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

Leandro Lorenzett Dihl

Rastreamento de Objetos Usando Descritores Estatísticos

São Leopoldo
2009

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

Rastreamento de Objetos Usando Descritores Estatísticos

por

LEANDRO DIHL

Dissertação submetida a avaliação como requisito parcial para a obtenção do grau de Mestre em Computação Aplicada

Orientador: Prof. Dr. Claudio Rosito Jung

São Leopoldo, Janeiro de 2009

Agradecimentos

Quero agradecer inicialmente à Deus, que através do conhecimento possibilita a todos nós, transformar este mundo em lugar cada vez melhor.

Agradeço a minha esposa Adriana, que com palavras de apoio, carinho e paciência está sempre ao meu lado. Aos meus filhos queridos que são os tesouros da minha vida. Aos meus pais, principalmente minha mãe pelo incentivo ao estudo desde a minha infância. E todos os familiares e grandes amigos que sempre me incentivaram.

Quero fazer um agradecimento especial ao meu orientador e professor Claudio Rosito Jung, exemplo de profissional e caráter, pela confiança, amizade e por ter mostrado o caminho para transpor este grande desafio.

A todos os amigos do laboratório e da Unisinos que de alguma forma me ajudaram no transcorrer desta jornada.

E agradeço a HP Brazil pelo apoio financeiro dado a este trabalho.

Resumo

O baixo custo dos sistemas de aquisição de imagens e o aumento no poder computacional das máquinas disponíveis têm causado uma demanda crescente pela análise automatizada de vídeo, em diversas aplicações, como segurança, interfaces homem-computador, análise de desempenho esportivo, etc. O rastreamento de objetos através de câmeras de vídeo é parte desta análise, e tem-se mostrado um problema desafiador na área de visão computacional.

Este trabalho apresenta uma nova abordagem para o rastreamento de objetos baseada em fragmentos. Inicialmente, a região selecionada para o rastreamento é dividida em sub-regiões retangulares (fragmentos), e cada fragmento é rastreado independentemente. Além disso, o histórico de movimentação do objeto é utilizado para estimar sua posição no quadro seguinte. O deslocamento global do objeto é então obtido combinando os deslocamentos de cada fragmento e o deslocamento previsto, de modo a priorizar fragmentos com deslocamento coerente. Um esquema de atualização é aplicado no modelo para tratar os problemas de mudanças de iluminação e forma do objeto.

Os resultados experimentais obtidos indicaram que o esquema mostrou-se robusto para oclusões parciais, e também durante as oclusões totais por breve período de tempo. Uma análise comparativa com outras técnicas de rastreamento de objetos indica que o algoritmo proposto apresenta uma boa relação custo/benefício entre qualidade do rastreamento e tempo de execução.

Palavras-chave: Visão Computacional, Rastreamento de Objetos, Múltiplos Fragmentos , Matrizes de Covariância , Distância de Bhattacharrya.

TITLE: "OBJECT TRACKING USING STATISTICAL DESCRIPTORS"

Abstract

The low cost of image acquisition systems and increase the computational power of available machines have caused a growing demand for automated video analysis in several applications, such as surveillance, human-computer interfaces, analysis of sports performance, etc. Object tracking through the video sequence is part of this analysis, and it has been a challenging problem in the computer vision area.

This work presents a new approach for object tracking based on fragments. Initially, the region selected for tracking is divided into rectangular subregions (patches, or fragments), and each patch is tracked independently. Moreover, the motion history of the object is used to estimate its position in the subsequent frames. The overall displacement of the object is then obtained combining the displacements of each patch and the predicted displacement vector in order to priorize fragments presenting consistent displacement. An update scheme is also applied to the model, to deal with illumination and appearance changes.

The experimental results indicated that the proposed approach proved to be robust with respect to partial occlusions, and also to total occlusions for brief periods of time. A comparative analysis with other tracking algorithms also indicated that the proposed algorithm provides a good compromise between accuracy and running time.

Keywords: Computer Vision, Object Tracking, Multiple Fragments, Covariance Matrices, Bhattacharyya Distance.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

CPD *Coherent Patch Displacement*

EMD *Earth Mover's Distance*

KL *Kullback-Leibler*

KLT *Kanade-Lucas-Tomasi*

NCC *Normalized Cross-Correlation*

RGB *Red Green Blue*

SIFT *Scale Invariant Feature Transform*

TSV *Temporal Spatio-Velocity*

WVMF *Weighted Vector Median Filters*

Sumário

Capítulo 1

Introdução

Com a disponibilidade de câmeras de alta qualidade a custos cada vez menores, o aumento do poder de processamento dos computadores e a diminuição dos custos de armazenamento de dados, tem crescido o interesse em algoritmos para análise automatizada de vídeos. Esse interesse insere-se tanto na área de pesquisa acadêmica quanto no desenvolvimento de produtos comerciais baseados em visão de máquina. Yilmaz e colaboradores [1] citam algumas aplicações com análise automatizada de vídeos, que têm despertado o interesse da comunidade de pesquisa:

- reconhecimento baseado em movimentos, como a identificação humana baseada no caminhar ou a detecção de objetos de forma automática;
- vigilância automatizada, detectando atividades suspeitas em uma cena ou eventos não-desejados;
- interação homem-computador, com o reconhecimento de gestos e/ou o rastreamento dos olhos;
- catalogação de vídeos para o armazenamento e recuperação de vídeos em banco de dados multimídia;
- monitoramento de tráfego de veículos para a coleta, em tempo real, de estatísticas de trânsito;
- uso em realidade aumentada;
- controle na navegação de veículos automatizados, para o planejamento de itinerários e desvios e contornos de obstáculos.

Segundo trabalho de Yilmaz [1], há três passos chave na análise automatizada de um vídeo: A **detecção** de objetos a serem acompanhados[1], o **rastreamento** de tais objetos de quadro para quadro, e a **análise** do objeto rastreado para o conhecimento da sua ação. Em particular, o rastreamento de objetos é uma tarefa importante e ainda muito desafiadora no campo da visão computacional. Atualmente os estudos de rastreamento estão focados no desenvolvimento de algoritmos robustos, que sejam rápidos o bastante para uso em tempo real.

De uma forma sucinta, rastreamento pode ser definido como o problema de identificar o mesmo objeto de interesse (ou parte dele) em uma sucessão de quadros, sendo empregado normalmente em um contexto de aplicações de alto nível que requerem a localização do objeto em todos os quadros de uma seqüência de vídeo. A evolução da posição do objeto ao longo do tempo (ou seja, sua trajetória) é normalmente utilizada em outras tarefas de alto nível, como a detecção e o reconhecimento de eventos.

1.1 O Problema

O rastreamento de objetos é um problema relevante na área de visão computacional, e encontra aplicações em uma variedade de situações. Entre elas, podemos citar a vigilância visual (identificando e rastreando pessoas em atividades suspeitas) e as interfaces baseadas em visão, entre várias outras. Um rastreador deve ser suficientemente genérico, robusto e, normalmente, ter um desempenho compatível com aplicações de tempo real. Todos esses requisitos geram diversos desafios na área de visão computacional, sem uma solução definitiva até agora. Um sistema com capacidade para rastrear um determinado objeto ou uma pessoa em uma seqüência de imagens pode apresentar alguns dos seguintes desafios:

- movimentos bruscos e complexos dos objetos rastreados;
- perda de informação causada pela projeção do ambiente em terceira dimensão para a imagem em duas dimensões;
- ruídos nas imagens;

[1] Os objetos de interesse são comumente denominados alvos.

- mudança na aparência de objetos, fundos e cenas;
- a estrutura de objetos não-rígidos, sua articulação natural ou sua forma complexa;
- oclusões parciais ou totais de objetos com objetos, ou de objetos com a cena;
- mudanças de iluminação ou sombras;
- movimentos da câmara;
- necessidade de processamento em tempo real.

Em geral, técnicas de rastreamento mais robustas aos desafios citados acima tendem a ser mais custosas do ponto de vista computacional. Assim, um desafio de pesquisa adicional é achar um bom comprometimento entre a robustez alcançada e o tempo de execução.

1.2 Objetivos

1.2.1 Objetivo Principal

O principal objetivo deste trabalho é desenvolver um novo algoritmo para o rastreamento de objetos baseado em fragmentos, considerando algumas das dificuldades referenciadas na seção anterior. Em particular, um foco maior será dado ao problema de oclusão (parcial e total).

1.2.2 Objetivos Específicos

A fim de atingir o objetivo principal deste trabalho, alguns objetivos específicos foram definidos:

- desenvolver representações de objetos baseadas em múltiplos fragmentos;
- desenvolver técnicas para rastrear cada fragmento de modo individual;
- estudar algoritmos para estimar a posição futura do alvo com base no histórico de movimentação;

- combinar as informações de deslocamento dos diversos fragmentos com o previsto;
- comparar a técnica proposta com outros algoritmos de rastreamento de objetos.

1.3 Metodologia

A metodologia de desenvolvimento deste trabalho consistiu em uma revisão bibliográfica aprofundada, seguida do desenvolvimento e a implementação do algoritmo proposto. A etapa seguinte foi a validação dos resultados, realizada de forma qualitativa e quantitativa.

E etapa de revisão bibliográfica indicou que o uso de múltiplos fragmentos apresentava uma boa solução para o problema das oclusões parciais. Ela também indicou que oclusões totais não são bem tratadas pelas técnicas existentes que usam múltiplos fragmentos, motivando o desenvolvimento de uma abordagem de predição de movimento para tratar tais oclusões.

O algoritmo desenvolvido foi implementado em C++, usando a biblioteca OpenCV [7] para as operações de captura e visualização das seqüências de vídeo. A validação da técnica proposta foi realizada de forma qualitativa, através da inspeção visual dos resultados, e de forma quantitativa, comparando o erro cometido pela técnica proposta e por outras existentes na bibliografia.

1.4 Estrutura deste trabalho

Este trabalho está estruturado da seguinte forma. O capítulo 2 apresenta uma revisão bibliográfica do estado da arte dos rastreadores de objetos atuais. O modelo adotado para a resolução do problema é desenvolvido e detalhado no capítulo 3. No capítulo 4, é apresentado um comparativo da abordagem desenvolvida com alguns modelos do estado da arte vistos no capítulo 2. Também, neste capítulo, é descrita uma abordagem do modelo proposto com a utilização de detectores de faces. No capítulo 5 é apresentada uma discussão sobre o modelo proposto. Finalmente, as conclusões e os trabalhos futuros são apresentados no capítulo 6.

Capítulo 2

Revisão Bibliográfica

Existem diferentes maneiras de caracterizar algoritmos de acompanhamento de objetos, analisando aspectos distintos dos algoritmos. Neste trabalho, será adotada uma categorização de acordo com [1], que classifica algoritmos de rastreamento de objetos em técnicas baseadas em características pontuais, em regiões e em contorno. Uma visão esquemática dessa taxionomia é apresentada na Figura 2.1.

Para cada sub-classe da categorização ilustrada na Figura 2.1, há uma grande variedade de abordagens diferentes. Este capítulo foca nas técnicas consideradas mais relevantes ou relacionadas com a abordagem adotada para o desenvolvimento do modelo proposto. Revisões bibliográficas mais abrangentes podem ser encontradas em artigos de *survey*, como [1].

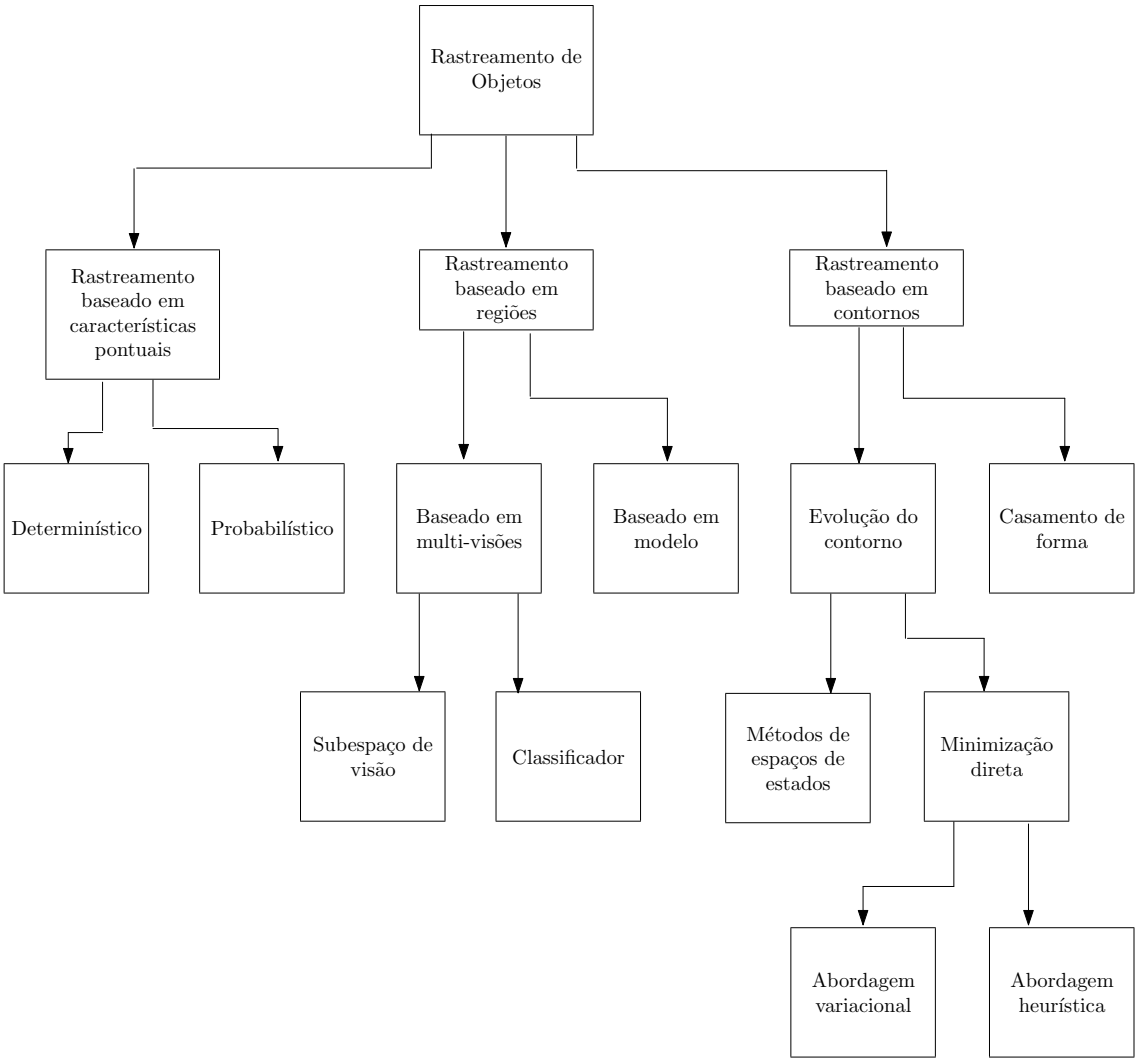


Figura 2.1 – Taxionomia das abordagens de rastreamento adaptando de Yilmaz [1]

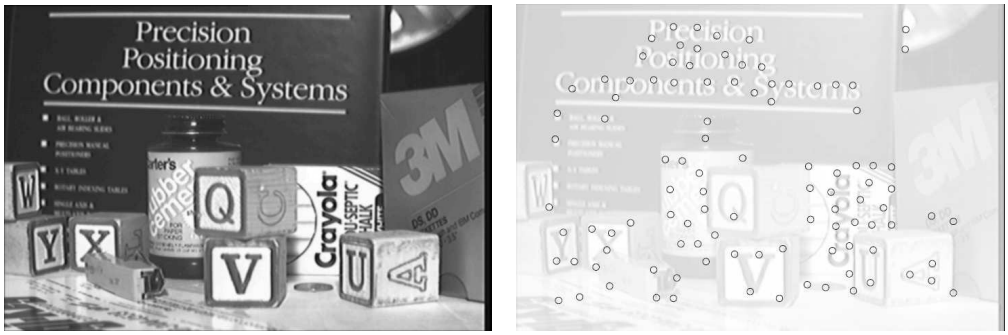
2.1 Rastreamento baseado em características pontuais

De acordo com Yilmaz e colaboradores [1], o rastreamento pode ser formulado como a similaridade de objetos representados por características pontuais através dos quadros de uma seqüência de imagens. A definição de métricas de similaridade entre pontos é um problema difícil durante o rastreamento, quando ocorrem oclusões ou entradas e saídas de objetos da cena.

No trabalho de Shi e Tomasi [2] é descrito o rastreador KLT *(Kanade-Lucas-Tomasi)*, que propõe um sistema de seleção de características pontuais para o rastreamento de objetos. Inicialmente, uma quantidade parametrizável de características é detectada automaticamente, com base em uma medida de "rastreabilidade". Durante o rastreamento, é realizado um monitoramento da qualidade das características da imagem, de modo que novas características podem ser incluídas caso outras percam a qualidade. O rastreamento de cada característica pontual é feita através de uma medida de similaridade entre uma pequena região em torno do ponto de análise, tanto no quadro atual quanto no quadro seguinte. Essa métrica pode levar em consideração transformações puramente translacionais, ou também transformações mais complexas (como as afins[1]). Conforme o tipo de transformação selecionada, seus parâmetros são calculados através de um método de minimização iterativo, similar ao de *Newton-Raphson*. Cada característica no KLT pode ser rastreada de forma bastante rápida, mas o deslocamento entre quadros adjacentes não pode ser muito grande. A Figura 2.2 apresenta o algoritmo KLT selecionando características de acordo com os critérios de textura da imagem, normalmente bordas e cantos de objetos na imagem.

