Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Coordenação de Pós-Graduação em Informática

Segmentação, Rastreamento de Objetos e Detecção de Eventos Primitivos com Aplicação no Monitoramento Automático de Ações Humanas em Vídeo

Bruno Alexandre Dias da Costa

Campina Grande
Setembro de 2008

Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Coordenação de Pós-Graduação em Informática

Segmentação, Rastreamento de Objetos e Detecção de Eventos Primitivos com Aplicação no Monitoramento Automático de Ações Humanas em Vídeo

Bruno Alexandre Dias da Costa

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Campina Grande - Campus I como parte dos requisitos necessários para obtenção do grau de Mestre em Informática.

Área de Concentração: Ciência da Computação
Linha de Pesquisa: Modelos Computacionais e Cognitivos

Orientador: Herman Martins Gomes

Campina Grande, Paraíba, Brasil

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL DA UFCG

C837s Costa, Bruno Alexandre Dias da.

2008 Segmentação, rastreamento de objetos e detecção de eventos primitivos com aplicação no monitoramento automático de ações humanas em vídeo/ Bruno Alexandre Dias da Costa. - Campina Grande, 2008.

116f. : il. Col.

Dissertação (Mestrado em Ciência da Computação) - Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática.

Orientador: Prof. Dr. Herman Martins Gomes.

Referências.

1. Visão Computacional 2. Segmentação de Objetos 3. Rastreamento de Objetos 4. Detecção de Eventos. I. Título

CDU - 004.8(043)

Resumo

A análise automática de movimento humano em vídeo digital vem sendo um tema de pesquisa freqüente devido ao grande número de aplicações promissoras, que fazem parte de um novo domínio, comumente denominado de "looking-at-people". Os objetivos gerais desta classe de aplicações são geralmente descobrir onde estão e quem são os humanos, bem como quais são as suas atividades no ambiente. Neste contexto, os sistemas de vigilância automática ganham cada vez mais espaço, devido à grande quantidade de sistemas de monitoramento (manual) instalados em locais públicos, como shopping centers, praças esportivas, aeroportos, estações de trem, vias públicas, dentre outros. Para a concepção de um sistema de vigilância automática podem ser identificados três importantes desafios de pesquisa, os quais foram objeto de estudo nesta dissertação: detecção de movimento, rastreamento e detecção/classificação de eventos, de acordo com os objetivos da aplicação. Neste trabalho, foi realizado um levantamento de técnicas computacionais com objetivo de desenvolver um protótipo para um sistema de detecção automática de abandono e remoção de objetos em ambientes monitorados por câmeras de vigilância. Esse protótipo foi implementado e resultados experimentais promissores foram conseguidos.

Abstract

Human motion analysis of human movement in digital video has been a frequent topic of research in computer vision due to the large number of promising applications, which are part of a new field, commonly referred to as "looking-at-people". The general objectives of this class of applications are usually to discover the location and identity of humans in the scene, as well as what are their activities on the environment. Within this context, video surveillance systems are ever more gaining popularity because of the large amount of (operator assisted) monitoring systems installed in public places such as shopping malls, sports centers, airports, train stations, public roads, among others. For the design of an automatic surveillance system it is possible to identify three major research challenges, which were the object of study in this work: motion detection, tracking and detection or classification of events according to the application. This dissertation includes a review of computational techniques needed to develop a prototype of a system for detecting the removal of unattended objects in environments monitored by surveillance cameras. This prototype has been implemented and promising experimental results were achieved.

Agradecimentos

Primeiramente agradeço a Deus, pela saúde, paciência e coragem proporcionadas durante essa jornada.

Aos meu pais Maria Aldeci e José Fernando que, sem medir esforços, me ensinaram valores essenciais como respeito e honestidade. Agradeço a meu irmão, Fernando Alexandre, a quem tenho como exemplo de disciplina e perseverança. A Amanda Ramos, pelo carinho, paciência e pelas palavras de conforto nos momentos de dificuldade.

Ao professor Herman Martins Gomes pelos ensinamentos, pelo incentivo e pela excelente orientação. A todos os professores e funcionários que fazem o Departamento de Sistemas e Computação da UFCG.

