resultados muito próximos aos publicados em [1]. Além disso, o erro médio obtido é bem pequeno, aproximadamente 2.3° para o eixo *Roll*, 4.6° para o eixo *Yaw* e 3.8° para o eixo *Pitch*.

Conforme verificado em testes com seqüências obtidas a partir de uma *webcam*, o sistema tende a falhar em casos onde há grande variação da pose, como por exemplo, em rotações acima de 40 graus. Para amenizar esse problema, causado pela acumulação do erro ao passar do tempo, o sistema é reiniciado sempre que a face estiver em pose frontal, utilizando-se para isso o detector de faces. Com a adição da predição, foi possível tornar o sistema menos sensível a variações bruscas de iluminação e a grandes oclusões. Contudo, em casos onde essas situações se prolonguem por muito tempo, é necessário registrar que o sistema preditivo não é suficiente.

Em relação ao custo computacional, foi possível processar uma média de 15 quadros por segundo, utilizando-se seqüências de vídeo $320 \times 240$ num computador com processador Pentium4 de 3GHz. Tanto o rastreamento dos atributos, quanto as iterações do algoritmo RANSAC são executados atualmente de forma sequencial, mas, ambos podem ser paralelizados. Acredita-se que, com o algoritmo paralelizado, a taxa de processamento de quadros por segundo seja muito maior, especialmente nas novas arquiteturas que estão sendo comercializadas atualmente, as quais possuem múltiplos núcleos de processamento.

## 5.1 Trabalhos Futuros

Uma das dificuldades encontradas na abordagem do rastreamento da pose ocorre quando a posição inicial do modelo 3D da cabeça não está alinhada com a imagem da mesma. Durante os testes com o detector de faces, foi observado que ele detecta faces ligeiramente rotacionadas, prejudicando a estimativa inicial da pose assumida frontal. Uma nova revisão bibliográfica já está sendo realizada com o objetivo de adicionar uma etapa intermediária entre as etapas de detecção e de estimativa da pose. Nessa etapa intermediária, pretende-se fazer um realinhamento da máscara 3D Candide com a face detectada na imagem. A análise das técnicas observadas nessa nova revisão bibliográfica apontam para o uso de *Active Appearence Models - AAMs* [81] [82], porém, é preciso verficar criteriosamente a viabilidade do uso dos AAMs devido ao alto custo computacional associado.

Além da inicialização, outro problema é justamente a reinicialização. A idéia da reinicialização é fazer o alinhamento do modelo 3D da cabeça com quadros chaves, a fim de evitar que o erro seja propagado. Esses quadros chaves [63] [64] representam um conjunto das poses mais freqüentes, obtidas durante o rastreamento ou através de um treinamento prévio *offline*.

Esses dois assuntos, quais sejam, melhorar a inicialização e fazer o realinhamento por quadros chaves não foram abordados nesta dissertação e serão tratados em trabalhos futuros.

# Bibliografia

[1] AN, K. H.; CHUNG, M. J. 3d head tracking and pose-robust 2d texture map-based face recognition using a simple ellipsoid model. In: *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, September 22-26, 2008, Acropolis Convention Center, Nice, France*. [S.l.: s.n.], 2008. p. 307–312.

[2] NATURALPOINT.COM. *Natural Point Inc.* Acesso em 22 de março de 2009. Disponível em: <http://www.naturalpoint.com>.

[3] LEE, J.-E. et al. Head detection and tracking for the car occupant's pose recognition. In: *IEA/AIE*. [S.l.: s.n.], 2006. p. 540–547.

[4] TRIVEDI, E. M.-C. A. D. M. M. Head pose estimation for driver assistance systems: A robust algorithm and experimental evaluation. In: *Intelligent Transportation Systems Conference*. [S.l.: s.n.], 2007. p. 709–714.

[5] WANG, C.; GRIEBEL, S.; BR, M. Robust automatic video-conferencing with multiple cameras and microphones. In: *ICME '00: International Conference on Multimedia and Expo*. New York, USA: [s.n.], 2000. p. 1585–1588.

[6] WANG, C.; BRANDSTEIN, M. Head pose estimation for video-conferencing with multiple cameras and microphones. In: *ICMI '00: Proceedings of the Third International Conference on Advances in Multimodal Interfaces*. London, UK: Springer-Verlag, 2000. p. 111–118. ISBN 3-540-41180-1.