O trabalho de Lowe [3] apresenta um método para extração de características distintivas em imagens. Tais características são invariantes com relação à escala e rotação da imagem, e mostraram-se adequadas para prover um casamento robusto entre pontos na presença de distorções afins, pequenas mudanças de perspectiva 3D, além de ruído e mudanças na iluminação. Essa abordagem, denominada SIFT (*Scale Invariant Feature Transform*), pode ser utilizada no rastreamento de objetos, através da caracterização do objeto de interesse com base em suas características invariantes, e da busca dessas mesmas características no quadro

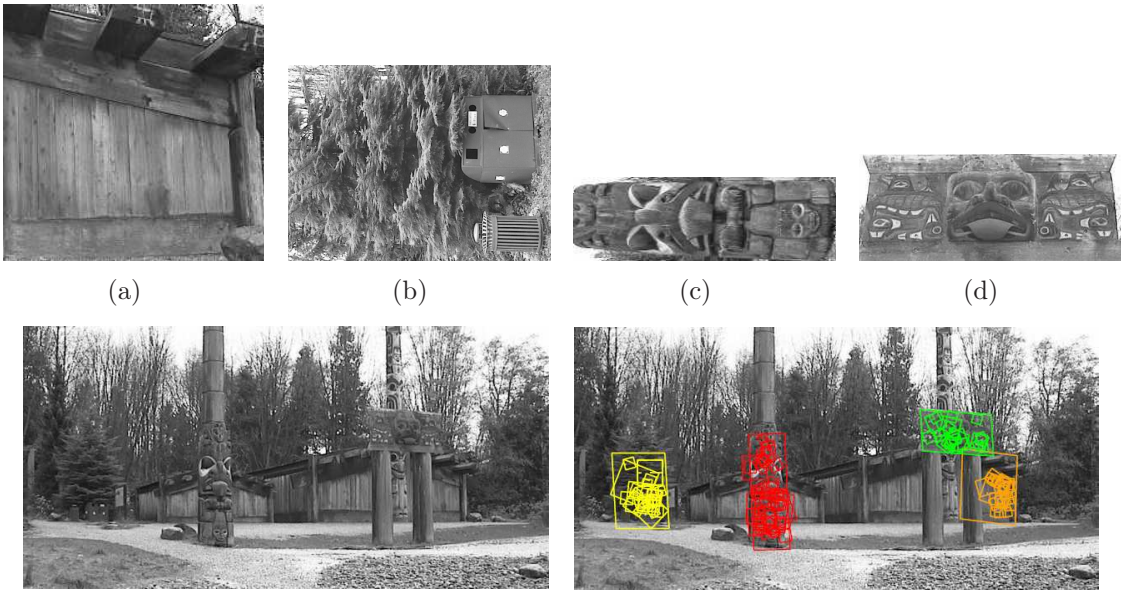
[1]Uma transformação afim pode ser caracterizada por $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$, e é amplamente utilizada em Computação Gráfica e Visão Computacional para representar transformações rígidas de objetos.



(a) (b)

Figura 2.2 – Rastreador KLT: (a) Imagem inicial. (b) Pontos selecionados seguindo o critério de texturas. Trabalho de Shi e Tomasi [2]

seguinte da seqüência de vídeo. O rastreamento por SIFT permite o casamento de objetos com deslocamentos significativos, mas o custo computacional no cálculo das características é relativamente alto para execução em tempo real. A Figura 2.3 apresenta alguns resultados do algoritmo SIFT, onde as Figuras 2.3(a), 2.3(b), 2.3(c) e a 2.3(d) são as imagens de treinamento, a Figura 2.3(e) é a imagem de teste e a Figura 2.3(f), as regiões reconhecidas.



(a) (b) (c) (d)

(e) (f)

Figura 2.3 – Algoritmo SIFT [3]. As Figuras (a), (b), (c) e (d) são as imagens de treinamento, a Figura (e) é a imagem para o teste. A Figura (f) mostra as regiões reconhecidas, com os pontos chaves representados como pequenos retângulos.

2.2 Rastreamento baseado em regiões

O rastreamento baseado em regiões é realizado normalmente pelo cálculo do movimento de um objeto, o qual é representado por uma primitiva (ou várias) da região do objeto, de um quadro para o outro. Esses algoritmos diferem entre si em termos do critério para seleção da região representativa do objeto, do número de regiões, e principalmente da maneira de caracterizar a região e a métrica para identificar a similaridade entre elas.

Uma abordagem para o rastreamento de objetos realizado através de regiões, no qual se enquadra a solução deste trabalho, é apresentada por Comaniciu et al. [4], que desenvolveram uma técnica para o rastreamento de objetos não-rígidos. Essa abordagem adota o uso de um núcleo (*kernel*), que codifica a configuração espacial da representação de uma área alvo elíptica, de modo que pixels longe do centro do alvo tenham uma contribuição menor na formação do histograma. Sendo $\{\boldsymbol{x}_i^\star\}_{i=1\ldots n}$ a localização do pixel normalizado na região do alvo (a origem é o vetor nulo $\mathbf{0}$), o histograma modulado $\hat{\mathbf{q}} = \{\hat{q}_u\}_{u=1\ldots m}$ é dado por

$$\hat{q}_u = C \sum_{i=1}^{n} k\left(\|\boldsymbol{x}_i^\star\|^2\right) \delta\left[b\left(\boldsymbol{x}_i^\star\right) - u\right], \tag{2.1}$$

onde k é um núcleo cujo perfil é dado por uma função decrescente, $b : A \rightarrow \{1, ..., m\}$ é uma função que associa cada ponto $\boldsymbol{x}_i^\star$ em A a um índice $b\left(\boldsymbol{x}_i^\star\right)$ no espaço de feições quantizado (ou seja, b é a função que gera o histograma original), e δ é a função delta de Kronecker discreta. A constante C, dada por

$$C = \frac{1}{\sum_{i=1}^{n} k\left(\|\boldsymbol{x}_i^\star\|^2\right)}, \tag{2.2}$$

é utilizada para normalizar o histograma de acordo com os pesos dos *kernels*. A motivação para o uso do núcleo é minimizar o efeito de pixels na periferia do alvo comprometidos por oclusões parciais.

A similaridade entre o modelo do alvo e os possíveis candidatos do quadro seguinte é feito através da medida de uma métrica, derivada do coeficiente de *Bhattacharyya* [4], que mede a similaridade entre histogramas. Um aspecto interessante dessa técnica é que o processo de busca é realizado iterativamente,

sem a necessidade de busca exaustiva, o que a torna eficiente do ponto de vista computacional. O resultado apresenta um rastreamento que enfrenta bem os problemas comuns de algoritmos de rastreamento de objetos, como movimentos complexos de câmeras, oclusões parciais do objeto (ao menos longe do centro do alvo), cenas confusas e variações de escala e aparência do objeto selecionado. Entretanto, deve haver uma certa sobreposição do alvo entre quadros adjacentes, para que o processo de busca iterativo convirja para a região correta, apesar de esforços recentes para melhorar a questão da convergência [8, 9]. A Figura 2.4 ilustra alguns resultados do trabalho de Comaniciu et al. [4].

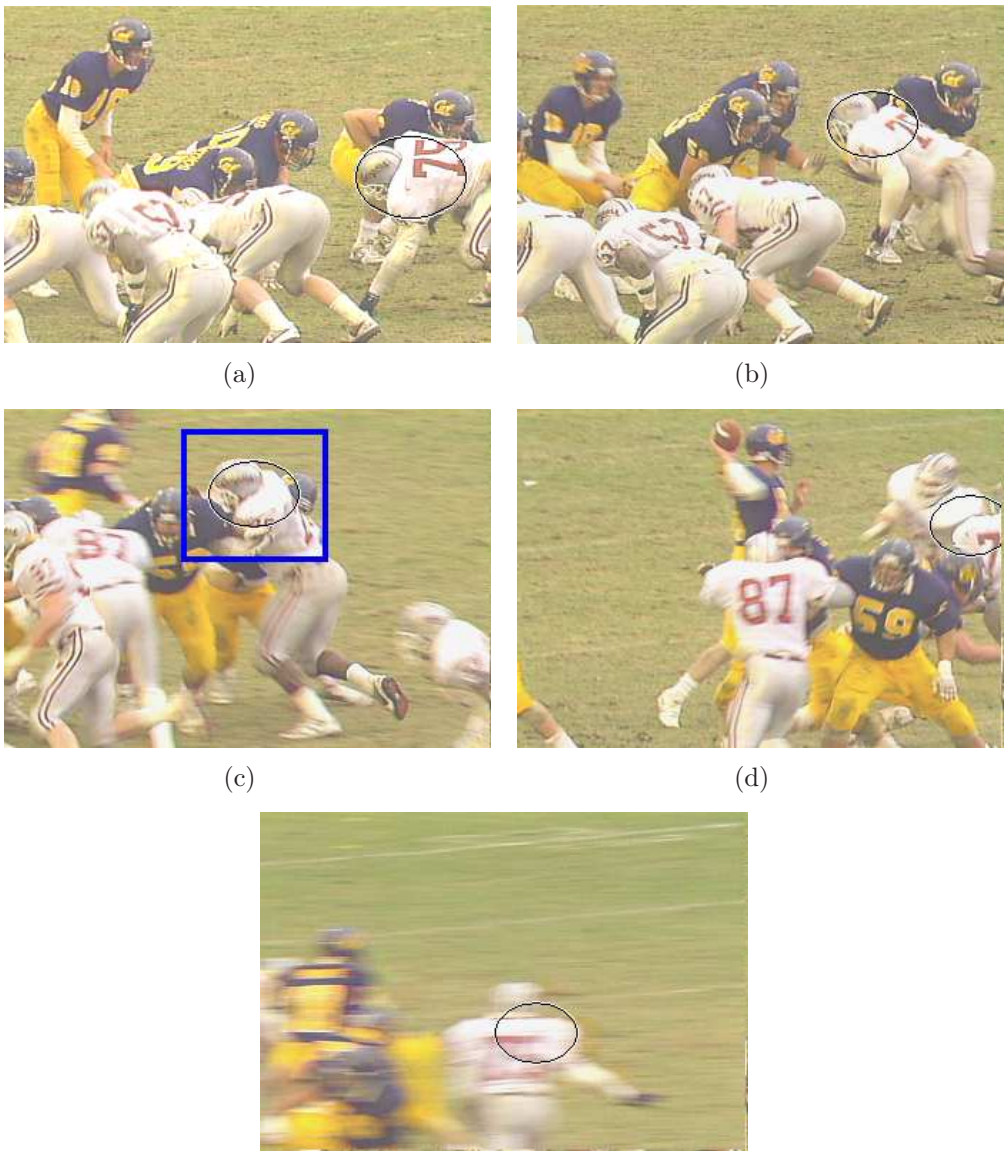


Figura 2.4 – Exemplo de rastreamento usando o trabalho de Comaniciu et al. [4].

No trabalho [10] foi explorado o conceito de imagem integral para a computação rápida de histogramas em qualquer sub-região retangular da imagem, permitindo o desenvolvimento de técnicas baseadas no casamento de histogramas com busca exaustiva para o rastreamento de objetos. O conceito de imagem integral foi empregada também por Adam et al. [5], que propuseram um algoritmo de rastreamento baseado em fragmentos com comparações de histogramas usando a EMD (*Earth Mover's Distance*). O uso de fragmentos permite robustez no rastreamento durante oclusões parciais, desde que os pedaços não oclusos possam fornecer o resultado de rastreamento correto.

Os fragmentos são definidos de forma arbitrária a partir de uma máscara inicial, e não são baseados sobre um determinado modelo do objeto. Um mapa de distâncias é computado para cada fragmento através da EMD entre a região do fragmento no quadro atual e uma vizinhança no quadro seguinte. Então, um mapa global é obtido através da combinação dos mapas individuais, levando a uma métrica de distância mais robusta. O cálculo exaustivo de histogramas (necessário no EMD) é feito com base em imagens integrais [10], reduzindo assim o custo computacional, permitindo a obtenção de histogramas de múltiplas regiões retangulares de uma forma muito mais eficiente.

Outra característica apresentada em [5] é que através das relações geométricas entre os fragmentos permite-se a verificação da distribuição espacial dos pixels, informação esta que é perdida em algoritmos baseados em histogramas tradicionais. O algoritmo permite também utilizar diversas métricas para comparar dois histogramas e não somente métricas analíticas, tais como *Bhattacharyya* ou equivalentes como a métrica de *Matusita* [11]. Apesar do uso de imagens integrais, esse método ainda é relativamente custoso do ponto de vista computacional, tanto que os autores sugerem o uso apenas de histogramas monocromáticos (ao invés de cores), e não possui uma forma de atualização do modelo inicial, o que pode deteriorar o desempenho do rastreamento, quando a aparência do objeto mudar significativamente (mesmo que de forma gradual). A Figura 2.5 apresenta um comparativo deste rastreador, denominado *FragTrack*(representada pelo retângulo vermelho), com um rastreador que utiliza a técnica denominada *Meanshift* [4](representada pelo retângulo pontilhado azul).

Outro trabalho utilizando histogramas foi apresentado por Marimon e

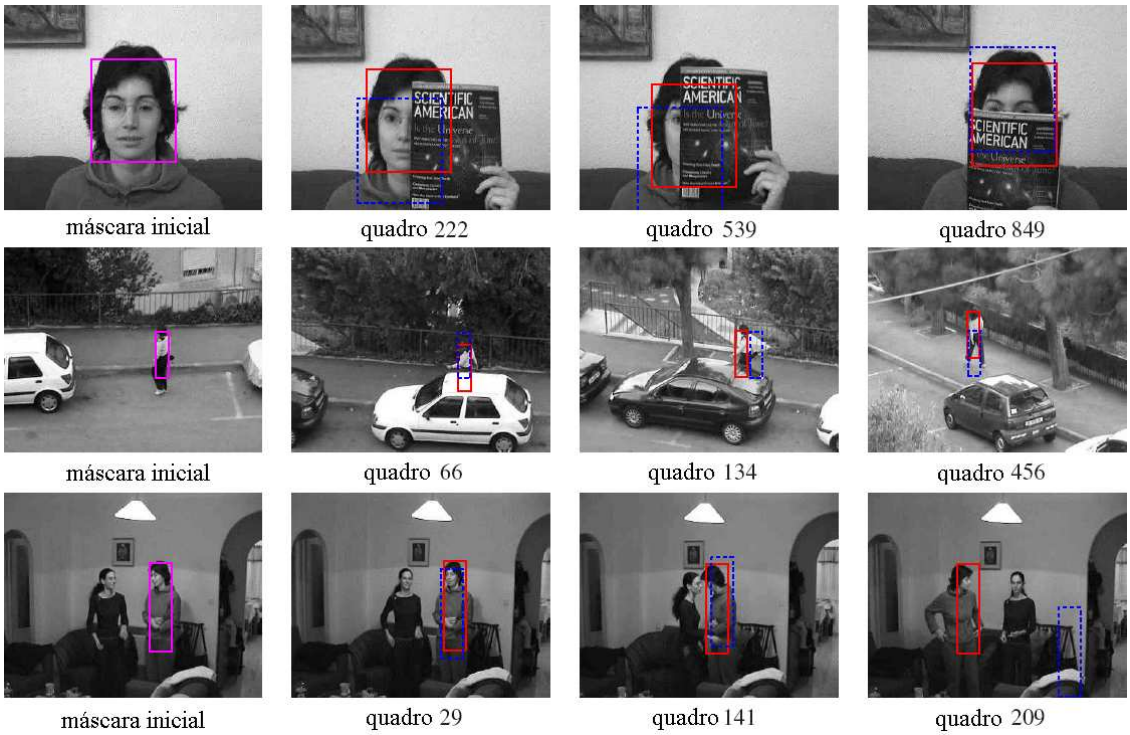


Figura 2.5 – Resultados do algoritmo *FragTrack* [5] (retângulos em vermelho), comparados com o rastreamento obtido pela técnica *Meanshift* [4] (representado pelos retângulos pontilhados em azul).

Ebrahimi [12], que combinaram a comparação dos histogramas de orientações de gradientes e comparação de máscaras através da correlação cruzada normalizada (NCC, ou *Normalized Cross-Correlation*), ajudando a estimar a rotação junto com a região de comparação. A técnica é dividida em dois passos, comparação dos histogramas de orientações de gradientes e a comparação de máscaras através da correlação cruzada normalizada.

Um método para estimar a rotação entre duas imagens é utilizado para aumentar o desempenho das comparações das máscaras. As vantagens apresentadas por este trabalho são: a não necessidade de treinar um classificador e a simplicidade do algoritmo com um número reduzido de parâmetros. Experimentos em seqüências de vídeos mostraram grande precisão e a correlação alcançada pela técnica proposta com uma boa aproximação da rotação da região. O método mostrou robustez diante de rotação, translação e escala (com fator de escala $f \in [0.8, 1.2]$) [12].

Embora o uso de imagens integrais tenha acelerado o cálculo de histogramas, a necessidade de memória é ainda um problema quando tratamos com histogramas multidimensionais: quando usa-se um histograma d-dimensional com N *bins*, é

necessário armazenar N^d valores (crescimento exponencial em d). De fato, tal limitação é declarada em [5, 13], e os algoritmos em [5, 12] usam somente histogramas de intensidade ($d = 1$).

Uma outra abordagem que também aproveita a estrutura de imagens integrais, mas para o cálculo de médias e matrizes de covariância em regiões retangulares quaisquer, foi proposta por Porikli et al. [13]. Os autores argumentam que a matriz de covariância é uma representação bem compacta de regiões caracterizadas por atributos multidimensionais. De fato, a matriz de covariância contém (d^2) entradas (como a matriz é simétrica, o valor de entradas diferentes é $d(d + 1)/2$), que representa um crescimento quadrático em d (ao contrário do crescimento exponencial do histograma). Nessa abordagem, o objeto de interesse é representado pela matriz de covariância, e no quadro adjacente é buscada a região cuja matriz de covariância minimiza um critério de distância. Dada a matriz $\boldsymbol{C}_i$ representando o objeto, e a matriz $\boldsymbol{C}_j$ representando um possível candidato, a distância adotada pelos autores é dada por

$$\rho\left(\boldsymbol{C}_i, \boldsymbol{C}_j\right) = \sqrt{\sum_{k=1}^{d} \ln^2 \lambda_k\left(\boldsymbol{C}_i, \boldsymbol{C}_j\right)}, \tag{2.3}$$

onde $\lambda_k\left(\boldsymbol{C}_i, \boldsymbol{C}_j\right)$ são os autovalores generalizados de $\boldsymbol{C}_i$ e $\boldsymbol{C}_j$, computados a partir de

$$\lambda_k \boldsymbol{C}_i \boldsymbol{x}_k - \boldsymbol{C}_j \boldsymbol{x}_k = 0; \qquad k = 1, \ldots, d. \tag{2.4}$$

Ainda, a cada intervalo de quadros pré-determinado é realizada a atualização do modelo, utilizando um conjunto de matrizes de covariância armazenadas e extraindo-se uma média intrínseca, usando a Álgebra de Lie. Embora essa técnica permita o uso de vetores de feições com dimensionalidade maior, o uso exclusivo de estatísticas de segunda ordem (matrizes de covariância) pode claramente levar a resultados errôneos, já que duas regiões com médias distintas podem ter exatamente a mesma matriz de covariância.