Aos grandes amigos Saulo de Tarso e Fernando Henrique, duas pessoas fantásticas que conheci durante o mestrado. Aos colegas de laboratório Francisco Fabian (Louco Fabian), Bruno de Brito, Francisco de Assis (Xiquinho), Vinícius Porto, Paulo de Tarso e aos colegas de trabalho da UEPB Dannylo Xavier e Thiago Nóbrega por compreenderem minhas faltas ao trabalho quando finalizava a dissertação. Sempre lembrarei de todos vocês.

Sumário

1 Introdução **1**

1.1 Objetivos 3

1.2 Relevância 4

1.3 Estrutura da Dissertação 4

2 Detecção de Movimento **6**

2.1 Segmentação de Movimento 6

2.1.1 Modelos de Subtração de Imagem de Fundo 7

2.1.2 Modelos Estatísticos 9

2.1.3 Modelos de Diferença Temporal 11

2.1.4 Modelos de Fluxo Óptico 12

2.2 Processamento Pós-Segmentação 14

2.2.1 Remoção de Sombras 15

2.2.2 Detecção de Regiões Conectadas 17

2.2.3 Remoção de Ruído 18

2.3 Classificação de Objetos 19

3 Rastreamento **21**

3.1 Rastreamento por Similaridade de Características 23

3.2 Rastreamento Utilizando Filtro de Kalman 26

3.3 Calibração de Câmera 28

3.4 Rastreamento Multi-Câmera 30

4 Detecção de Ações **33**

4.1 Técnicas Tradicionais 34

4.2 Técnicas Baseadas em Simples Heurísticas 36

5 Proposta de um Sistema de Vigilância Automática a partir de Análise de Vídeo 39

5.1 Arquitetura do Sistema Proposto 39

5.2 Detalhes de Implementação 43

6 Experimentos e Resultados 49

6.1 Bases de Dados e Avaliação Quantitativa 49

6.2 Detecção de Movimento 51

6.3 Rastreamento 62

6.4 Detecção de Eventos 72

7 Conclusões e Trabalhos Futuros 84

7.1 Detecção de Movimento 84

7.2 Rastreamento 85

7.3 Detecção de Eventos 86

7.4 Trabalhos Futuros 87

A Modelo Bimodal Adaptativo (*W4*) 89

B Modelo de Mistura de Gaussianas (*GMM*) 93

C Análise *ROC* dos Modelos *GMM* e *W4* 95

D Análise *ROC* baseada em Pixel 100

E Módulo de Rastreamento 103

E.1 Algoritmo de Rastreamento por Similaridade de Características 103

E.2 Algoritmo de Rastreamento com Filtro de Kalman 103

E.3 Possíveis Estados Assumidos por um Objeto Rastreado 103

E.4 Estratégia para Detecção de Oclusão 103

E.5 Rastreamento de Objetos em Oclusão 107

E.6 Estratégia para Detecção de Divisões 107

E.7 Matrizes utilizadas no Filtro de Kalman 109

F Módulo de Detecção de Eventos 110

Lista de Figuras

1.1 Níveis de abstração de um sistema típico de classificação/reconhecimento de ações humanas. 3

2.1 Exemplo de segmentação de movimento. (a) Quadro do vídeo original utilizado como referência para segmentação do movimento. (b) Quadro resultante do processo de segmentação. 7

2.2 Exemplo de segmentação de movimento: (a) Quadro de vídeo no tempo t. (b) Quadro de vídeo no tempo t+1. (c) Diferença temporal. (d) Resultado da segmentação após eliminação de ruído. 13

2.3 Exemplo de problema de segmentação causado por sombras. (a) Imagem original. (b) Imagem segmentada. 14

2.4 Exemplo de remoção de sombras. (a) Imagem original. (b) Imagem segmentada com *foreground* na cor preta e sombras na cor cinza. 17