[7] CHAN, H.; BLEDSOE, W. *A Man-Machine Facial Recognition System: Some Preliminary Results*. [S.l.], 1965.

[8] KANADE, T. *Picture Processing System by Computer Complex and Recognition of Human Faces*. Tese (Doutorado) — Kyoto University, Japan, November 1973.

[9] YANG, M.-H.; KRIEGMAN, D. J.; AHUJA, N. Detecting faces in images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE Computer Society, Los Alamitos, CA, USA, v. 24, n. 1, p. 34–58, 2002. ISSN 0162-8828.

[10] MURPHY-CHUTORIAN; TRIVEDI, M. M. Head pose estimation in computer vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE Computer Society, Los Alamitos, CA, USA, v. 30, n. 6, p. 1–20, 2008. ISSN 0162-8828.

[11] JAIN, A. K.; LI, S. Z. *Handbook of Face Recognition*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005. ISBN 038740595X.

[12] PLATANIOTIS, K. N.; VENETSANOPOULOS, A. N. *Color image processing and applications*. New York, NY, USA: Springer-Verlag New York, Inc., 2000. ISBN 3-540-66953-1.

[13] CHAI, S. L. P. . A. B. . D. Skin segmentation using color pixel classification: Analysis and comparison. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, IEEE Computer Society, Washington, DC, USA, v. 27, n. 1, p. 148–154, 2005. ISSN 0162-8828. Member-Son Lam Phung and Sr. Member-Abdesselam Bouzerdoum and Sr. Member-Douglas Chai.

[14] ZARIT, B. D.; SUPER, B. J.; QUEK, F. K. H. Comparison of five color models in skin pixel classification. In: *Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, 1999. Proceedings. International Workshop on*. Washington, DC, USA: IEEE Computer Society, 1999. p. 58–63. ISBN 0-7695-0378-0.

[15] WONG, K.-W.; LAM, K.-M.; SIU, W.-C. An efficient color compensation scheme for skin color segmentation. In: *Circuits and Systems, 2003. ISCAS '03. Proceedings of the 2003 International Symposium on*. [S.l.: s.n.], 2003. p. 676–679.

[16] RAJA, Y.; MCKENNA, S.; GONG, S. Tracking and segmenting people in varying lighting conditions using colour. In: *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*. Los Alamitos, CA, USA: IEEE Computer Society, 1998. v. 00, p. 228–233. ISBN 0-8186-8344-9.

[17] SORIANO, M. et al. Adaptive skin color modeling using the skin locus for selecting training pixels. *Pattern Recognition*, v. 36, n. 3, p. 681–690, 2003.

[18] STÖRRING, M. et al. Tracking regions of human skin through illumination changes. *Pattern Recognition Letters*, v. 24, n. 11, p. 1715–1723, 2003.

[19] NGUYEN, T.; HUANG, T. Segmentation, grouping and feature detection for face image analysis. In: *Computer Vision, 1995. Proceedings., International Symposium on*. Los Alamitos, CA, USA: IEEE Computer Society, 1995. v. 00, p. 593–598. ISBN 0-8186-7190-4.

[20] WANG, J.-G.; SUNG, E. Frontal-view face detection and facial features extraction using color and morphological operations. *Pattern Recognition Letters*, Elsevier Science Inc., New York, NY, USA, v. 20, n. 10, p. 1053–1068, 1999. ISSN 0167-8655.

[21] LIN, C.; FAN, K.-C. Pose classification of human faces by weighting mask function approach. *Pattern Recognition Letters*, v. 24, n. 12, p. 1857–1869, 2003.

[22] LIN, C.; FAN, K.-C. A color-triangle-based approach to the detection of human face. In: *BMVC '00: Proceedings of the First IEEE International Workshop on Biologically Motivated Computer Vision*. London, UK: Springer-Verlag, 2000. p. 359–368. ISBN 3-540-67560-4.

[23] PAPAGEORGIOU, C. P.; OREN, M.; POGGIO, T. A general framework for object detection. In: *Computer Vision, 1998. Sixth International Conference on*. Los Alamitos, CA, USA: IEEE Computer Society, 1998. v. 00, p. 555–562. ISBN 81-7319-221-9.

[24] DUDA, R. O.; HART, P. E.; STORK, D. G. *Pattern Classification (2nd Edition)*. [S.l.]: Wiley-Interscience, 2000. ISBN 0471056693.