Avidan [14, 15] considera o rastreamento como um problema de classificação binária. Isso é feito pela construção de um vetor de características para cada pixel da imagem, e um conjunto de classificadores fracos é treinado para separar os pixels que pertencem ao objeto dos pixels que pertencem ao fundo. No quadro atual do

vídeo, um classificador forte (usando AdaBoost) é usado para testar os pixels e formar um mapa de confiança. O pico do mapa indica a direção do objeto e nessa região é utilizado o algoritmo *Meanshift* [4] para encontrá-lo. O rastreador ajusta a mudança de aparência dos objetos com o treinamento de novos classificadores fracos e atualizando o classificador forte. O rastreador mostrou-se bastante robusto, podendo trabalhar em uma grande variedade de cenários, com câmeras estáticas ou móveis, o rastreador pode manipular oclusões parciais pela rejeição dos pixels que pertencem a parte oclusa do objeto.

2.3 Rastreamento baseado em contorno

Outras abordagens bastante empregadas para o rastreamento de objetos são baseadas em contornos. Nos trabalhos [6, 16], são propostos modelos de rastreamento realizados pela atualização do contorno do objeto quadro a quadro. Essa atualização é realizada através da minimização de alguma função de energia, avaliada em uma região de interesse (em torno do contorno do objeto detectado no quadro anterior). A energia é obtida usando uma estrutura Bayesiana, e o rastreamento emprega características visuais como cor e textura que são obtidos por modelos semi-paramétricos. Tais características são mescladas através de um conceito de votação. O sistema trabalha com a manipulação de oclusões no qual mantém a forma anterior do objeto, recobrindo desta forma as partes oclusas. Os resultados apresentados mostram um sistema de rastreamento robusto e com bom desempenho. A Figura 2.6 ilustra em exemplo de rastreamento baseado em contorno usando o algoritmo descrito em [6] em um fundo bastante confuso.

No trabalho de Sato [17] é descrita uma transformada denominada TSV (*Temporal Spatio-Velocity*) para a extração das velocidades dos pixels em uma seqüência de imagens binárias. A transformada TSV é derivada da transformada Hough sobre janelas de imagens espaço-temporais. A intensidade de cada pixel na imagem TSV representa uma medida de probabilidade de ocorrência de cada pixel com velocidade instantânea na posição corrente. A limiarização das imagens TSV gera os componentes conexos binários *blobs* contendo pixels com similaridade de velocidade e posição. A transformada TSV provê uma forma eficiente de remover ruídos pela focalização nas velocidades estáveis/coerentes, e cria *blobs* com pouco

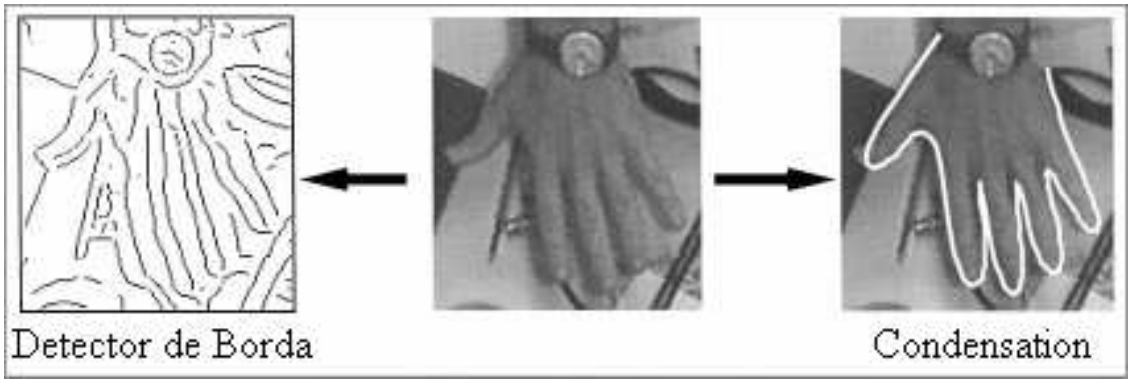


Figura 2.6 – Rastreamento usando o algoritmo *Condensation*. Trabalho de Isard e Blake [6]

ruído. A transformada TSV foi aplicada no rastreamento de figuras humanas em uma calçada, utilizando remoção do fundo para separar o primeiro plano do fundo da imagem, extraindo pessoas em pé e gerando uma seqüência de imagens binárias de uma dimensão. Com uma limiarização, as imagens TSV geram os *blobs* humanos, e as trajetórias humanas são obtidas pela combinação dos *blobs* segmentados pelas suas características.

2.4 Considerações sobre o capítulo

Este capítulo apresentou algumas técnicas para o rastreamento de objetos em seqüências de vídeo, utilizando uma gama de abordagens diferentes. Embora seja difícil uma comparação completa entre as técnicas, uma regra geral é que há um comprometimento entre a qualidade do rastreamento (acurácia, robustez a oclusões e mudanças de aparências, etc.) e o custo computacional.

Uma análise dessas técnicas indica que o uso de imagens integrais permite calcular rapidamente histogramas e estatísticas locais em regiões retangulares quaisquer da imagem. Por um lado, os histogramas fornecem informações mais completas do que matrizes de covariância, mas com um custo de armazenagem bem maior. Por outro lado, o uso apenas da matriz de covariância parece não capturar satisfatoriamente as características da região. Assim, a solução proposta deste trabalho é utilizar um modelo baseado na média e na matriz de covariância, aproveitando o uso de fragmentos múltiplos para obter maior robustez com relação a oclusões.

A técnica proposta se enquadra de acordo com a classificação proposta por Yilmaz [1] na classe de rastreamento baseada em regiões e ela é descrita no capítulo a seguir.

Capítulo 3

Modelo Proposto

Neste capítulo é apresentado um modelo de rastreamento de objetos, utilizando uma abordagem baseada em fragmentos. O modelo foi denominado algoritmo de rastreamento CPD (*Coherent Patch Displacement*).

3.1 Visão Geral

O algoritmo inicia-se com a seleção do alvo a ser rastreado no quadro inicial do filme. Em geral, o usuário seleciona manualmente o alvo (através do *mouse*), já que a definição do objeto de interesse a ser rastreado depende significativamente da aplicação. Entretanto, em algumas aplicações específicas, o processo da seleção do alvo pode ser automatizada (por exemplo, no problema de rastreamento de faces, um detector automático pode ser aplicado para obter o alvo no quadro inicial).

A região selecionada é então dividida automaticamente em fragmentos. O uso de modelos de rastreamento com múltiplos fragmentos, como em [5], apresentam maior robustez na ocorrência de oclusões parciais, mas as oclusões totais ainda são um grande desafio. O algoritmo proposto também emprega o uso de múltiplos fragmentos, mas a abordagem de combinação da informação de cada fragmento é diferente do modelo desenvolvido em [5], pois permite a inclusão de outras informações de movimento de forma simples, as quais serão vistas detalhadamente ao longo deste capítulo.

Cada fragmento é rastreado de forma independente, de acordo com uma métrica de casamento de máscaras. Um vetor de deslocamento de predição baseado no movimento da máscara global nos quadros anteriores também é calculado. O

resultado do rastreamento de cada fragmento e o vetor de predição são então combinados de uma forma robusta usando um WVMF (*Weighted Vector Median Filter*) para obter o deslocamento total do objeto, visando minimizar os problemas de oclusões parciais (e, em alguns casos, globais). Também é realizada uma abordagem de atualização do modelo, a fim de amenizar o problema da mudança de aparência do alvo rastreado ao longo do tempo. Essa mudança ocorre devido a algumas variações que ocorrem durante o processo de rastreamento, como por exemplo, mudanças de iluminação, sombras e alteração na forma do alvo, dentre outros.

Pode-se definir a idéia principal do algoritmo de rastreamento proposto nos seguintes passos:

1. dividir a região total em pequenos fragmentos retangulares não sobrepostos;
2. calcular o vetor de média e a matriz de covariância para cada fragmento, considerando que os pixels são representados por um vetor de características, tais como, cor, informação do gradiente, textura dentre outras características;
3. encontrar a melhor comparação para cada fragmento no quadro posterior da imagem usando a distância de *Bhattacharyya* como medida de dissimilaridade;
4. calcular um vetor de deslocamento de predição baseado no movimento da máscara global nos quadros anteriores;
5. combinar o resultado de cada fragmento em um vetor de deslocamento único para a máscara global;
6. atualizar o modelo (vetor de média e matriz de covariância) de todos os fragmentos;
7. voltar ao passo 3.

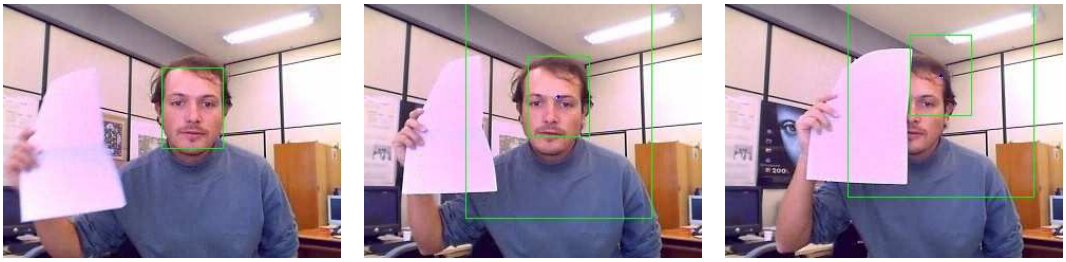
Cada passo será detalhado nas seções seguintes.

3.2 Seleção dos Fragmentos

A divisão da máscara global de rastreamento em múltiplos fragmentos apresenta uma informação adicional, que pode ser usada para resolver ambigüidades,

modelos mais complexos de movimento e também melhorar a robustez no que diz respeito a oclusões. A oclusão ocorre quando o objeto que está sendo rastreado fica atrás de outro objeto, ou pode ocorrer também, quando ele é encoberto por alguma região do fundo. As oclusões podem ser consideradas um dos principais desafios nos algoritmos de rastreamento de objetos, pois a partir de uma oclusão, os dados recebidos pelo algoritmo da região rastreada passam a ser divergentes dos que estavam sendo recebidos anteriormente.

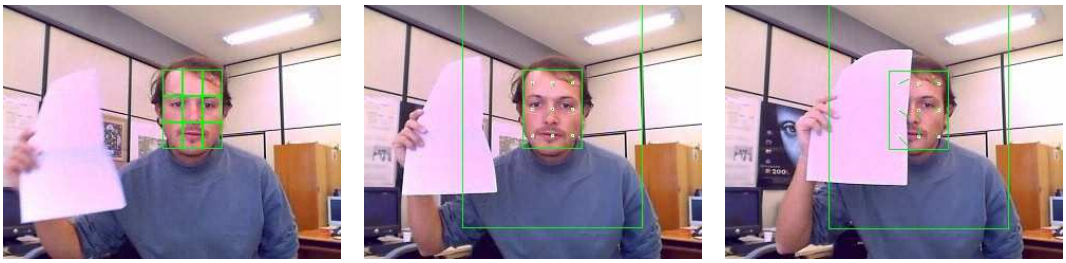
Segundo [5], a divisão da região do objeto em pequenos fragmentos tende a prover melhor recuperação contra oclusões parciais, desde que somente alguns dos fragmentos sejam realmente afetados. Partindo desta idéia, o algoritmo proposto neste trabalho também realiza a divisão da máscara inicial em fragmentos. A Figura 3.1 ilustra-se o problema de oclusões parciais no rastreamento de objetos, em que a máscara que descreve a região de interesse é afetada significativamente pela folha de papel colocada na frente da face. Já a Figura 3.2 ilustra o resultado do rastreamento usando fragmentos. Nota-se que a utilização de fragmentos é robusta com relação a oclusões parciais, pois apenas alguns fragmentos são afetados, enquanto que os outros corretamente acompanham o objeto de interesse.



(a) (b) (c)

Figura 3.1 – Exemplo de quadros do rastreamento da face de uma pessoa em uma seqüência de vídeo, o deslocamento é obtido pela medida da distância sobre um fragmento somente, sofrendo com a oclusão.

A seleção ideal de fragmentos deve conduzir a um erro mínimo global exatamente na posição do objeto rastreado. Claramente, esta seleção é altamente dependente da medida de dissimilaridade usada para comparar os fragmentos. Neste trabalho, a seleção é realizada da seguinte forma: A máscara retangular da região representando o alvo é dividida em uma grade uniforme de $n \times m$ fragmentos adjacentes retangulares. Essa subdivisão uniforme pode produzir fragmentos



(a) (b) (c)

Figura 3.2 – Exemplo de quadros do rastreamento da face em uma seqüência de vídeo, o deslocamento final é obtido pela medida da distância de uma região de 3 x 3 fragmentos, reduzindo o problema da oclusão.

similares, o que possivelmente causaria o casamento errôneo de alguns fragmentos. Entretanto, espera-se que os fragmentos restantes que apresentam um casamento correto auxiliarão no cálculo do rastreamento da máscara global, tornando-o mais robusto.

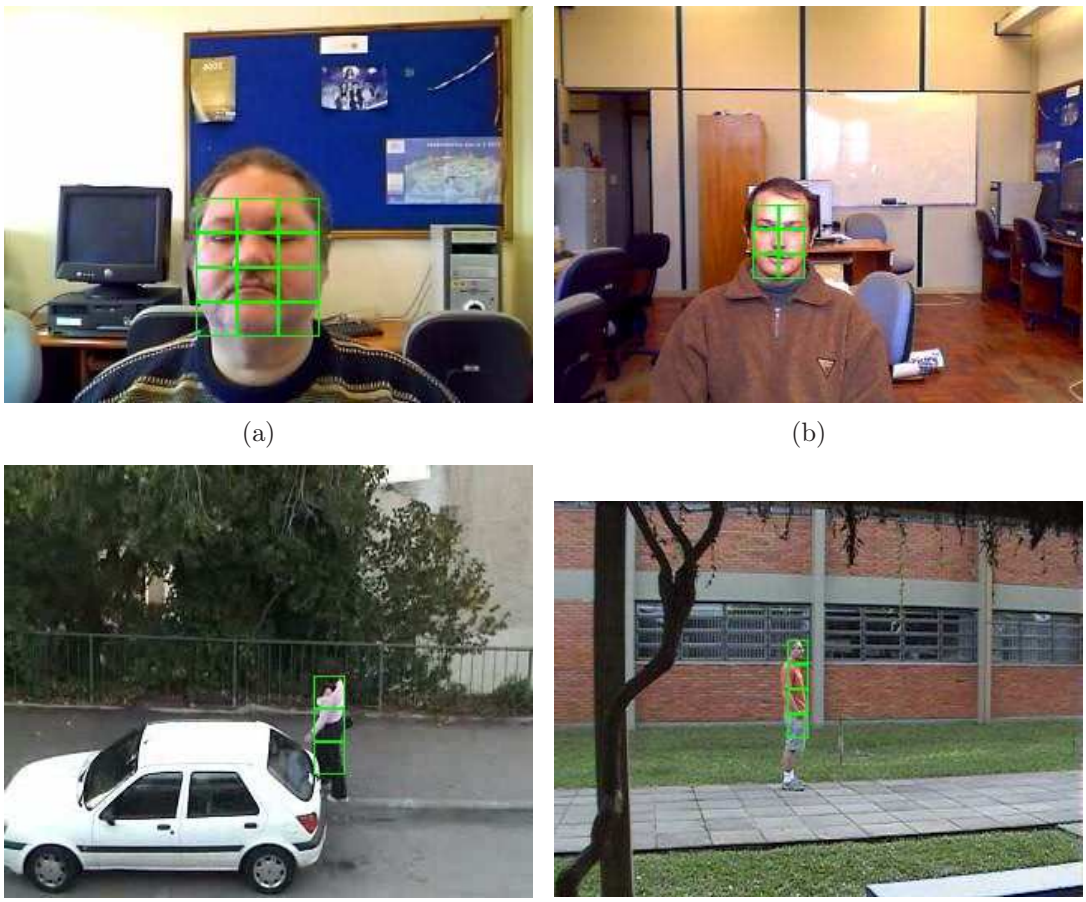
Como será descrito na próxima seção, cada fragmento é representado através do vetor de média e da matriz de covariância. Portanto, os fragmentos selecionados devem possuir um tamanho mínimo suficiente para fornecer estimativas confiáveis destes parâmetros estatísticos. Se a máscara original é uma região retangular com dimensão $N_T \times M_T$ pixels, e n_p é o tamanho de lado mínimo desejado para cada fragmento, a máscara é subdividida em $n \times m$ fragmentos, onde

$$n = \max\left\{1, \left[\frac{N_T}{\min\{N_T, M_T, n_p\}}\right]\right\}, m = \max\left\{1, \left[\frac{M_T}{\min\{N_T, M_T, n_p\}}\right]\right\}. \quad (3.1)$$

Aqui, $[\cdot]$ representa o truncamento para o menor valor inteiro. A Equação (3.1) tenta dividir a máscara original em fragmentos com uma dimensão $n_p \times n_p$, exceto quando uma (ou ambas) dimensões da máscara forem menores que n_p. Experimentalmente, foi definido $n_p = 20$ como o tamanho padrão de lado desejado.

A Figura 3.3 ilustra a seleção dos fragmentos para algumas seqüências de vídeo analisadas neste trabalho. O retângulo externo é manualmente criado pelo usuário através do *mouse*, e os sub-retângulos menores internos são os fragmentos obtidos através da Equação (3.1).

O procedimento para a seleção dos fragmentos adotado neste trabalho é similar ao trabalho [5], mas a abordagem para a combinação das informações de distância de cada fragmento é totalmente distinto. Em [5], o mapa de distância calculado



(a) (b)

(c) (d)

Figura 3.3 – Seleção automática dos fragmentos de uma máscara retangular definida manualmente pelo usuário.

para cada fragmento é agrupado em um mapa de distância unificado, e o vetor de deslocamento de toda a região é então calculado. Embora tal procedimento apresente bons resultados, esse método não permite a inclusão de outras informações relevantes (tal como uma posição de predição, baseada em modelos de movimento) de uma forma simples, o que torna a abordagem mais suscetível a falhas no que diz respeito a oclusões totais ou próximas da total.

Na solução da abordagem proposta, em vez de criar um simples mapa de distância único, o vetor de deslocamento é calculado independentemente para cada fragmento, e o movimento final do objeto é obtido pela combinação dos vetores de deslocamento individuais. Portanto, a inclusão de outras abordagens de movimento é instantânea, como será discutido na seção 3.5.