2.5 Exemplo detecção de regiões conectadas com respectivos rótulos e retângulos mínimos. 18

3.1 Tipos de algoritmos de rastreamento. 22

3.2 Objetos vistos de diferentes pontos do mesmo ambiente: (a) Com oclusão . (b) Sem oclusão. 29

4.1 Representação gráfica de uma Cadeia de Markov Escondida. 34

4.2 Fluxo do processo proposto por Yamato et al. (1992). 35

4.3 Quadros extraídos do conjunto de dados *PETS* 2006 exibindo abandono de bagagem em uma estação de trem. 37

4.4 Exemplo de abandono de bagagem apresentado em Dedeoglu (2004). (a) Imagem original. (b) Destaque para mochila abandonada e regiões *S* e *R* avaliadas para decidir o tipo de objeto detectado (abandonado ou removido). 38

5.1 Modelo conceitual para um sistema de vigilância automática. 40
5.2 Proposta de arquitetura para um sistema de vigilância automática. 41
5.3 Diagrama de componentes com módulo detector de movimento e fonte genérica de imagens. 45
5.4 Diagrama de atividades do módulo detector de movimento. 46
5.5 Diagrama de atividades do módulo rastreador de objetos. 47
5.6 Diagrama de atividades do módulo detector de eventos. 48

6.1 Exemplo de interseção (I) entre um objeto do conjunto verdade, representado por *GT*, e um objeto detectado pelo modelo de segmentação, representado por *DT*. 52
6.2 Exemplos de segmentação fragmentada pelo modelo *W4*. (a) Imagem de entrada. (b) Imagem segmentada. (c) Comparativo entre conjunto verdade (cor preta) e objetos segmentados (cor vermelha). 57
6.3 Exemplo de segmentação com região conjunta. (a) Imagem de entrada. (b) conjunto verdade. (c) Comparativo entre conjunto verdade (cor preta) e imagem segmentada (cor vermelha). 59
6.4 Exemplo de segmentação com ruído. (a) Imagem de entrada. (b) Conjunto verdade. (c) Comparativo entre os objetos do conjunto verdade (cor preta) e objetos da imagem segmentada (cor vermelha). 59
6.5 Gráfico Box-plot com porcentagem (eixo vertical) das métricas coletadas (eixo horizontal) *W4*. 60
6.6 Gráfico Box-plot com porcentagem (eixo vertical) das métricas coletadas (eixo horizontal) para o modelo *GMM*. 61
6.7 Resultados obtidos de imagens sintéticas. (a) Objetos em estado *Novo* na cor verde. (b) Objetos em estado *Rastreando* na cor vermelha. (c) Objetos em estado *Oclusão* em azul. 65
6.8 Resultados obtidos pelo Algoritmo 1 para o Objeto 1. (a) Saída do módulo detector de movimento. (b) Objeto 1 rastreado, em estado *Novo* (verde) e *Rastreando* (vermelho). 67
6.9 Resultados obtidos pelo Algoritmo 1 para o Objeto 2. (a) Saída do módulo detector de movimento. (b) Objeto 2 (identificador 013) rastreado, em estado *Reastreando* (vermelho), *Divisão* (azul claro) e *Junção* (em amarelo). 68

6.10 Resultados obtidos pelo Algoritmo 1 para os Objetos 3 e 4. (a) Saída do módulo detector de movimento. (b) Objeto 3 (identificador 096) e 4 (identificador 97) em estado *Rastreando* (vermelho), *Divisão* (azul claro) e *Junção* (em amarelo). 69

6.11 Resultados obtidos pelo Algoritmo 1 para o Objeto 5. (a) Saída do módulo detector de movimento. (b) Objeto 5 (identificadores 394 e 690) em estado *Rastreando* (vermelho). 70

6.12 Resultados obtidos pelo Algoritmo 1 para o Objeto 6. (a) Saída do módulo detector de movimento. (b) Objeto 5 (identificador 698) em estado *Rastreando* (vermelho). . . 71

6.13 Resultados obtidos pelo Algoritmo 2 para o Objeto 7. (a) Saída do módulo detector de movimento. (b) Objeto 7 (identificador 36) em estado *Oclusão* (em Azul) e posteriormente recuperado em estado *Rastreando* (em vermelho). 73