[25] HEISELE, B.; SERRE, T.; POGGIO, T. A component-based framework for face detection and identification. *International Journal of Computer Vision*, Kluwer Academic Publishers, Hingham, MA, USA, v. 74, n. 2, p. 167–181, 2007. ISSN 0920-5691.

[26] VIOLA, P.; JONES, M. Rapid object detection using a boosted cascade of simple features. In: *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. [S.l.: s.n.], 2001. v. 1, p. 511–518. ISSN 1063-6919.

[27] VIOLA, P.; JONES, M. J. Robust real-time face detection. *International Journal of Computer Vision*, Kluwer Academic Publishers, Hingham, MA, USA, v. 57, n. 2, p. 137–154, 2004. ISSN 0920-5691.

[28] HOU, X.; LIU, C.-L.; TAN, T. Learning boosted asymmetric classifiers for object detection. In: *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*. [S.l.: s.n.], 17–22 June 2006. v. 1, p. 330–338. ISSN 1063-6919.

[29] LIENHART, R.; MAYDT, J. An extended set of haar-like features for rapid object detection. In: *Image Processing. 2002. Proceedings. 2002 International Conference on*. [S.l.: s.n.], 2002. v. 1, p. 900–903. ISSN 1522-4880.

[30] MESSOM, C.; BARCZAK, A. Fast and efficient rotated haar-like features using rotated integral images. In: *Proceedings of the 2006 Australian Conference on Robotics and Automation*. Auckland, New Zealand: [s.n.], 2006. p. 1–6.

[31] ICHIKAWA, K.; MITA, T.; HORI, O. Component-based robust face detection using adaboost and decision tree. In: *Automatic Face and Gesture Recognition, 2006. 7th International Conference on*. [S.l.: s.n.], 10–12 April 2006. p. 413–420.

[32] GARCIA, C.; DELAKIS, M. Convolutional face finder: A neural architecture for fast and robust face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE Computer Society, Los Alamitos, CA, USA, v. 26, n. 11, p. 1408–1423, 2004. ISSN 0162-8828.

[33] FERAUND, R. et al. A fast and accurate face detector based on neural networks. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, v. 23, n. 1, p. 42–53, Jan 2001. ISSN 0162-8828.

[34] SUNG, K.-K.; POGGIO, T. Example-based learning for view-based human face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE Computer Society, Los Alamitos, CA, USA, v. 20, n. 1, p. 39–51, 1998. ISSN 0162-8828.

[35] TURK, M. A.; PENTLAND, A. P. Face recognition using eigenfaces. In: *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR '91.* [S.l.: s.n.], 1991. p. 586–591.

[36] HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. H. *The Elements of Statistical Learning*. [S.l.]: Springer, 2001. Hardcover. ISBN 0387952845.

[37] OSUNA, E.; FREUND, R.; GIROSIT, F. Training support vector machines: an application to face detection. In: *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*. [S.l.: s.n.], 17–19 Jun 1997. p. 130–136.

[38] TERRILLON, J.-C. et al. Invariant face detection in color images using orthogonal fourier-mellin moments and support vector machines. In: *ICAPR '01: Proceedings of the Second International Conference on Advances in Pattern Recognition*. London, UK: Springer-Verlag, 2001. p. 83–92. ISBN 3-540-41767-2.

[39] BASSIOU, N. et al. Frontal face detection using support vector machines and back-propagation neural networks. In: *Image Processing, 2001. Proceedings. 2001 International Conference on*. [S.l.: s.n.], 2001. v. 1, p. 1026–1029 vol.1.

[40] WONG, K.-W.; LAM, K.-M. A reliable approach for human face detection using genetic algorithm. In: *Circuits and Systems, 1999. Proceedings of the 1999 IEEE International Symposium on*. [S.l.: s.n.], Jul 1999. v. 4, p. 499–502 vol.4.

[41] WU, B. et al. Fast rotation invariant multi-view face detection based on real adaboost. In: *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*. [S.l.: s.n.], 17–19 May 2004. p. 79–84.

[42] WANG, Y. et al. Real-time multi-view face detection and pose estimation in video stream. In: *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*. [S.l.: s.n.], 0–0 0. v. 4, p. 354–357. ISSN 1051-4651.