3.3 Casamento dos Fragmentos

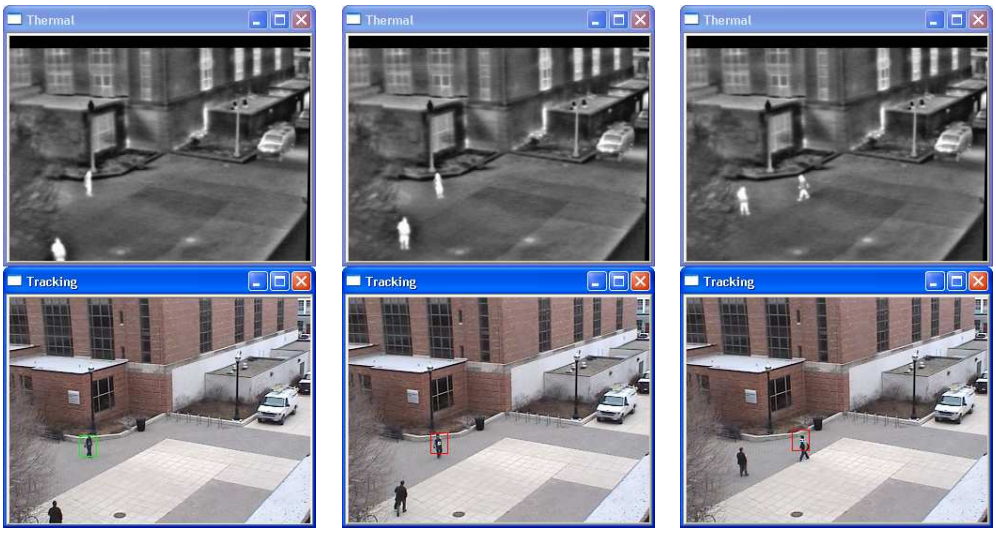
Há vários métodos para a comparação de regiões baseadas em estatísticas dos pixels. Alguns autores [5, 12] empregam a comparação direta de histogramas usando métricas de similaridade. Já em outros trabalhos, as comparações de regiões são realizadas pela avaliação de parâmetros estatísticos que descrevem a região, como por exemplo o algoritmo de rastreamento da covariância proposto em [13].

Um problema dos métodos baseados em histograma é o rápido aumento na complexidade computacional com o aumento da dimensão do espaço das características. Se N_b *bins* são usados para representar cada uma das d dimensões, o histograma requer N_b^d *bins* para uma representação total. Por exemplo, somente valores de intensidade são usados no algoritmo de rastreamento baseado em histogramas proposto em [5], já que os autores alegam que usando três canais de cores deverá deixar o algoritmo muito custoso.

Por outro lado, descritores de características estatísticas usualmente requerem uma pequena quantidade de memória e as métricas de comparação são simples para implementar, mas a escolha das características e dos descritores estatísticos ainda é uma tarefa desafiadora. Por exemplo, o algoritmo de rastreamento descrito em [13] usa uma representação compacta de cada região (a matriz de covariância, que requer $d\,(d+1)\,/2$ parâmetros) para cobrir a região de comparação. Contudo está claro que duas regiões comparadas podem ser bastante distintas, mas ter exatamente a mesma matriz de covariância, conduzindo a possíveis casamentos errôneos.

É importante salientar que o CPD foi projetado para poder utilizar qualquer conjunto de descritores de características, tais como, a intensidade, o gradiente em x ou y, diversos espaços de cores (*RGB*, *YCrCb*) dentre outros. Dados multimodais também podem ser facilmente explorados. Por exemplo, a Figura 3.4 ilustra o rastreamento usando informações de cores combinadas com dados de imagens termais (vídeo original disponível em [18]).

A definição da métrica de dissimilaridade em si é também um assunto em aberto. Rubner e colegas [19] realizaram uma comparação empírica de vários algoritmos de casamento (incluindo a distância de *Bhattacharyya*, divergência KL (*Kullback-Leibler*) e a EMD, entre outras) focada para características de cor e textura, e concluíram que não há medida com um melhor desempenho em qualquer



(a) imagem termal inicial (b) imagem termal 2 (c) imagem termal 3

Figura 3.4 – Exemplos de quadros de rastreamento usando RGB combinado com imagens termais.

caso. De fato, os autores argumentam que a métrica ótima depende da tarefa específica.

Para alguns modelos paramétricos, há expressões com forma fechada para algumas métricas clássicas para comparar duas funções de densidade de probabilidade (tal como a distância de *Bhattacharyya* e a divergência de KL) que envolvem somente os parâmetros estatísticos de ambas as distribuições. Se a estimação desses parâmetros for barata do ponto de vista do custo computacional, essas expressões de forma fechada são muito rápidas de calcular.

Um modelo que pode ser calculado rapidamente e que conduz a uma expressão de forma fechada para a distância de *Bhattacharyya* e divergência KL é a distribuição Gaussiana multivariada. Essa distribuição é caracterizada por um vetor de média $\boldsymbol{\mu}$ e uma matriz de covariância $\boldsymbol{C}$, que podem ser estimadas usando dados amostrais por

$$\boldsymbol{\mu} = \frac{1}{N}\sum_{\boldsymbol{x}\in S}\boldsymbol{x}, \tag{3.2}$$

$$\boldsymbol{C} = \frac{1}{N}\sum_{\boldsymbol{x}\in S}(\boldsymbol{x}-\boldsymbol{\mu})(\boldsymbol{x}-\boldsymbol{\mu})^T, \tag{3.3}$$

onde S é o conjunto usado para estimar a distribuição, e $N = \#S$ é o número de exemplos (amostras). Ambos $\boldsymbol{\mu}$ e $\boldsymbol{C}$ podem ser calculados eficientemente em

qualquer região retangular da imagem, usando o algoritmo proposto em [20], que se utiliza de "imagens integrais". De fato, o custo computacional para o cálculo de uma imagem integral em uma região retangular com dimensões $H \times W$, usando vetores de características de dimensão d é $\mathcal{O}(HWd^2)$, e o custo para avaliar $\boldsymbol{\mu}$ e $\boldsymbol{C}$ dentro de qualquer sub-região retangular é $\mathcal{O}(d^2)$.

Dadas duas distribuições Gaussianas com os parâmetros $(\boldsymbol{\mu}_1, \boldsymbol{C}_1)$ e $(\boldsymbol{\mu}_2, \boldsymbol{C}_2)$, a distância de *Bhattacharyya* é dada por

$$B = \frac{1}{8}\left(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\right)^T \left[\frac{\boldsymbol{C}_1 + \boldsymbol{C}_2}{2}\right]^{-1} \left(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\right) + \frac{1}{2}\ln\left(\frac{|(\boldsymbol{C}_1 + \boldsymbol{C}_2|)/2}{\sqrt{|\boldsymbol{C}_1||\boldsymbol{C}_2|}}\right), \tag{3.4}$$

e a divergência simétrica de *Kullback-Leibler* é dada por

$$KL = \frac{1}{2}\text{trace}\left(\boldsymbol{C}_1^{-1}\boldsymbol{C}_2 + \boldsymbol{C}_2^{-1}\boldsymbol{C}_1\right) + \frac{1}{2}\left(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\right)^T \left[\boldsymbol{C}_1^{-1} + \boldsymbol{C}_2^{-1}\right]\left(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\right) - d. \tag{3.5}$$

Em ambas expressões, a inversão de matriz é a operação mais custosa, que pode ser calculada tradicionalmente com $\mathcal{O}(d^3)$ operações (embora há alguns algoritmos [21] que reduzam a complexidade para $\mathcal{O}(d^{2.376})$).

Um problema com a hipótese Gaussiana é que, em geral, a distribuição dos vetores de características (e.g. canais de cores RGB) não é nem mesmo unimodal. No rastreamento de pessoas, por exemplo, uma pessoa pode ter uma camiseta de uma cor e as calças de outra, conduzindo a uma distribuição bimodal.

Na abordagem proposta, visto que a região de interesse é dividida em pequenos fragmentos, assume-se que a distribuição dos vetores de características dentro de cada fragmento é aproximadamente homogênea, e o desvio da hipótese normal pode não ser muito grande. Uma possível desvantagem do uso de fragmentos é que, quando se tem uma grande região homogênea em um objeto, ela poderá ter vários fragmentos similares, que pode gerar casamentos errôneos entre os mesmos. Outra desvantagem é que os fragmentos não podem ser muito pequenos devido à informação estatística que eles fornecem. Contudo, o deslocamento final de todo o objeto é calculado usando-se uma abordagem robusta de médias com pesos dos deslocamentos individuais, assim os fragmentos com resultados errados são amortizados pelos outros restantes. De fato, no capítulo 4, os resultados apresentados demonstram a robustez da técnica proposta.

Deve-se salientar que a utilização de uma quantidade maior de fragmentos (ou seja, geração de fragmentos menores) tende a estar em maior consonância com a hipótese Gaussiana dentro de cada fragmento. Por outro lado, fragmentos muito pequenos podem gerar estimativas não confiáveis para a média e matriz de covariância, já que há poucas observações no fragmento. De fato, um desafio ainda em aberto é achar uma subdivisão em fragmentos que sejam ao mesmo tempo compatíveis com a hipótese Gaussiana e ofereçam uma amostra suficiente para o cálculo das estatísticas.

Para obter o vetor de deslocamento individual v_i para cada fragmento i, uma região de busca com dimensão $S_x \times S_y$ é colocada no centro de cada fragmento no quadro corrente. A distância de *Bhattacharyya* entre o fragmento e todos os fragmentos candidatos na região de busca é calculada exaustivamente, e o fragmento selecionado será aquele que apresentar a menor distância.

3.4 Predição do Movimento

A predição do movimento pode ser muito útil para reduzir a área de busca ou para eliminar movimentos falsos devido a oclusões. Há dois tópicos importantes referentes à predição de movimento: como realizá-la de uma forma rápida e como utilizar as informações obtidas de uma forma fácil e coerente. Filtro de Kalman é uma ferramenta amplamente conhecida para calcular a predição de uma posição futura, com aplicações em problemas de rastreamento, como em [22]. Embora o filtro de Kalman alcance uma boa predição, ele possui certo custo computacional.

Neste trabalho, propõe-se o uso da técnica *Double Exponential Smoothing* proposta em [23] para a predição de posição. De acordo com [23], essa técnica de predição é de simples implementação, muito mais rápida que o Filtro de Kalman, e com um desempenho de predição equivalente. Dada uma série temporal de vetores $\boldsymbol{v}_t$, a predição no tempo $t+\tau$ conforme [23] é dada por

$$\boldsymbol{v}_{t+\tau} = \left(2 + \frac{\alpha\tau}{1-\alpha}\right)\boldsymbol{\rho}_t - \left(1 + \frac{\alpha\tau}{1-\alpha}\right)\boldsymbol{\rho}_t^{[2]}, \tag{3.6}$$

onde $\boldsymbol{\rho}_t$ e $\boldsymbol{\rho}_t^{[2]}$ são variáveis auxiliares calculadas através das seguintes expressões:

$$\boldsymbol{\rho}_t = \alpha \boldsymbol{v}_t + (1-\alpha)\boldsymbol{\rho}_{t-1}, \tag{3.7}$$

$$\boldsymbol{\rho}_t^{[2]} = \alpha \boldsymbol{\rho}_t + (1-\alpha)\boldsymbol{\rho}_{t-1}^{[2]}. \tag{3.8}$$

Aqui, α é o fator de decaimento exponencial (pequenos valores para α produzem predições suavizadas). Na abordagem proposta, foi determinado experimentalmente o valor $\alpha = 0.1$, e foi utilizado o valor $\tau = 1$ para gerar um vetor de predição $\boldsymbol{v}_p = \boldsymbol{v}_{t+1}$ no quadro adjacente $t+1$.

3.5 Combinando Informações dos Fragmentos e o Movimento de Predição

A métrica de comparação descrita na seção 3.3 e o movimento de predição descrito na seção 3.4 geram a partir de um conjunto de N_p fragmentos, vetores de deslocamento $\boldsymbol{v}_i$ e um vetor de deslocamento de predição $\boldsymbol{v}_p$, em um total de N_p+1 informações de movimento individual. Por simplicidade de notação, será definido $\boldsymbol{v}_{N_p+1} = \boldsymbol{v}_p$.

Se o movimento for apenas translacional, todos esses vetores devem ser similares. Contudo, os vetores de deslocamento $\boldsymbol{v}_j$ em um ou mais fragmentos podem ser distorcidos quando ocorrer oclusões parciais, houver mais de um fragmento sobre regiões uniformes, ou houver mudanças de iluminação. Para combinar esses vetores, o simples cálculo da média dos vetores de deslocamento individuais não fornece bons resultados, pois a média pode ser afetada significativamente por um simples vetor discrepante dos demais. Assim, uma abordagem mais adequada é o uso de WVMFs (*Weighted Vector Median Filters*), que implicitamente rejeitam esses vetores discrepantes da média esperada [24].

Para o cálculo da WVMF, primeiramente é calculada a soma das distâncias entre cada vetor e os demais:

$$D_j = D(\boldsymbol{v}_j) = \sum_{i=1}^{N_p+1} \left\| \boldsymbol{v}_j - \boldsymbol{v}_i \right\|, \quad \text{para} \quad j = 1, ..., N_p+1, \tag{3.9}$$

onde N_p+1 é o número total de vetores, e $\|\cdot\|$ é uma norma vetorial (neste trabalho, se propõe o uso da norma L^1, por sua simplicidade e velocidade de cálculo). O vetor filtrado $\boldsymbol{v}_f$ é então definido por

$$\boldsymbol{v}_f = \frac{\sum_{i=1}^{N_p+1} w_i \boldsymbol{v}_i}{\sum_{i=1}^{N_p+1} w_i}, \tag{3.10}$$

onde $w_i = f(D_i)$, e f é uma função monótona decrescente não negativa, para que os vetores mais distantes da mediana tenham um peso menor.

Neste trabalho, também é incluído um peso com base na métrica de similaridade, para que fragmentos com menor erro de comparação possuam maior peso no cálculo da WVMF. Em particular, usando a distância de *Bhattacharyya* B_i para cada fragmento i, calculada de acordo com a Equação (3.4), a proposta modificada da WVMF é dada por

$$w_i = g\left(D_i, B_i\right), \tag{3.11}$$

onde $g(x, y)$ é outra função monótona decrescente não negativa quando consideradas as variáveis x e y individualmente, i.e., $g_x(x, y) < 0$ e $g_y(x, y) < 0$, $\forall x, y > 0$.

Uma classe de funções 1D que tem mostrado bons resultados para a remoção de pixels discrepantes (*outliers*) em imagens coloridas ruidosas [25] é $\exp(-x^r/\xi)$, onde ξ e r são parâmetros escolhidos experimentalmente para gerar bons resultados de modo geral ao filtro. A escolha deste trabalho por uma função 2D $g(x, y)$ segue a mesma idéia, e é dada por

$$g(x, y) = e^{-\left[(x/\beta)^2+(y/\gamma)^2\right]}, \tag{3.12}$$

onde β e γ controlam o decaimento de g como uma função de x e y, respectivamente. Valores menores para β e γ priorizam vetores de deslocamento $\boldsymbol{v}_i$ que apresentam menor distância D_i e menor erro de *Bhattacharyya* B_i, respectivamente. À medida que β e γ aumentam, o WVFM tende a um simples filtro de média, já que todos os pesos w_i tendem a ser similares.

Neste trabalho, β foi selecionado adaptativamente através de $\beta = \min_i D_i$, para que o decaimento de $g(x, y)$ seja grande para os vetores de deslocamento distantes do valor mínimo. Também foi proposto o valor $\gamma = 0.15$, pois os resultados dos experimentos indicaram que os fragmentos sem oclusões normalmente apresentaram erros de *Bhattacharyya* em torno ou abaixo de 0.15. De fato, coeficiente de *Bhattacharyya* $b = e^{-B}$ refere-se ao limite do erro de classificação entre duas classes equiprováveis [26], e isso é usado como uma medida de similaridade em outros trabalhos [4, 27]. Portanto, $B = 0.15$ conduz a $b \approx 0.86$, valor coerente com a medida de similaridade relatada em [27].

Deve-se notar que não existe um erro de casamento B_{N_p+1} associado ao vetor do movimento de predição $\boldsymbol{v}_{N_p+1}$, o qual é requerido na Equação (3.11). Se um valor pequeno é atribuído a B_{N_p+1}, o deslocamento da predição terá um peso maior na WVMF, e o oposto ocorrerá para valores grandes atribuídos a B_{N_p+1}. A primeira situação é adequada a situações de oclusão total, pois os fragmentos não forneceram uma informação confiável. Por outro lado, a segunda situação é apropriada para condições normais de rastreamento, onde as informações fornecidas pelos fragmentos devem ser predominantes. Na abordagem proposta, o valor de B_{N_p+1} é calculado adaptativamente, baseado sobre o erro dos fragmentos dos quadros anteriores.

Para cada quadro t, há $N_p + 1$ distâncias de *Bhattacharyya* $B_i(t)$ relativas aos N_p fragmentos e ao vetor de predição[1] $\boldsymbol{v}_{N_p+1}$. O erro representativo $B(t)$ para toda máscara no frame t é obtido da informação do deslocamento individual que é mais coerente com o deslocamento de toda máscara, i.e.

$$B(t) = B_j(t), \quad \text{onde} \quad j = \underset{i}{\operatorname{argmin}} \left\| \boldsymbol{v}_f - \boldsymbol{v}_i \right\|. \tag{3.13}$$

Em condições normais de rastreamento, supõe-se que $B(t)$ seja representado pelas distâncias de *Bhattacharyya* de um fragmento existente. Por outro lado, durante as oclusões totais, espera-se que os fragmentos produzam vetores de deslocamentos adulterados ou falsos (provavelmente com grandes erros de casamento), e $B(t)$ tende a ser representado pela distância de *Bhattacharyya* do vetor de predição do quadro anterior.

Finalmente, o erro selecionado B_{N_p+1} para o vetor de predição $\boldsymbol{v}_{N_p+1}$ é dado

[1] Exceto para o primeiro quadro, onde B_{N_p+1} não é definido.

por

$$B_{N_p+1} = \underset{k\in\{t-T_p,\ldots,t\}}{\text{mediana}} B(k), \tag{3.14}$$

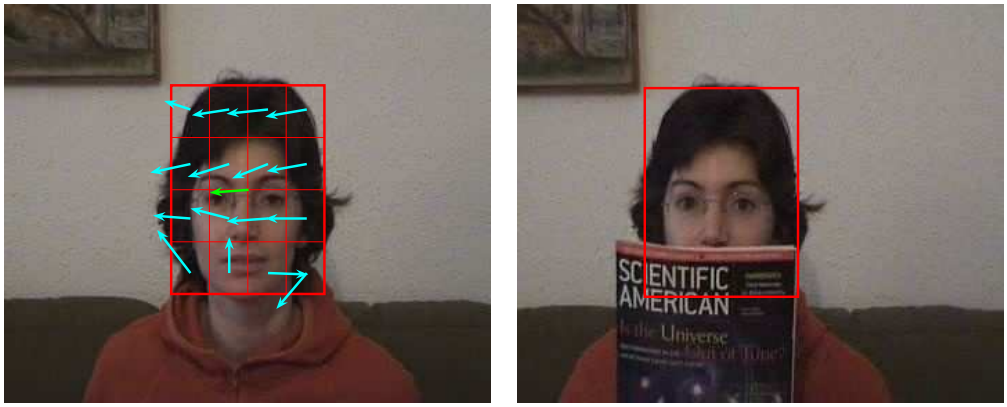
onde T_p é a janela temporal na qual a mediana será calculada (neste trabalho, foi definido experimentalmente o valor $T_p = 30$). Novamente, a lógica para esta escolha é que quando o objeto começar a sofrer oclusão, B_{N_p+1} deva retornar os erros de casamento de fragmentos comparados corretamente dos quadros anteriores, tendendo a apresentar erros menores do que dos fragmentos rastreados correntemente (os quais estão sob oclusão). Conseqüentemente, o vetor de predição tende a possuir maior peso durante as oclusões.

Deve-se verificar também que há somente um vetor de predição $\boldsymbol{v}_{N_p+1}$, e N_p vetores de deslocamentos gerados dos fragmentos. Se N_p é suficientemente grande, o deslocamento total da máscara tende a ser dominado pelos deslocamentos dos fragmentos $\boldsymbol{v}_i$, $i = 1, ..., N_p$, mesmo se B_{N_p+1} é pequeno. Para tratar esse problema, foi introduzido um fator de compatibilidade $0 \leq c \leq 1$ para redefinir o peso de $\boldsymbol{v}_{N_p+1}$ como uma função de N_p. Mais exatamente, o peso w_{N_p+1} na Equação (3.10) é recalculado através de

$$w_{N_p+1} = cN_p g(D_{N_p+1}, B_{N_p+1}). \tag{3.15}$$

Pequenos valores para c diminuem o peso de $\boldsymbol{v}_{N_p+1}$, e o oposto acontece para valores grandes de c. Neste trabalho foi usado $c = 0.5$ em todos os experimentos.

Um exemplo do procedimento para a combinação dos vetores de deslocamento individuais para cada fragmento, com vistas a obter o deslocamento global, é ilustrado na Figura 3.5. A Figura 3.5(a) mostra a máscara inicial (retângulo externo) e os fragmentos individuais (sub-retângulos menores). Os vetores de cada sub-retângulo indicam o deslocamento individual de cada fragmento, o vetor em verde, no centro, refere-se ao deslocamento global da máscara. A máscara deslocada no quadro subseqüente é visualizada na Figura 3.5(b). É interessante verificar que os quatro fragmentos na parte inferior da máscara foram ocultados de um quadro para outro, e deslocamentos errôneos dos vetores foram criados para os respectivos fragmentos. Contudo, o WVFM descartou a influência destes vetores, conduzindo o deslocamento global da máscara de forma correta.



(a) (b)

Figura 3.5 – Exemplo do procedimento adaptado obtendo o deslocamento de toda a máscara baseado no deslocamento individual de cada fragmento através da WVMFs.

3.6 Modelo de Atualização

Para tratar os problemas de mudanças dos objetos (aparência, iluminação, etc.) ao longo do tempo, o modelo deve ser atualizado para melhorar o resultado do rastreamento. Assim, propõe-se que a matriz de covariância e o vetor de média de cada fragmento sejam atualizados todos a cada T_u quadros, considerando a média e a covariância do modelo nos quadros anteriores. Uma abordagem simples e direta seria calcular a covariância usando as máscaras rastreadas em diferentes instantes de tempo, como foi citado por Porikli et al. [13]. No entanto, como é indicado pelos próprios autores, o custo de calcular a matriz de covariância usando T fragmentos com dimensão $M \times N$ cada é $\mathcal{O}(NMTd^2)$, o qual é muito custoso além de necessitar de grande quantidade de memória. De fato, eles propuseram um esquema de atualização baseada na Álgebra de Lie para superar este problema.

Neste trabalho, é proposta uma abordagem rápida e com baixo custo de memória para a atualização do vetor de média e da matriz de covariância. A abordagem proposta é equivalente a calcular essas estatísticas diretamente, sendo mais simples e rápido do que ao algoritmo baseado na Álgebra de Lie utilizado em [13].

Sejam $\boldsymbol{\mu}_1$ e $\boldsymbol{C}_1$ a média e a covariância do modelo no quadro t, e sejam $\boldsymbol{\mu}_2$ e $\boldsymbol{C}_2$ os mesmos parâmetros para o quadro $t + T_u$ (o quadro corrente). A média $\boldsymbol{\mu}$ e

a matriz de covariâcia $\boldsymbol{C}$ atualizadas são dadas por

$$\boldsymbol{C} = (1-w)\left(\boldsymbol{C}_1 + \boldsymbol{\mu}_1\boldsymbol{\mu}_1^T\right) + w\left(\boldsymbol{C}_2 + \boldsymbol{\mu}_2\boldsymbol{\mu}_2^T\right) - \boldsymbol{\mu}\boldsymbol{\mu}^T, \tag{3.16}$$

$$\boldsymbol{\mu} = (1-w)\boldsymbol{\mu}_1 + w\boldsymbol{\mu}_2, \tag{3.17}$$

onde $0 < w < 1$ é a taxa de atualização do modelo, tal que $w \approx 1$ indica uma atualização rápida, enquanto $w \approx 0$ indica uma atualização mais lenta do modelo. Se $\boldsymbol{\mu}_1$, $\boldsymbol{C}_1$ referem-se um conjunto S_1 contendo N vetores de características, e $\boldsymbol{\mu}_2$, $\boldsymbol{C}_2$ referem-se a um conjunto S_2 contendo M vetores de características, pode-se mostrar (ver prova no apêndice) que a média e a covariância usando todos os vetores em $S_1 \cup S_2$ podem ser obtidos com as Equações (3.16) e (3.17) usando $w = M/(M+N)$ e $1 - w = N/(M+N)$. Experimentalmente definiu-se $w = 0.0$.

O custo computacional desse procedimento é da ordem $\mathcal{O}(d^2)$, e não depende das dimensões dos fragmentos. Além disso, como as Equações (3.16) e (3.17) são aplicadas recursivamente para todo T_u quadros, a média e a matriz de covariância resultante traz informações sobre os quadros anteriores, capturando as variações temporais dos fragmentos. A forma proposta também é muito eficiente em termos de memória: somente os descritores anteriores ($\boldsymbol{\mu}_1$, $\boldsymbol{C}_1$) e o descritores atuais ($\boldsymbol{\mu}_2$, $\boldsymbol{C}_2$) são usados, e as amostras usadas para calcular o modelo nos quadros anteriores não são necessárias. Salienta-se que a taxa de atualização do modelo T_u pode ser definida pelo usuário, mas todos os resultados apresentados neste trabalho utilizaram o valor $T_u = 1$, ou seja, atualização a cada quadro.

Capítulo 4

Resultados Experimentais

Neste capítulo, serão apresentados os resultados experimentais obtidos com o algoritmo desenvolvido neste trabalho, denominado *Coherent Patch Displacement* (CPD). A validação experimental foi realizada, qualitativamente, pela inspeção visual dos resultados do rastreamento e, quantitativamente, pela comparação dos erros de rastreamento produzidos pela solução proposta e por duas técnicas consideradas estado-da-arte: o algoritmo *Meanshift* [4] e o algoritmo *FragTrack* [5].

Também, neste capítulo, é apresentada uma versão do modelo proposto, focado ao problema de rastreamento de faces. Nesse caso, para obter a seleção inicial do alvo a ser rastreado, foi utilizado um detector de faces automático [28] ao invés da seleção manual através do *mouse*.

4.1 Comparativo com Abordagens do Estado-da-Arte

Os resultados apresentados nesta seção foram obtidos usando o algoritmo CPD implementado na linguagem C++, e rodando em um computador PC com as seguintes configurações: processador *Pentium Core Duo 2.33GHz*, memória principal de 1GB RAM, sem placa aceleradora gráfica e usando sistema operacional *Microsoft Windows XP*.

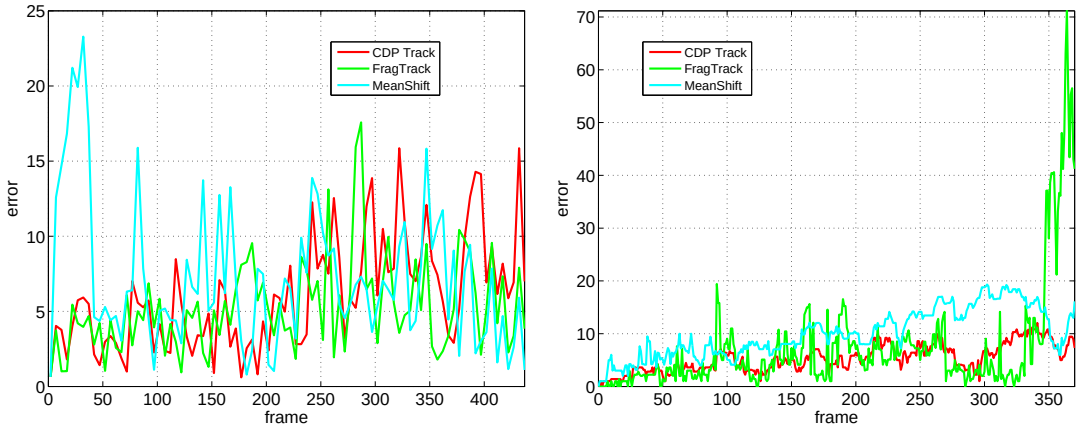
As análises foram realizadas utilizando cinco diferentes seqüências de vídeo, chamadas de: *Mulher*, *Face1*, *Face2*, *CAVIAR* e *Homem*. A seqüência *Mulher* ilustra uma senhora caminhando em uma calçada. Durante seu trajeto, ela sofre oclusões parciais de carros estacionados na via e também sofre mudanças de iluminação e sombreamento através das árvores. A câmera neste vídeo não é estática, e a

seqüência foi obtida em [29]. A seqüência *Face1* apresenta a face de uma pessoa que sofre oclusão parcial e total. O vídeo *Face2* ilustra a face de uma pessoa com movimentos suaves e sofrendo oclusão total, além de sofrer mudanças de iluminação. A seqüência *CAVIAR* é um dos vídeos do projeto *CAVIAR* [30], no qual algumas pessoas caminham em um corredor de um *shopping center*. Nesse vídeo, aparece uma mulher caminhando em uma área interna, e sofrendo apenas oclusão parcial. A câmera, tanto neste vídeo como nas seqüências *Face1* e *Face2*, está fixa. Finalmente, a seqüência de vídeo *Homem*, mostra uma pessoa caminhando em uma área externa, que sofre oclusões totais por diversas vezes, mudanças de iluminação e sobreamentos. A câmera nesse vídeo não é fixa, acompanhando a pessoa enquanto ela realiza seu trajeto. As seqüências de vídeo *Mulher*, e *CAVIAR* estão disponíveis com os dados de *ground truth*[1]. Nos vídeos *Face1*, *Face2* e *Homem*, o *ground truth* foi feito de forma manual.

Para essas cinco seqüências de vídeo, foram calculados dois conjuntos de dados: o tempo de execução e o erro de rastreamento, que é definido como a distância Euclidiana entre a posição rastreada e a posição do *ground truth* em cada quadro. Para rodar o algoritmo CPD em todos os experimentos, foi utilizada a distância de *Bhattacharyya* como métrica de distância. No algoritmo *FragTrack*, a distância empregada foi a EMD e foram usados 16 *bins* para a representação e comparação dos histogramas, configuração indicada pelos autores [5] como a que apresenta os melhores resultados. A região de busca usada no algoritmo CPD e no algoritmo *FragTrack* foi de 30×30 pixels.

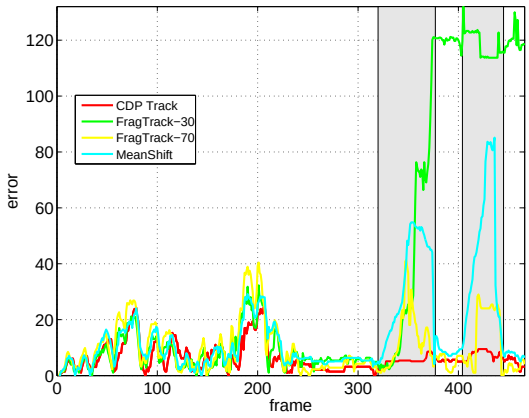
Na configuração do algoritmo CPD, para a obtenção do modelo de cada fragmento (a matriz de covariância e o vetor de médias), foram empregadas cinco características para todas as seqüências testadas, sendo elas: os três canais de cores (RGB) e os dois componentes do gradiente calculados do componente de luminância. É importante enfatizar que embora o algoritmo do CPD apresente outros parâmetros que podem ser ajustados, (n_p, w, T_p, γ e c), eles permaneceram com os valores inalterados em todos os experimentos realizados, com os valores padrão definidos no capítulo anterior.

[1]No contexto de rastreamento de objetos, o *ground truth* é o conjunto de dados que indica a posição correta do alvo em uma seqüência de vídeo, normalmente determinado pelo ponto central da máscara sobre o alvo.

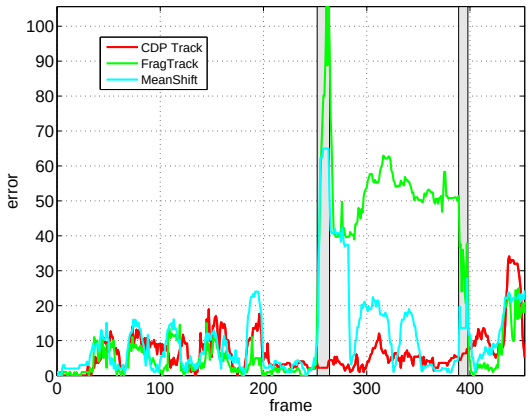


(a) Mulher

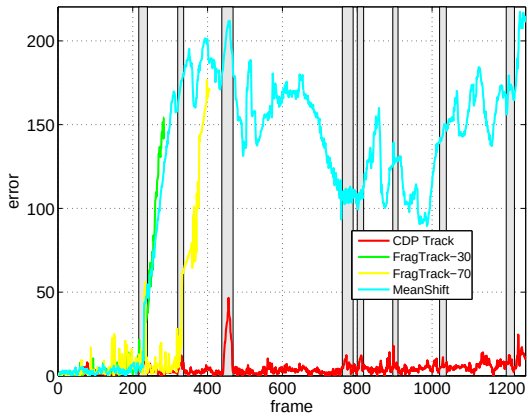
(b) Caviar



(c) Face 1



(d) Face 2



(e) Homem

Figura 4.1 – Erro de Rastreamento das técnicas analisadas (*Meanshift*, *FragTrack* e CPD) para as cinco seqüências de vídeo. Os retângulos cinzas indicam os trechos das seqüências onde ocorre a oclusão total dos alvos.

Os gráficos da Figura 4.1 descrevem os erros de rastreamento obtidos com os três algoritmos para todas as seqüências de vídeo. Como pode ser observado, o algoritmo *Meanshift* é o que apresenta os maiores erros na maioria dos quadros. Isso ocorre devido ao problema ocasionado pelas oclusões parciais e totais, já que esse algoritmo não possui uma abordagem específica para o tratamento desse problema. Sobre as oclusões parciais, que ocorrem freqüentemente nas seqüências *Mulher* e *Caviar*, os algoritmos *FragTrack* e CPD apresentam erros similares. Mas durante as oclusões parciais mais significativas (como as que ocorrem na parte final do vídeo *Caviar*), ou as oclusões totais (marcadas com os retângulos cinzas nas seqüências *Face1*, *Face2* e *Homem*), o algoritmo *FragTrack* apresenta um aumento dos erros de distância, conseguindo recuperar o alvo inicial somente se ele se encontra dentro da região de busca do algoritmo, após o término da oclusão.

O *FragTrack* perde o alvo durante o rastreamento devido a oclusão total na seqüência *Face1*, entre os quadros 340 e 350 (Figura 4.1(c)). Ele também perde o alvo na seqüência *Face2*, quando ocorre a primeira oclusão total, aproximadamente no quadro 255 (Figura 4.1(d)) e perde o alvo na seqüência *Homem*, quando também ocorre a primeira oclusão total, aproximadamente no quadro 230(Figura 4.1(e)). Para essas três seqüências foi aumentada a região de busca do algoritmo *FragTrack* para 70×70 pixels, a fim de avaliar o processo de recuperação do alvo pelo algoritmo. Na seqüência de vídeo *Face1* o alvo foi efetivamente recuperado após a oclusão com o aumento da região de busca, mas o algoritmo voltou a falhar quando ocorreu a oclusão total seguinte (próximo ao quadro 330).

Na verdade, o código do *FragTrack* (disponibilizado pelos autores) parou sua execução durante a seqüência de vídeo *Homem* após a máscara de busca atingir os limites da imagem, devido à perda do alvo durante a oclusão. Isto ocorreu com a região de busca 30×30, no quadro 282, e com a região de busca 70×70 no quadro 404. O mesmo problema ocorreu após a primeira oclusão da seqüência *Face2*. Visto que não ocorre uma melhora quando a região de busca é estendida, os gráficos de erros com essa configuração não foram incluídas na Figura 4.1(e).

Um resumo dos resultados pode ser visto na Tabela 4.1, onde são mostrados: a média dos erros, o erro máximo em cada seqüência, o desvio padrão, e o tempo médio de execução (em segundos por quadro). Como pode ser observado, o *Meanshift* apresenta os maiores erros, seguido pelo *FragTrack* e pelo CPD. É interessante notar

que ambos, *FragTrack* e CPD, possuem resultados similares em termos de precisão, quando ocorrem apenas oclusões parciais (vídeos *Mulher* e *Caviar*). Entretando, o algoritmo proposto (CPD) possui uma performance superior ao *FragTrack* quando ocorrem oclusões totais (seqüências *Face1*, *Face2* e *Homem*).

Em termos de tempo de execução, CPD é mais rápido que o *FragTrack*, mais de cinqüenta vezes no melhor caso e mais de vinte vezes no pior caso, considerando a mesma região de busca (30×30). Essa diferença é maior ainda quando o CPD é comparado com *FragTrack* com a região de busca estendida (70×70). O *Meanshift* é o que apresenta a execução mais rápida de todas, devido a forma de busca iterativa que este algoritmo executa em vez de uma busca exaustiva. Contudo seu desempenho quanto à precisão e robustez durante as oclusões parciais e totais é pior que os algoritmos *FragTrack* e CPD.