[43] RAE, R.; RITTER, H. Recognition of human head orientation based on artificial neural networks. *Neural Networks, IEEE Transactions on*, v. 9, n. 2, p. 257–265, Mar 1998. ISSN 1045-9227.

[44] LI, Y. et al. Multi-view face detection using support vector machines and eigenspace modeling. In: *in Proceedings Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies*. [S.l.]: Science, 2000. p. 241–244.

[45] MCKENNA, S. J.; GONG, S.; COLLINS, J. J. Face Tracking and Pose Representation. In: FISHER, R. B.; TRUCCO, E. (Ed.). *British Machine Vision Conference*. Edinburgh: BMVA, 1996. v. 2, p. 755–764.

[46] HARRIS, C.; STEPHENS, M. A combined corner and edge detection. In: *Proceedings of The Fourth Alvey Vision Conference*. [S.l.: s.n.], 1988. p. 147–151.

[47] KRUGER, N.; POTZSCH, M.; MALSBURG, C. von der. Determination of face position and pose with a learned representation based on labeled graphs. *Image and Vision Computing*, v. 15, n. 8, p. 665–673, August 1997.

[48] LANITIS, A.; TAYLOR, C. J.; COOTES, T. F. Automatic interpretation and coding of face images using flexible models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 19, n. 7, p. 743–756, 1997.

[49] LEPETIT, V.; FUA, P. Monocular model-based 3D tracking of rigid objects: A survey. *Foundations and Trends in Computer Graphics and Vision*, v. 1, n. 1, p. 1–89.

[50] CASCIA, M. L.; SCLAROFF, S. Fast, reliable head tracking under varying illumination: An approach based on robust registration of texture-mapped 3d models. *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 22, p. 322–336, 2000.

[51] XIAO, J.; KANADE, T.; COHN, J. F. Robust full-motion recovery of head by dynamic templates and re-registration techniques. In: *Multimedia and Expo, 2006 IEEE International Conference on*. Los Alamitos, CA, USA: IEEE Computer Society, 2002. v. 0, p. 225–228. ISBN 0-7695-1602-5.

[52] WEISSENFELD, A. et al. Robust rigid head motion estimation based on differential evolution. In: *IEEE International Conference on Multimedia & Expo 2006*. [S.l.: s.n.], 2006. v. 0, n. 0, p. 225–228.

[53] AHLBERG, J. *CANDIDE-3 - An Parameterised Face*. Acesso em 22 de março de 2009. Disponível em: <http://www.bk.isy.liu.se/candide>.

[54] STORN, R.; PRICE, K. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *J. of Global Optimization*, Kluwer Academic Publishers, Hingham, MA, USA, v. 11, n. 4, p. 341–359, 1997. ISSN 0925-5001.

[55] TORDOFF, B. et al. Head pose estimation for wearable robot control. In: *British Machine Vision Conference*. [S.l.: s.n.], 2002. p. 2–5.

[56] TORDOFF, B. J. Guided-mlesac: Faster image transform estimation by using matching priors. *IEEE Trans. Pattern Anal. Mach. Intell.*, IEEE Computer Society, Washington, DC, USA, v. 27, n. 10, p. 1523–1535, 2005. ISSN 0162-8828. Member-David W. Murray.

[57] CHOI, C.; BAEK, S.-M.; LEE, S. Real-time 3d object pose estimation and tracking for natural landmark based visual servo. In: *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on.* [S.l.: s.n.], 2008. p. 3983–3989.

[58] LOWE, D. G. Object recognition from local scale-invariant features. In: *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on.* [S.l.: s.n.], 1999. v. 2, p. 1150–1157.

[59] LUCAS, B. D.; KANADE, T. An iterative image registration technique with an application to stereo vision. In: *Proceedings of the 1981 DARPA Image Understanding Workshop.* [S.l.: s.n.], 1981. p. 674–679.

[60] DEMENTHON, D. F.; DAVIS, L. S. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, v. 15, p. 123–141, 1995.

[61] J., H.; J., H.; M., P. A real-time facial feature based head tracker. In: *Proc. Advanced Concepts for Intelligent Vision Systems (ACIVS 2004), Brussels, Belgium.* [S.l.: s.n.], 2004. p. 267–272.

[62] VALENTE, S.; DUGELAY, J. A visual analysis/synthesis feedback loop for accurate face tracking. *Signal Processing: Image Communication*, v. 16, n. 6, p. 585–608, February 2001.