Tabela 4.1 – Erro do rastreamento (em pixels) e tempo de execução médio (em segundos por quadro) para os métodos de rastreamento CPD, *FragTrack* e *MeanShift*. Os menores valores para cada seqüência de vídeo são mostrados em negrito.

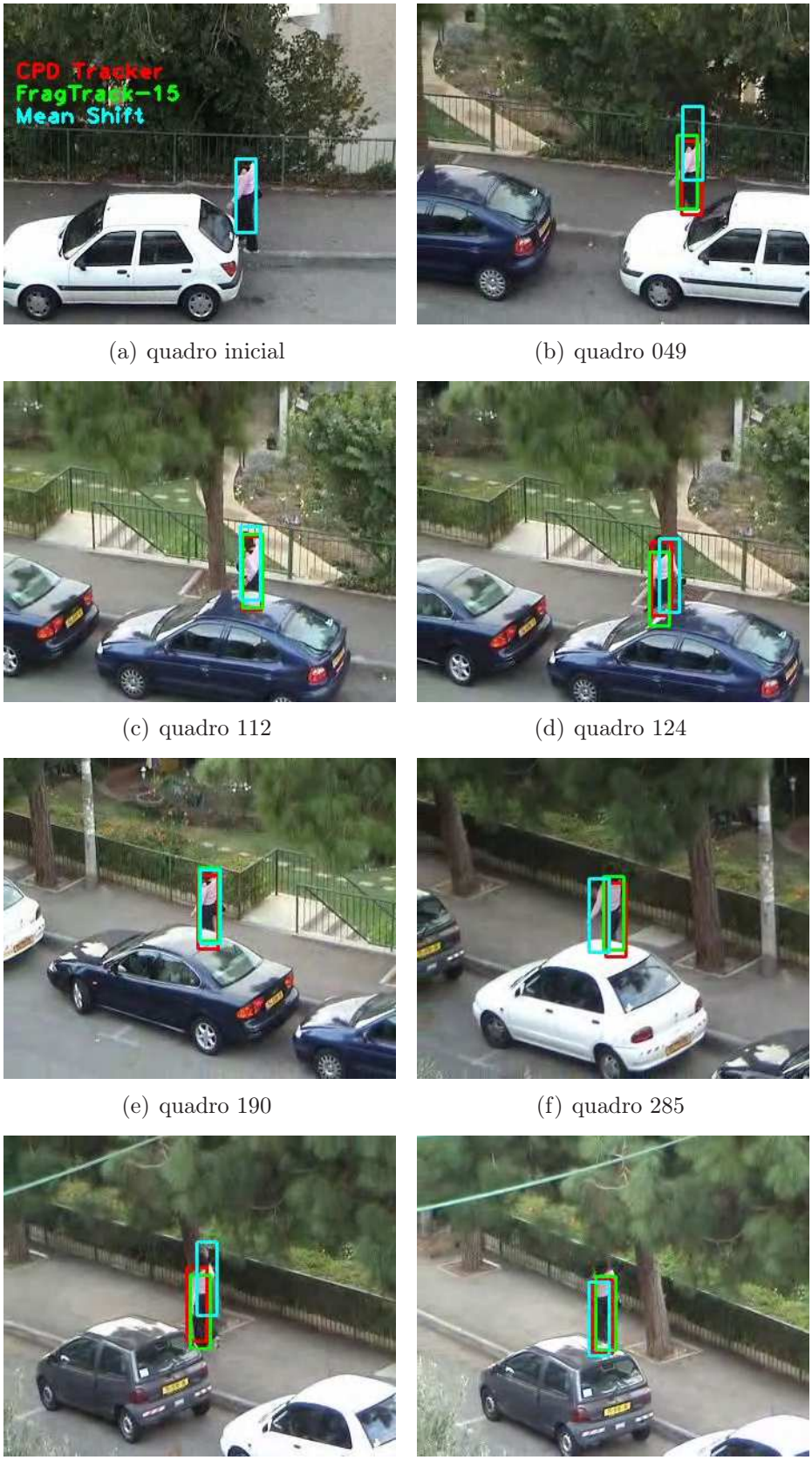
		CPD Tracker	FragTrack-30	MeanShift	FragTrack-70
Mulher	Média dos Erros	6.03	**5.23**	7.20	–
	Maior Erro	**15.86**	17.57	23.28	–
	Desvio Padrão dos Erros	3.65	**3.09**	4.75	–
	Tempo Médio	0.055	2.892	**0.036**	–
Caviar	Média dos Erros	**5.30**	7.33	10.05	–
	Maior Erro	**12.04**	71.18	19.31	–
	Desvio Padrão dos Erros	**2.60**	10.38	4.56	–
	Tempo Médio	0.050	2.323	**0.034**	–
Face1	Média dos Erros	**6.44**	33.19	15.13	10.15
	Maior Erro	**25.24**	132.02	85.15	41.34
	Desvio Padrão dos Erros	**4.90**	45.35	16.48	9.35
	Tempo Médio	0.121	2.648	**0.032**	14.68
Face2	Média dos Erros	**6.49**	19.94	10.95	–
	Maior Erro	**34.13**	105.60	65.00	–
	Desvio Padrão dos Erros	**5.96**	23.75	12.06	–
	Tempo Médio	0.061	2.819	**0.031**	–
Homem	Média dos Erros	**4.50**	20.04	120.65	24.53
	Maior Erro	**46.57**	154.26	217.37	176.65
	Desvio Padrão dos Erros	**4.38**	38.63	63.34	43.60
	Tempo Médio	0.049	2.204	**0.031**	7.53

As Figuras 4.2-4.6 ilustram os resultados do rastreamento dos três algoritmos analisados. Elas mostram alguns quadros das cinco seqüências de vídeo com a máscara de rastreamento de cores diferentes para cada técnica. No primeiro quadro de cada seqüência, todas as técnicas são inicializadas na mesma posição, e desta forma somente uma máscara é visível. No quadro 49 da seqüência *Mulher* (Figura 4.2(b)), quando parte do seu corpo é encoberta por um carro, a máscara da técnica *Meanshift* (representada pelo retângulo azul) é deslocada do alvo, mostrando

a deficiência deste algoritmo em tratar as oclusões. Por outro lado, tanto o *FragTrack* (representado pelo retângulo verde) como o CPD (representado pelo retângulo vermelho), são capazes de rastrear a mulher de uma forma precisa. O mesmo ocorre em situações de oclusão parcial durante o vídeo *Caviar*. Contudo, no final desta seqüência (ver quadro 329 da Figura 4.3(g)), quando a oclusão parcial é substancial, o *FragTrack*(verde) perde-se do alvo enquanto que o CPD (vermelho) se mantém correto.

As outras três seqüências apresentam situações de oclusão total e, como pode ser observado, o algoritmo CPD é capaz de estimar a posição do alvo sobre estas oclusões, enquanto o *Meanshift* (azul) e o *FragTrack* (verde) procuram erradamente o alvo na região de fundo da cena (por exemplo, veja o quadro 360 da seqüência *Face1*, Figura 4.4(h)). Na seqüência *Face2* ocorre um problema semelhante, mas neste caso a oclusão total é ainda mais duradoura, como pode ser verificado nos quadros 253, 258 e 263 (Figuras 4.5(e), 4.5(f) e 4.5(g)), e assim mesmo o CPD (vermelho) manteve uma distância satisfatória do alvo correto, influenciado pelo vetor de predição que atua naquele determinado momento. Na última seqüência (*Homem*), o problema da oclusão total ocorre nos quadros 229, 324 e 461 (Figuras 4.6(b), 4.6(d) e 4.6(g)).

Em alguns casos (como nas seqüências *Face1*, *Face2* e *Homem*), o *Meanshift* (azul) e/ou *FragTrack* (verde) não conseguem recuperar o alvo após a oclusão total, e a máscara perambula pela imagem. Como o *Meanshift* (azul) é baseado em um processo iterativo de busca, pode ocorrer que ele não encontre mais o alvo, caso o alvo não apareça próximo da posição corrente da máscara. A capacidade de recuperação do alvo no algoritmo *FragTrack* (verde) é altamente dependente da sua região de busca, mas o custo computacional aumenta de acordo com o aumento da região de busca. O CPD (vermelho) também apresenta a mesma desvantagem, mas como ele prevê o movimento durante as oclusões totais, o alvo encontra-se normalmente dentro da região de busca após a oclusão.



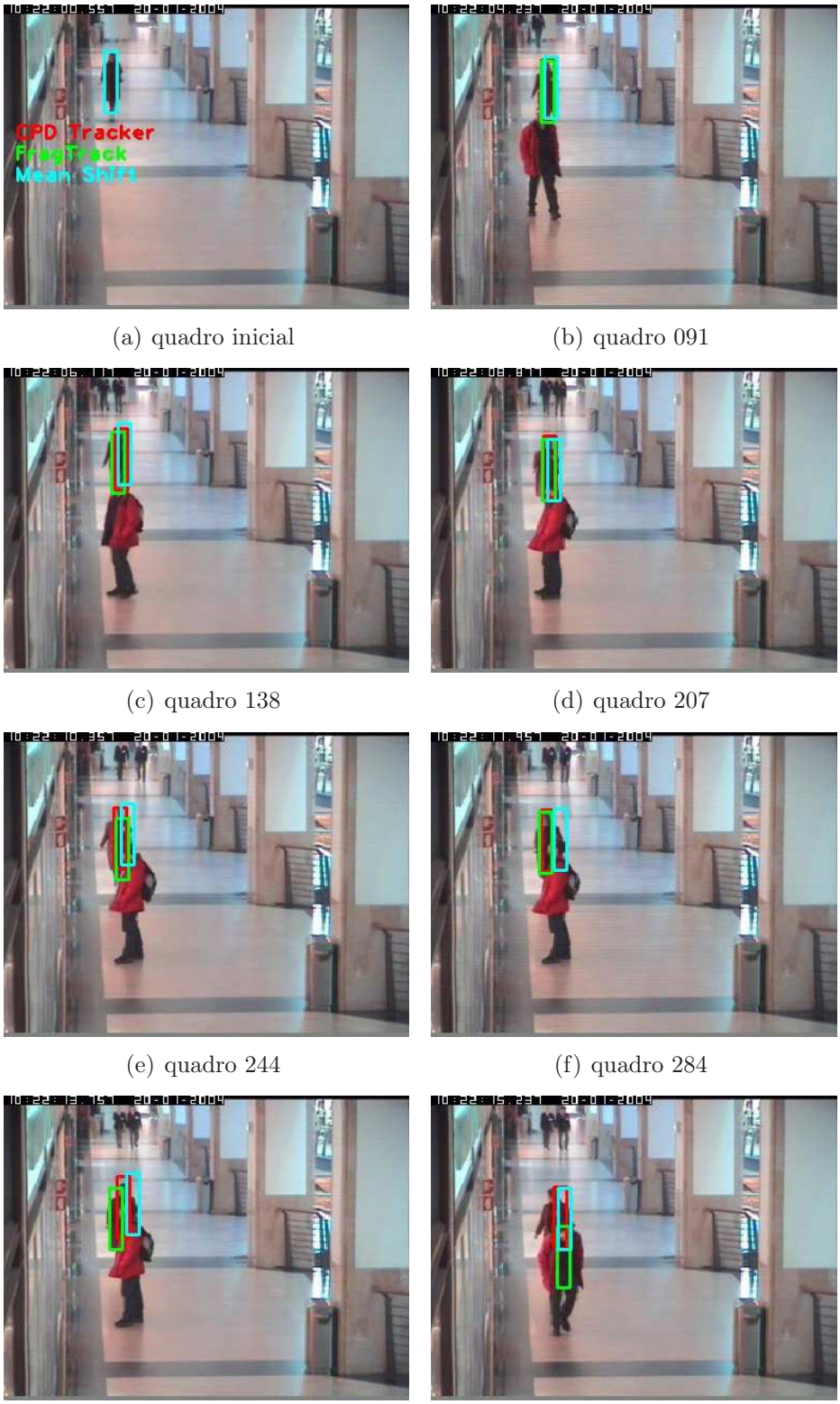
(a) quadro inicial (b) quadro 049

(c) quadro 112 (d) quadro 124

(e) quadro 190 (f) quadro 285

(g) quadro 354 (h) quadro 380

Figura 4.2 – Exemplo de quadros do rastreamento de uma pessoa na seqüência de vídeo *Mulher*.



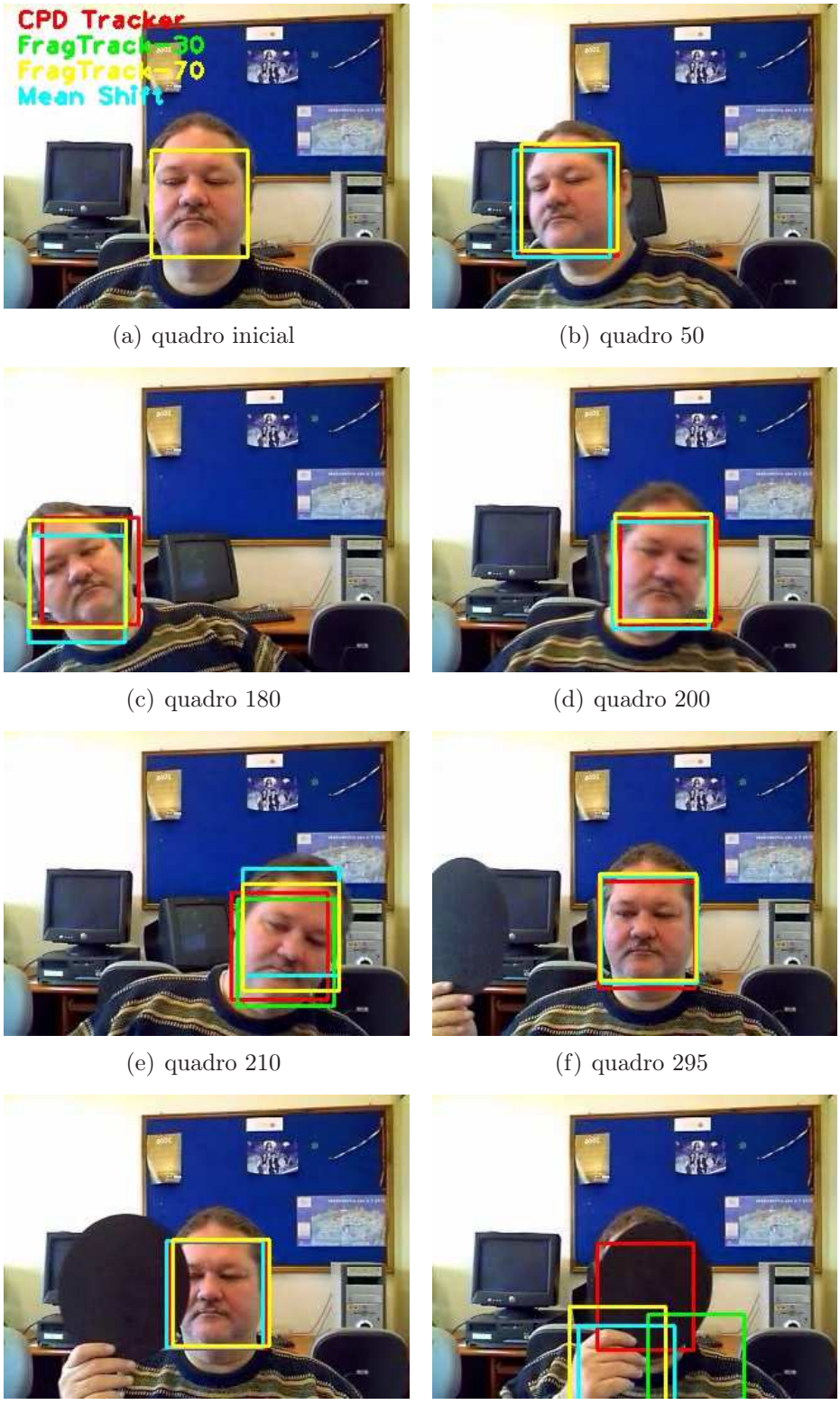
(a) quadro inicial (b) quadro 091

(c) quadro 138 (d) quadro 207

(e) quadro 244 (f) quadro 284

(g) quadro 329 (h) quadro 366

Figura 4.3 – Exemplo de quadros do rastreamento de uma pessoa na seqüência de vídeo *CAVIAR*.



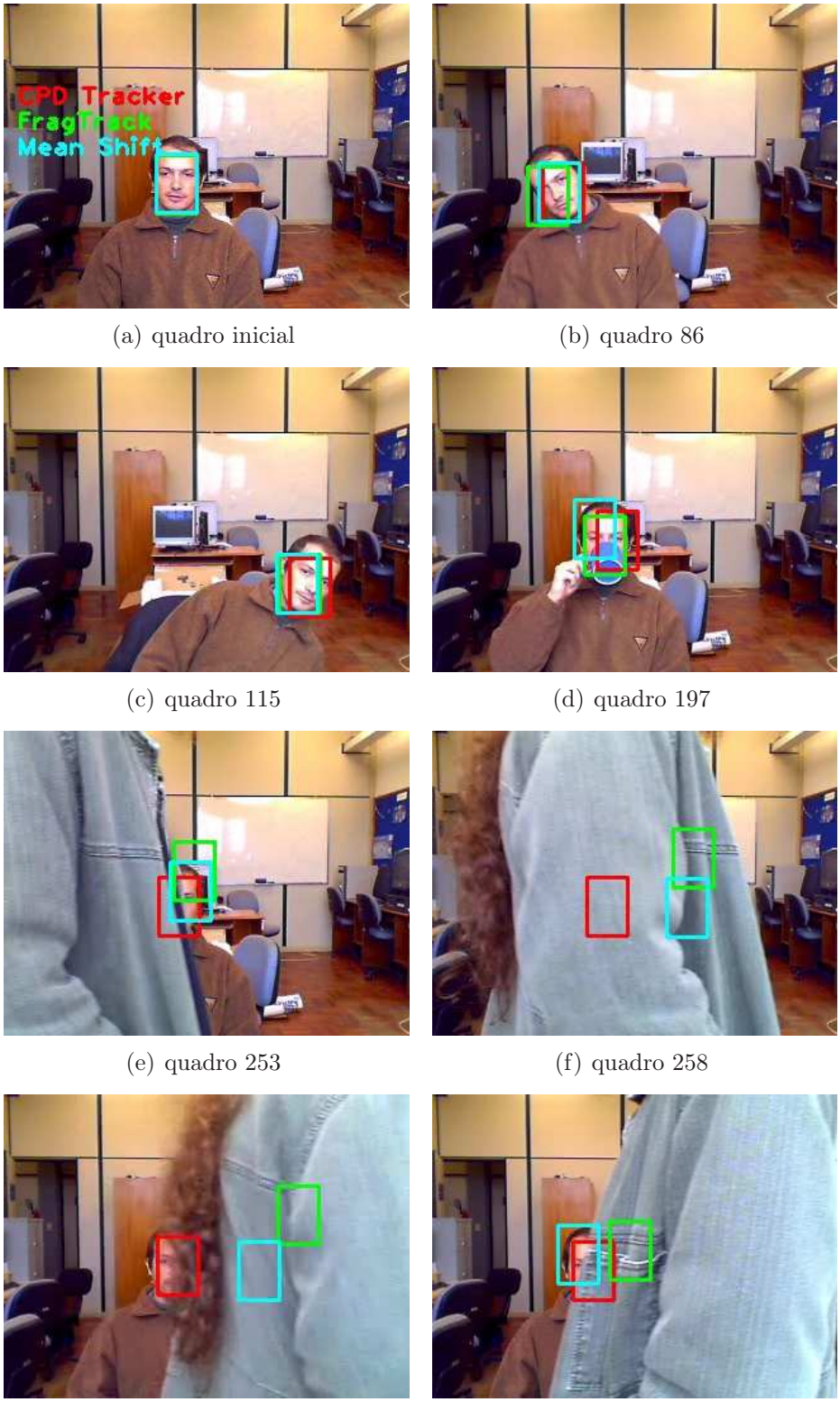
(a) quadro inicial (b) quadro 50

(c) quadro 180 (d) quadro 200

(e) quadro 210 (f) quadro 295

(g) quadro 320 (h) quadro 360

Figura 4.4 – Exemplo de quadros do rastreamento da face na seqüência de vídeo *Face1*. (O algoritmo apresenta robustez na oclusão total com o alvo parado.)



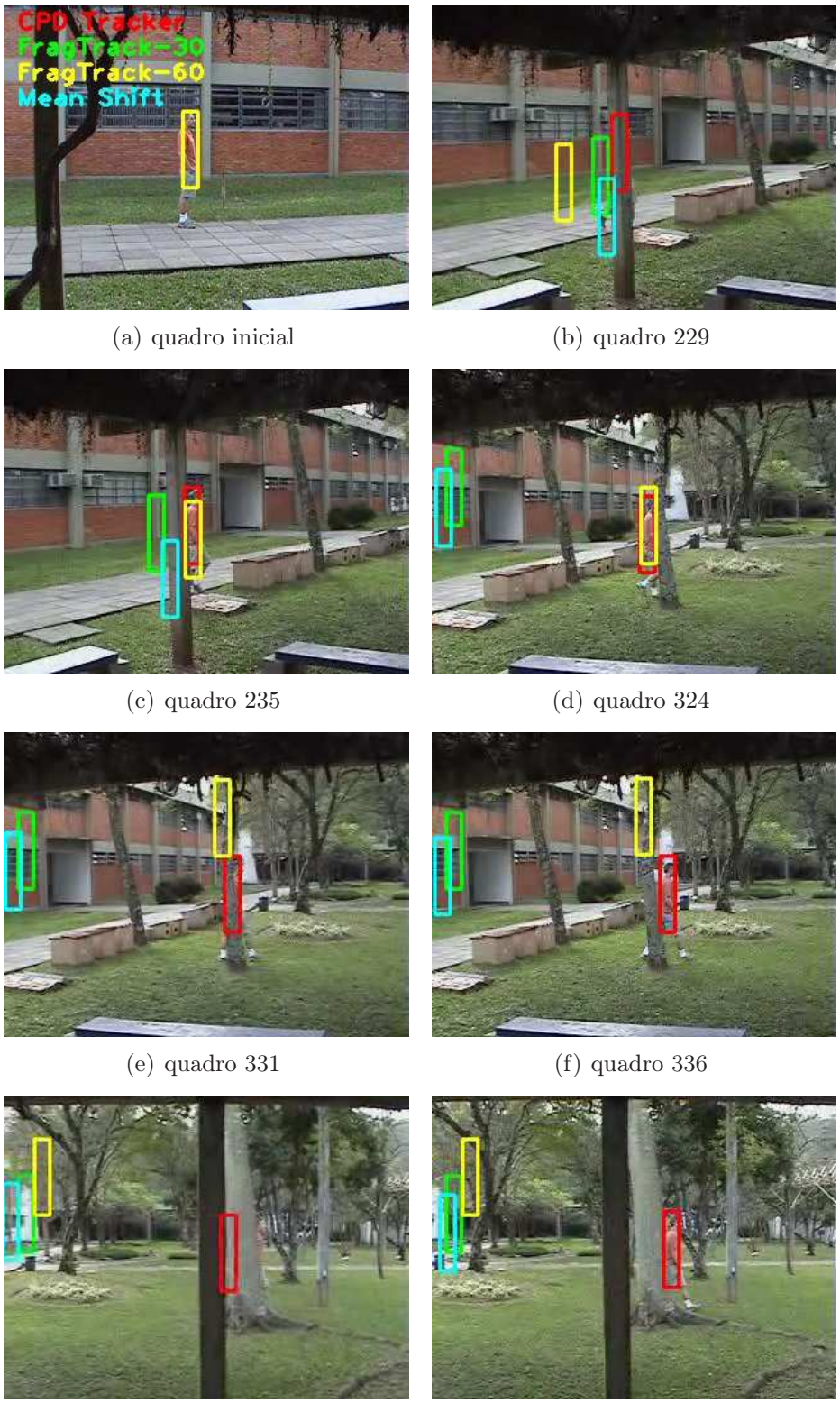
(a) quadro inicial (b) quadro 86

(c) quadro 115 (d) quadro 197

(e) quadro 253 (f) quadro 258

(g) quadro 263 (h) quadro 390

Figura 4.5 – Exemplo de quadros do rastreamento da face na seqüência de vídeo *Face2*. (Mesmo com as oclusões totais o algoritmo CPD mostrou robustez.)



(a) quadro inicial (b) quadro 229

(c) quadro 235 (d) quadro 324

(e) quadro 331 (f) quadro 336

(g) quadro 461 (h) quadro 466

Figura 4.6 – Exemplo de quadros do rastreamento de uma pessoa na seqüência de vídeo *Homem*.

4.2 Abordagem empregada com Detector de Faces

Esta seção apresenta uma abordagem da solução proposta visando o problema específico do rastreamento de faces. Dessa forma, para obter o alvo no quadro inicial foi utilizado um detector de faces automático. O detector é baseado no trabalho de Viola e Jones [28], onde o processo de reconhecimento é baseado na detecção de características que codificam algumas informações sobre a classe a ser detectada (no caso, faces em poses aproximadamente frontais). Mais especificamente, foram utilizadas características *Haar-like* (o nome é originado da semelhança das características com a Transformada de Haar), que codificam a existência de contrastes entre as regiões retangulares adjacentes na imagem. Um conjunto dessas características é usado para codificar os contrastes exibidos por diferentes características faciais humanas e suas vizinhanças. O detector de face emprega o algoritmo AdaBoost [31] para construir um classificador em cascata que seleciona um pequeno conjunto das características *Harr-like*, visando robustez na detecção sem um custo computacional muito grande.

A partir da detecção da face, o algoritmo CPD pode calcular o tamanho e a posição da máscara inicial e, conseqüentemente, os fragmentos conforme a seção 3.2. As Figuras 4.7(a)-4.7(h) ilustram a detecção automática de faces e a obtenção inicial de seus respectivos fragmentos para diferentes pessoas em escalas distintas.

A fim de verificar quais características que poderiam apresentar maior dissimilaridade entre região rastreada (no caso, faces humanas) e a região do fundo onde está o alvo, foram realizados experimentos utilizando o rastreador CPD com o detector de faces automático e configurado com diferentes atributos para caracterizar os fragmentos. Em particular, foram analisadas diferentes combinações entre canais de cor dos espaços RGB e YC_bC_r.

Os resultados apresentados não foram conclusivos na escolha de uma determinada característica, e variaram muito nos vídeos testados, principalmente quando estes apresentaram variações referentes à iluminação, à cor de pele e ao fundo da cena. Durante os testes, as características RGB foram as que apresentaram os melhores resultados comparados com os dados de *ground truth* de cada vídeo. Outras características baseadas em cor de pele podem ser empregadas, como as desenvolvidas nos trabalhos [32, 33]. Evitando fugir do escopo estabelecido

(a) detecção inicial (b) obtenção dos fragmentos

(c) detecção inicial (d) obtenção dos fragmentos

(e) detecção inicial (f) obtenção dos fragmentos

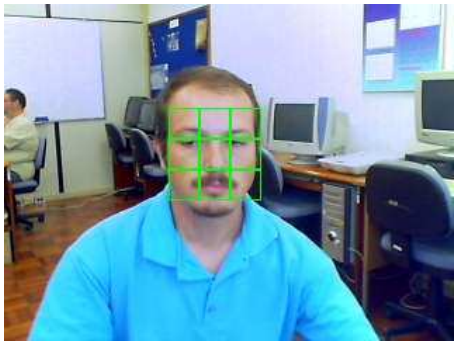
(g) detecção inicial (h) obtenção dos fragmentos

Figura 4.7 – Exemplo do rastreamento usando um detector de faces automático.

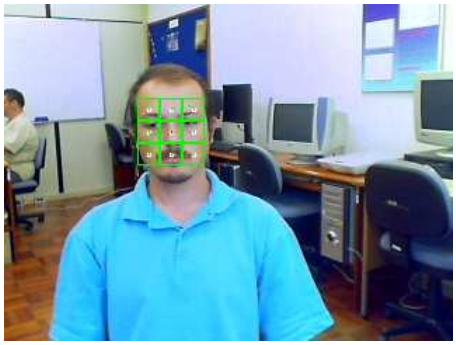
inicialmente para este trabalho, um estudo mais aprofundado sobre a escolha de características para o rastreamento de faces fica em aberto, devendo ser o objeto de

trabalhos futuros.

O algoritmo CPD proposto neste trabalho não lida com mudanças de escala (aumento ou diminuição do tamanho do alvo) para o rastreamento de objetos em geral. Entretanto, a mudança de escala pode ser implementada para o problema específico do rastreamento de faces: quando a pessoa se aproxima da câmera, o tamanho da face na imagem capturada aumenta, e o contrário ocorre, quando ela se afasta. Desta forma, a aplicação do detector de faces em intervalos de tempo pode ser utilizada para detectar mudanças de escala do alvo e, assim, redimensionar a máscara que o descreve de tempos em tempos (a detecção da face não é realizada em todos os quadros para não tornar o sistema muito lento com o uso excessivo do detector). De fato, a detecção da face é realizada somente em uma sub-região da imagem (em uma área retangular em torno da posição atual do rastreador) para reduzir o custo computacional do procedimento. A partir do tamanho da face obtida na nova detecção, é possível aumentar ou diminuir os fragmentos da máscara, conforme ilustrado na Figura 4.8.

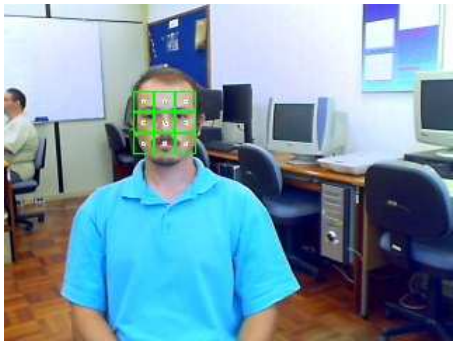


(a) quadro 001

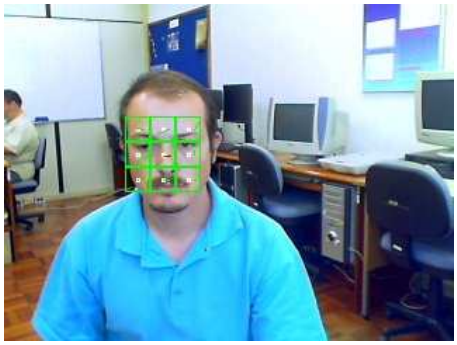


(b) quadro 056

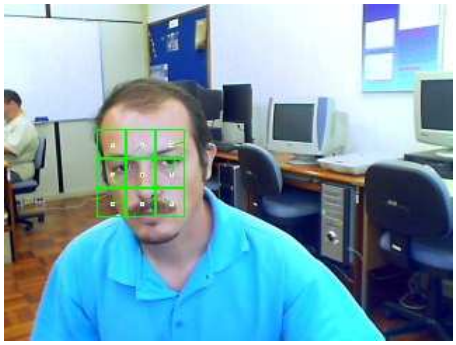
(c) quadro 100



(d) quadro 126



(e) quadro 198



(f) quadro 261

Figura 4.8 – Tratamento da escala da face usando o detector de faces automático.

Capítulo 5

Discussão

Este capítulo apresenta algumas discussões sobre o modelo proposto, analisando aspectos como a seleção dos atributos para o rastreamento, o número de fragmentos, o custo computacional e possíveis limitações da técnica proposta.

5.1 Seleção de Vetores de Características

Na solução proposta, é possível verificar que várias combinações de vetores de características podem ser utilizadas para a construção do modelo de cada fragmento. Estas características podem ser a cor (empregando diferentes espaços de cores), o gradiente, informação de textura, informação de profundidade (se câmeras estéreo forem utilizadas), informações termais (se câmeras termais forem utilizadas), entre outras. Claramente, verificou-se que as melhores características são aquelas que apresentam a melhor dissimilaridade entre o alvo a ser rastreado e o fundo da cena, em particular em torno do objeto, ou que discrimine cada fragmento de maneira única. Salienta-se que nem sempre a escolha de mais características trará um resultado melhor na medida de distância entre o modelo e a busca realizada, principalmente quando a característica adicionada for similar ao fundo da cena. Além disso, o custo computacional do modelo cresce à medida que a dimensão d do vetor de atributos aumenta.

Durante os testes de execução do modelo proposto, foi empregada uma variedade de combinações de características envolvendo diferentes espaços de cores (tais como HSV e RGB normalizada), mas observou-se que as características RGB juntamente com o gradiente são as que apresentam bons resultados na maioria dos

casos. Na verdade, o uso de características *RGB* é normalmente suficiente para alcançar bons resultados, e a inclusão da informação do gradiente tende a melhorar os resultados quando ocorrem as oclusões. Na Tabela 5.1 são apresentados os resultados do modelo proposto usando somente as características *RGB*, nas cinco seqüências de vídeo apresentadas no capítulo anterior. Como pode ser observado, os erros tiveram (na média) um pequeno aumento em relação aos erros apresentados com o modelo empregando as características *RGB* mais o gradiente. Por outro lado, os tempos de execução diminuíram consideravelmente, como era esperado com a redução do número de atributos.

Tabela 5.1 – Erro do rastreamento (em pixels) e tempo de execução (em quadros por segundo) para o Rastreador CPD usando somente características *RGB*.

Mulher		*Caviar*		*Face1*		*Face2*		*Homem*	
Média.	Max.	Média.	Max.	Média.	Max.	Média.	Max.	Média.	Max.
5.08	13.89	7.42	19.70	6.31	26.63	11.14	43.10	4.99	47.54
D.P.	Tempo	D.P.	Tempo	D.P.	Tempo	D.P.	Tempo	D.P.	Tempo
3.09	0.029	4.67	0.031	5.19	0.049	7.80	0.031	5.20	0.018

5.2 Número de Fragmentos

Na seleção ideal dos fragmentos, a variação dos atributos dentro de cada fragmento deve ser suave (para uma melhor caracterização do modelo usando uma função densidade de probabilidade Gaussiana), e deve haver uma certa dissimilaridade entre os diferentes fragmentos (para que um fragmento não seja casado com outro em um quadro adjacente). Na prática, fazer a partição dos fragmentos seguindo essas condições não é uma tarefa fácil.

Neste trabalho, o número de fragmentos para cada máscara é obtido automaticamente a partir da região inicial selecionada, através de uma partição da máscara em fragmentos com tamanhos iguais e aproximadamente quadrados. Embora tal partição possa gerar mais de um fragmento com características similares (o que poderia resultar em casamentos errôneos), o WVMF descrito no capítulo 3 fornece um rastreamento robusto quando os fragmentos restantes são casados corretamente.

5.3 Custo Computacional

Para analisar o custo computacional da abordagem proposta, foi considerado que a máscara inicial é dividida em N_p fragmentos, e as estatísticas dos fragmentos são calculadas usando vetores de características de dimensão d. Também foi considerada uma região de busca de dimensões $S_x \times S_y$.

O custo para o cálculo da representação da imagem integral dentro da região de busca é $\mathcal{O}(S_x S_y d^2)$ [20], e a complexidade para obter a média e a covariância para todos N_p fragmentos é $\mathcal{O}(N_p d^2)$. O custo para calcular a distância entre dois fragmentos, usando a distância de *Bhattacharyya* (Equação (3.4)) é $\mathcal{O}(d^3)$[1], e o custo para a comparação exaustiva para todos N_p fragmentos, na região $S_x \times S_y$, é da ordem de $\mathcal{O}(S_x S_y N_p d^3)$. O custo para se obter o deslocamento global da máscara baseado nos fragmentos individuais usando WWMF é $\mathcal{O}(N_p^2)$ e o método de atualização apresenta um custo na ordem de $\mathcal{O}(d^2)$. Assim, o custo total do procedimento proposto é $\mathcal{O}(S_x S_y d^2 + N_p d^2 + S_x S_y N_p d^3 + N_p^2 + d^2)$. Na prática, $S_x S_y N_p d^3$ é o maior termo. Assim, a complexidade para uma implementação seqüencial pode ser aproximada para $\mathcal{O}(S_x S_y N_p d^3)$. Pode-se salientar também que, como cada fragmento é rastreado individualmente, é possível explorar a redução do tempo de processamento com a utilização de computação paralela usando *hardwares* específicos, tais como as Unidades Programáveis Gráficas (GPUs) ou processadores com mais de um núcleo (*multicore*).

5.4 Limitações

Embora o método proposto tenha um desempenho normalmente satisfatório sob oclusões parciais até oclusões totais (por breve período de tempo), existem algumas situações em que o método está propenso a erros. Por exemplo, durante as oclusões totais por períodos mais longos, a regra de atualização fará com que o método incorpore as novas estatísticas do objeto oclusor, de modo que a máscara possivelmente passe a acompanhar o objeto oclusor em vez do objeto inicial após determinado tempo. Além disso, se o alvo tem sua aparência modificada significativamente durante uma oclusão total (seja de curto período ou longo

[1]Contudo, como mencionado anteriormente, há algoritmos para reduzir a complexidade para $\mathcal{O}(d^{2.376})$ [21].

período), o alvo pode não ser recuperado quando ele reaparece após a oclusão.