[63] RAHIMI, A.; MORENCY, L.-P.; DARRELL, T. Reducing drift in parametric motion tracking. In: *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on.* [S.l.: s.n.], 2001. v. 1, p. 315–322.

[64] VACCHETTI, L.; LEPETIT, V.; FUA, P. Stable real-time 3d tracking using online and offline information. *Pattern Analysis and Machine Intelligence*, IEEE Computer Society, Washington, DC, USA, v. 26, n. 10, p. 1385–1391, October 2004. ISSN 0162-8828.

[65] LEPETIT, V.; PILET, J.; FUA, P. Point matching as a classification problem for fast and robust object pose estimation. In: *Computer Vision and Pattern*

*Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*. [S.l.: s.n.], 2004. v. 2, p. 244–250. ISSN 1063-6919.

[66] RAVELA, S. et al. Adaptive tracking and model registration across distinct aspects. In: *International Conference on Intelligent Robots and Systems*. [S.l.: s.n.], 1995. p. 174–180.

[67] LEPETIT, V.; FUA, P. Keypoint recognition using randomized trees. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, v. 28, n. 9, p. 1465–1479, Sept. 2006. ISSN 0162-8828.

[68] AMIT, Y.; GEMAN, D. Shape quantization and recognition with randomized trees. *Neural Computation*, v. 9, p. 1545–1588, 1997.

[69] SEEMANN, E.; NICKEL, K.; STIEFELHAGEN, R. Head pose estimation using stereo vision for human-robot interaction. In: *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*. [S.l.: s.n.], 2004. p. 626–631.

[70] ZHANG, H. et al. Robust pose estimation for 3d face modeling from stereo sequences. In: *Image Processing. 2002. Proceedings. 2002 International Conference on*. [S.l.: s.n.], 2002. v. 3, p. 333–336. ISSN 1522-4880.

[71] NIESE, R.; AL-HAMADI, A.; MICHAELIS, B. A stereo and color-based method for face pose estimation and facial feature extraction. In: *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*. [S.l.: s.n.], 2006. v. 1, p. 299–302. ISSN 1051-4651.

[72] INTEL. *Open Source Computer Vision Library*. Acesso em 22 de março de 2009. Disponível em: <http://sourceforge.net/projects/opencvlibrary>.

[73] NUSIRWAN; WEI, K. C.; SEE, J. Rgb-h-cbcr skin colour model for human face detection. In: *Proceedings of The MMU International Symposium on Information and Communications Technologies*. [S.l.: s.n.], 2006.

[74] LIENHART, R.; KURANOV, E.; PISAREVSKY, V. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In: *In DAGM 25th Pattern Recognition Symposium*. [S.l.: s.n.], 2003. p. 297–304.

[75] FORSYTH, D.; PONCE, J. *Computer Vision: A Modern Approach*. Upper Saddle River, NJ, USA: Prentice Hall, 2003. ISBN 0130851981.

[76] HAGER, G. D.; BELHUMEUR, P. N. Efficient region tracking with parametric models of geometry and illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE Computer Society, Los Alamitos, CA, USA, v. 20, n. 10, p. 1025–1039, 1998. ISSN 0162-8828.

[77] LIANG, G.; ZHA, H.; LIU, H. Affine correspondence based head pose estimation for a sequence of images by using a 3d model. In: *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*. [S.l.: s.n.], 2004. p. 632–637.

[78] FISCHLER, M. A.; BOLLES, R. C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, ACM, New York, NY, USA, v. 24, n. 6, p. 381–395, 1981. ISSN 0001-0782.

[79] SHI, J.; TOMASI, C. Good features to track. In: *1994 IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 1994. p. 593 – 600.

[80] LAVIOLA, J. J. Double exponential smoothing: an alternative to kalman filter-based predictive tracking. In: *Proceedings of the Workshop on Virtual Environments*. New York, NY, USA: ACM Press, 2003. p. 199–206.

[81] GU, L.; KANADE, T. A generative shape regularization model for robust face alignment. In: *ECCV08*. [S.l.: s.n.], 2008. p. 413–426.

[82] GROSS IAIN MATTHEWS, S. B. R. Generic vs person specific active appearance models. *Image and Vision Computing*, Elsevier, v. 23, n. 1, p. 1080–1093, November 2005.