Outra questão é a que diz respeito ao movimento do alvo durante a oclusão. Como o modelo do movimento de predição utiliza o deslocamento de vetores dos quadros anteriores para estimar a posição futura do alvo, se o alvo altera a direção de seu movimento durante a oclusão (por exemplo, quando uma pessoa que está se deslocando atrás de uma parede pára ou muda sua direção ou sentido), a posição prevista baseada nos quadros anteriores será errônea.

Também deve ser notado que os fragmentos são rastreados com base em estimativas dos vetores de média e das matrizes de covariância. Se os fragmentos utilizados forem muito pequenos, a estimativa desses parâmetros pode ser muito sensível a mudanças de poucos pixels dos fragmentos, o que pode levar a resultados errados no acompanhamento do alvo.

Alguns exemplos de situações em que o modelo proposto pode falhar estão ilustrados na Figura 5.1. Na Figura 5.1(a), o alvo é muito pequeno, e ainda uma subdivisão em pequenos fragmentos levaria a estimativas dos vetores de médias e matrizes de covariância pouco confiáveis. Na Figura 5.1(b), o alvo é obstruído por um guarda-roupa por um longo tempo (linha tracejada), de modo que o modelo de atualização acaba "aprendendo" as estatísticas do roupeiro e não recupera o alvo após a oclusão. Finalmente, a Figura 5.1(c) mostra um alvo que muda completamente o seu padrão de movimento durante a oclusão, fazendo com que o modelo de previsão seja errôneo e por conseqüência, o alvo seja perdido após a oclusão.

(a) Alvo pequeno

(b) Oclusão por longo tempo

(c) Mudança de movimento

Figura 5.1 – Situações onde o modelo proposto pode apresentar falhas.

Capítulo 6

Conclusões e Trabalhos Futuros

6.1 Conclusões

Neste trabalho foi apresentada uma abordagem para o rastreamento de objetos baseada em fragmentos. A máscara inicial é dividida em sub-regiões retangulares disjuntas (fragmentos), e cada um desses fragmentos é rastreado independentemente, empregando o cálculo da distância de *Bhattacharyya* dentro de uma região de busca. O vetor de deslocamento de cada fragmento é combinado com um vetor de movimento estimado usando uma média vetorial ponderada baseada na mediana (WVMF), obtendo o vetor de deslocamento final. Finalmente, um esquema eficiente de atualização é usado para tratar o problema de mudanças de aparência e iluminação.

Os resultados experimentais indicaram que o método proposto consegue, efetivamente, rastrear objetos em seqüências de vídeo, sendo robusto com respeito a oclusões parciais e totais (por breve período de tempo). O modelo mostrou-se capaz também de adaptar-se às variações de iluminação e às mudanças de aparência dos objetos. Resultados quantitativos indicaram que o desempenho do modelo proposto é comparável e até melhor do que abordagens competitivas [5, 4], apresentando um bom comprometimento entre tempo de execução e acurácia. De fato, o método proposto foi o único que manteve o alvo durante as oclusões totais, e apresentou (na média) o menor erro de rastreamento, computado como sendo a distância Euclideana entre a posição retornada pelo algoritmo e a posição real do alvo. Em termos de tempo de execução, o desempenho da técnica proposta depende basicamente da região de busca e do tamanho do vetor de atributos utilizado para representar os

fragmentos. Usando os valores padrão descritos no capítulo 4, a técnica proposta apresentou tempo médio de execução bem inferior ao *FragTrack*, e superior ao *Meanshift*.

Durante este trabalho foram apresentadas diversas contribuições na área de visão computacional, dentre as quais podemos enfatizar o uso da WVMF para a combinação dos fragmentos da máscara do alvo rastreado e o vetor de movimento previsto. Tal combinação visou tratar o problema das oclusões parciais e totais, que é uma das maiores deficiências dos modelos analisados durante este trabalho. O uso da WVMF permite que fragmentos apresentando movimento inconsistente com a maioria recebam um peso menor na média ponderada. Além disso, a modificação da WVMF proposta neste trabalho incluiu um termo adicional no cálculo do peso, de modo que a qualidade do casamento dos fragmentos (medida pela distância de *Bhattacharyya*) também seja utilizada para penalizar fragmentos com casamento mais fraco. Finalmente, a utilização do vetor de movimento previsto e sua inclusão na WVFM tende a fornecer uma trajetória mais suave para o alvo, além de possibilitar o rastreamento durante algumas oclusões totais de curta duração.

O trabalho desenvolvido nesta dissertação de mestrado foi submetido ao periódico "Image and Vision Computing", sob o título "Robust Adaptive Patch-based Object Tracking using Weighted Vector Median Filters". O artigo foi submetido em agosto de 2008 e está em fase de revisão.

6.2 Trabalhos Futuros

Há várias possibilidades não exploradas que ainda podem trazer melhorias à técnica desenvolvida. Por exemplo, um estudo aprofundado sobre técnicas de seleção automática de características pode melhorar a discriminação entre o alvo e o fundo da cena. De fato, uma abordagem ideal deveria considerar a adaptatividade da seleção de características *on-line* baseadas nas características do objeto e o que está a sua volta em cada quadro do vídeo, similarmente ao trabalho descrito em [34]. Outra possibilidade para trabalhos futuros seria o estudo de técnicas para redução do tamanho do espaço de busca, usando abordagens como as empregadas nos algoritmos de casamento de blocos (e.g. [35]), melhorando a eficiência e o desempenho computacional.

Apêndice

Prova dos resultados mostrados nas Equações (3.16) e (3.17). Sejam $\boldsymbol{\mu}_1$ e $\boldsymbol{C}_1$ a média e a covariância relatadas para os conjuntos S_1 (contendo N amostras), e $\boldsymbol{\mu}_2$ e $\boldsymbol{C}_2$ os mesmos parâmetros para o conjunto S_2 (contendo M amostras). Conseqüentemente, nós temos :

$$\begin{aligned} \boldsymbol{\mu}_1 &= \frac{1}{N}\sum_{\boldsymbol{x}\in S_1}\boldsymbol{x}, && (6.1)\\ \boldsymbol{C}_1 &= \frac{1}{N}\sum_{\boldsymbol{x}\in S_1}(\boldsymbol{x}-\boldsymbol{\mu}_1)(\boldsymbol{x}-\boldsymbol{\mu}_1)^T \\ &= \frac{1}{N}\left(\sum_{\boldsymbol{x}\in S_1}\boldsymbol{x}\boldsymbol{x}^T - \boldsymbol{\mu}_1\sum_{\boldsymbol{x}\in S_1}\boldsymbol{x}^T - \left(\sum_{\boldsymbol{x}\in S_1}\boldsymbol{x}\right)\boldsymbol{\mu}_1^T + \sum_{\boldsymbol{x}\in S_1}\boldsymbol{\mu}_1\boldsymbol{\mu}_1^T\right) && (6.2) \end{aligned}$$

Já que $\sum_{\boldsymbol{x}\in S_1}\boldsymbol{x} = N\boldsymbol{\mu}_1$, nós podemos escrever

$$\boldsymbol{C}_1 = \frac{1}{N}\left(\sum_{\boldsymbol{x}\in S_1}(\boldsymbol{x}\boldsymbol{x}^T) - N\boldsymbol{\mu}_1\boldsymbol{\mu}_1^T\right). \qquad (6.3)$$

Similarmente,

$$\begin{aligned} \boldsymbol{\mu}_2 &= \frac{1}{M}\sum_{\boldsymbol{x}\in S_2}\boldsymbol{x}, && (6.4)\\ \boldsymbol{C}_2 &= \frac{1}{M}\left(\sum_{\boldsymbol{x}\in S_2}(\boldsymbol{x}\boldsymbol{x}^T) - M\boldsymbol{\mu}_2\boldsymbol{\mu}_2^T\right). && (6.5) \end{aligned}$$

A média $\boldsymbol{\mu}$ considerando todas as amostras em $S_1 \cup S_2$ é dada por

$$\boldsymbol{\mu} = \frac{1}{N+M}\sum_{\boldsymbol{x}\in S_1\cup S_2}\boldsymbol{x} = \frac{1}{N+M}\left(\sum_{\boldsymbol{x}\in S_1}\boldsymbol{x} + \sum_{\boldsymbol{x}\in S_2}\boldsymbol{x}\right) = \frac{1}{N+M}\left(N\boldsymbol{\mu}_1 + M\boldsymbol{\mu}_2\right). \qquad (6.6)$$

A matriz de covariância $\boldsymbol{C}$ considerando todas as amostras em $S_1 \cup S_2$ é dada por

$$\begin{aligned}
\boldsymbol{C} &= \frac{1}{M+N} \sum_{\boldsymbol{x} \in S_1 \cup S_2} (\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^T \\
&= \frac{1}{N+M} \left(\sum_{\boldsymbol{x} \in S_1 \cup S_2} (\boldsymbol{x}\boldsymbol{x}^T) - (N+M)\boldsymbol{\mu}\boldsymbol{\mu}^T \right) \\
&= \frac{1}{N+M} \left(\sum_{\boldsymbol{x} \in S_1} (\boldsymbol{x}\boldsymbol{x}^T) + \sum_{\boldsymbol{x} \in S_2} (\boldsymbol{x}\boldsymbol{x}^T) - (N+M)\boldsymbol{\mu}\boldsymbol{\mu}^T \right). \qquad (6.7)
\end{aligned}$$

Das equações (6.3) e (6.5), nós obtemos

$$\sum_{\boldsymbol{x} \in S_1} \boldsymbol{x}\boldsymbol{x}^T = N\boldsymbol{C}_1 + N\boldsymbol{\mu}_1\boldsymbol{\mu}_1^T \quad \text{e} \quad \sum_{\boldsymbol{x} \in S_2} \boldsymbol{x}\boldsymbol{x}^T = M\boldsymbol{C}_1 + M\boldsymbol{\mu}_2\boldsymbol{\mu}_2^T. \qquad (6.8)$$

Finalmente, substituindo a equação (6.8) na equação (6.7) leva ao resultado desejado:

$$\begin{aligned}
\boldsymbol{C} &= \frac{1}{M+N} \left(N\boldsymbol{C}_1 + N\boldsymbol{\mu}_1\boldsymbol{\mu}_1^T + M\boldsymbol{C}_2 + M\boldsymbol{\mu}_2\boldsymbol{\mu}_2^T - (M+N)\boldsymbol{\mu}\boldsymbol{\mu}^T \right) \qquad (6.9) \\
&= (1-w) \left(\boldsymbol{C}_1 + \boldsymbol{\mu}_1\boldsymbol{\mu}_1^T \right) + w \left(\boldsymbol{C}_2 + \boldsymbol{\mu}_2\boldsymbol{\mu}_2^T \right) - \boldsymbol{\mu}\boldsymbol{\mu}^T,
\end{aligned}$$

onde $w = M/(M+N)$ e $1 - w = N/(M+N)$.

Deve ser notado que a Equação (3.3) usa uma estimativa parcial da matriz de covariância. Se estimativas imparciais forem usadas (divisão pelo número de elementos -1), pode ser verificado que as regras de atualização dadas pelas Equações (3.16) e (3.17) mudam para:

$$\boldsymbol{\mu} = \frac{1}{M+N} \left(N\boldsymbol{\mu}_1 + M\boldsymbol{\mu}_2 \right), \qquad (6.10)$$

$$\begin{aligned}
\boldsymbol{C} &= \frac{1}{M+N-1} [(N-1)\boldsymbol{C}_1 + N\boldsymbol{\mu}_1\boldsymbol{\mu}_1^T + (M-1)\boldsymbol{C}_2 + \qquad (6.11) \\
&+ M\boldsymbol{\mu}_2\boldsymbol{\mu}_2^T - (M+N)\boldsymbol{\mu}\boldsymbol{\mu}^T].
\end{aligned}$$

Bibliografia

[1] YILMAZ, A.; JAVED, O.; SHAH, M. Object tracking: A survey. *ACM Computer Surveys*, ACM Press, New York, NY, USA, v. 38, n. 4, 2006.

[2] SHI, J.; TOMASI, C. Good features to track. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*. Seattle: [s.n.], 1994.

[3] LOWE, D. G. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, Kluwer Academic Publishers, Hingham, MA, USA, v. 60, n. 2, p. 91–110, 2004. ISSN 0920-5691.

[4] COMANICIU, D.; RAMESH, V.; MEER, P. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 25, n. 5, p. 564–577, 2003.

[5] ADAM, A.; RIVLIN, E.; SHIMSHONI, I. Robust fragments-based tracking using the integral histogram. In: *Conference on Computer Vision and Pattern Recognition*. Washington, DC, USA: IEEE Computer Society, 2006. p. 798–805.

[6] ISARD, M.; BLAKE, A. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, v. 29, n. 1, p. 5–28, 1998.

[7] BRADSKI, G.; KAEHLER, A. *Learning OpenCV: Computer Vision with the OpenCV Library*. Cambridge, MA: O'Reilly, 2008.

[8] SHEN, C.; BROOKS, M. J.; HENGEL, A. van den. Fast global kernel density mode seeking with application to localisation and tracking. In: *IEEE International Conference on Computer Vision*. Washington, DC, USA: IEEE Computer Society, 2005. p. 1516–1523.

[9] SHEN, C.; BROOKS, M. J.; HENGEL, A. van den. Fast global kernel density mode seeking: applications to localisation and tracking. *IEEE Transactions on Image Processing*, v. 16, n. 5, p. 1457–1469, 2007.

[10] PORIKLI, F. Integral histogram: A fast way to extract histograms in cartesian spaces. In: *IEEE Conference on Computer Vision and Pattern Recognition*. Washington, DC, USA: IEEE Computer Society, 2005. v. 1, p. 829–836.

[11] HAGER, G.; DEWAN, M.; STEWART, C. Multiple kernel tracking with ssd. *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, v. 1, p. I–790–I–797 Vol.1, June-2 July 2004. ISSN 1063-6919.

[12] MARIMON, D.; EBRAHIMI, T. Orientation histogram-based matching for Region Tracking. In: *Eighth International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS 2007)*. [S.l.: s.n.], 2007.

[13] PORIKLI, F.; TUZEL, O.; MEER, P. Covariance tracking using model update based on lie algebra. In: *IEEE Conference on Computer Vision and Pattern Recognition*. Washington, DC, USA: IEEE Computer Society, 2006. p. 728–735.

[14] AVIDAN, S. Ensemble tracking. In: . [s.n.], 2005. Disponível em: <citeseer.ist.psu.edu/avidan05ensemble.html>.

[15] AVIDAN, S. Ensemble tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 29, n. 2, p. 261–271, 2007.

[16] YILMAZ, A.; LI, X.; SHAH, M. Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Washington, DC, USA, v. 26, n. 11, p. 1531–1536, 2004.

[17] SATO, K.; AGGARWAL, J. Temporal spatio-velocity transform and its application to tracking and interaction. In: . New York, NY, USA: Elsevier Science Inc, 2004. v. 96, p. 100–128.

[18] OTCBVS Benchmark Dataset Collection. 2007. Disponível em: <http://www.cse.ohio-state.edu/OTCBVS-BENCH/bench.html>.

[19] RUBNER, Y. et al. Empirical evaluation of dissimilarity measures for color and texture. *Computer Vision and Image Understanding*, Elsevier Science Inc., New York, NY, USA, v. 84, n. 1, p. 25–43, 2001. ISSN 1077-3142.

[20] TUZEL, O.; PORIKLI, F.; MEER, P. Region covariance: A fast descriptor for detection and classification. In: *European Conference on Computer Vision*. [S.l.: s.n.], 2006. v. 2, p. 589–600.

[21] COPPERSMITH, D.; WINOGRAD, S. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computing*, v. 9, n. 3, p. 251–280, 1990.

[22] GAO, J.; KOSAKA, A.; KAK, A. C. A multi-kalman filtering approach for video tracking of human-delineated objects in cluttered environments. *Computer Vision and Image Understanding*, Elsevier Science Inc., New York, NY, USA, v. 99, n. 1, p. 1–57, 2005. ISSN 1077-3142.

[23] 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.

[24] ASTOLA, J.; HAAVISTO, P.; NEUVOS, Y. Vector median filters. *Proceedings of the IEEE*, v. 78, p. 678–689, 1990.

[25] CREE, M. Observations on adaptive vector filters for noise reduction in color images. *Signal Processing Letters, IEEE*, v. 11, n. 2, p. 140–143, Feb. 2004. ISSN 1558-2361.

[26] DUDA, R. O.; HART, P. E.; STORK, D. G. *Pattern Classification*. [S.l.]: Wiley-Interscience Publication, 2000.

[27] LERDSUDWICHAI, C.; MOTTALEB, M. A.; ANSARI, A. Tracking multiple people with recovery from partial and total occlusion. *Pattern Recognition*, v. 38, n. 7, p. 1059–1070, July 2005.

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

[29] ADAM, A. *Fragtrack - Robust Fragments-based Tracking using the Integral Histogram.* 2004. Disponível em: <http://www.cs.technion.ac.il/ amita/fragtrack/fragtrack.htm>.

[30] FISHER, R. *CAVIAR: Context Aware Vision using Image-based Active Recognition.* 2002. Disponível em: <http://homepages.inf.ed.ac.uk/rbf/CAVIAR/>.

[31] FREUND, Y.; SCHAPIRE, R. E. 1997, a decision-theoretic generalization of on-line learning and an application to boosting. In: *European Conference on Computational Learning Theory.* [s.n.], 1995. p. 23–37. Disponível em: <http://citeseer.ist.psu.edu/freund95decisiontheoretic.html>.

[32] HSU, R.-L.; ABDEL-MOTTALEB, M.; JAIN, A. Face detection in color images. *Image Processing, 2001. Proceedings. 2001 International Conference on*, v. 1, p. 1046–1049 vol.1, 2001.

[33] PAI, Y.-T. et al. A simple and accurate color face detection algorithm in complex background. *Multimedia and Expo, 2006 IEEE International Conference on*, p. 1545–1548, July 2006.

[34] COLLINS, R. T.; LIU, Y.; LEORDEANU, M. Online selection of discriminative tracking features. *IEEE Transactions on Pattern Analysis and Machince Intelligence*, IEEE Computer Society, Washington, DC, USA, v. 27, n. 10, p. 1631–1643, 2005.

[35] CHAU, L.-P.; ZHU, C. A fast octagon-based search algorithm for motion estimation. *Signal Processing*, v. 83, n. 3, p. 671–675, 2003.