6.14 Resultados obtidos pelo Algoritmo 2 para o objeto 8. (a) Saída do módulo detector de movimento. (b) Objeto 8 (identificador 227) antes, durante e após recuperação do estado de *Oclusão*. 74

6.15 Resultados obtidos pelo Algoritmo 2 para o objeto 9. (a) Saída do módulo detector de movimento. (b) Objeto 9 (identificador 447) antes, durante e após recuperação do estado de *Oclusão*. 75

6.16 Resultados obtidos pelo Algoritmo 2 para o objeto 10. (a) Saída do módulo detector de movimento. (b) Objeto 10 (identificador 47) antes, durante e após recuperação do estado de *Oclusão*. 76

6.17 Exemplo de quadro com objeto abandonado. Estratégia de classificação utiliza histogramas de duas áreas: a do objeto (identificado pelo retângulo menor) e a de sua vizinhança (identificado pela área compreendida entre os retângulos menor e maior). 78

6.18 Exemplo de histogramas. (a) Histograma da região do objeto da Figura 6.17. (b) histograma da vizinhança do objeto da Figura 6.17. 78

6.19 Exemplo de quadro com objeto abandonado. Estratégia de classificação utiliza histogramas de duas áreas: a do objeto (identificado pelo retângulo menor) e a de sua vizinhança (identificado pela área compreendida entre os retângulos menor e maior). 79

6.20 Exemplo de histogramas. (a) Histograma da região do objeto da Figura 6.19. (b) histograma da vizinhança do objeto da Figura 6.19. 79

6.21 Alarme gerado para abandono de objeto no Vídeo 1. Os objetos envolvidos no alarme são destacados por um retângulo amarelo. A letra *O* indica o dono do objeto. A letra *A* indica o tipo de ação primitiva: abandono (A) ou remoção (R). 80

6.22 Alarme gerado para abandono de objeto no Vídeo 2. Os objetos envolvidos no alarme são destacados por um retângulo amarelo. A letra *O* indica o dono do objeto. A letra *A* indica o tipo de ação primitiva: abandono (A) ou remoção (R). 81
6.23 Exemplos de falsos alarmes gerados para o Vídeo 2. Os objetos envolvidos no alarme são destacados por um retângulo amarelo. A letra *O* indica o dono do objeto. A letra *A* indica o tipo de ação primitiva: abandono (A) ou remoção (R). 82

C.1 Análise *ROC* do modelo *W4*. À esquerda ((a) e (c)), ambiente interno. À direita ((b) e (d)) ambiente externo. 96
C.2 Análise *ROC* do modelo *W4*. Continuação. À esquerda ((e), (g) e (i)), ambiente interno. À direita ambiente externo ((f), (h) e (j)) 97
C.3 Análise ROC do modelo *GMM*. À esquerda ((a), (c) e (e)), ambiente interno. À direita ambiente externo ((b), (d) e (f)). 98
C.4 Análise ROC do modelo *GMM*. Continuação. À esquerda ((a), (c) e (e)), ambiente interno. À direita ambiente externo ((b), (d) e (f)). 99

D.1 Análise *ROC* para o parâmetro *t* do modelo *W4*. 101
D.2 Análise *ROC* para o parâmetro *t* do modelo *W4* com valor 0. 102

E.1 Fluxograma do algoritmo de rastreamento por similaridade de características. 104
E.2 Fluxograma do Algoritmo 2 de rastreamento por similaridade de características com auxílio do Filtro de Kalman. Diferenças em relação ao Algoritmo 1 destacadas em vermelho. 105

F.1 Fluxograma do algoritmo de detecção de eventos utilizado neste trabalho. 111

Lista de Tabelas

3.1 *Características utilizadas por Amer (2005).* 24

3.2 *Características utilizadas por Dedeoglu (2004).* 25

3.3 *diferenças entre características de dois objetos Q e S utilizadas por Humphreys (2004).* 26

6.1 *Possiveis relações definidas por Nascimento and Marques (2006), entre objetos obtidos pelo modelo e objetos do conjunto verdade.* 53

6.2 *Valores testados na análise quantitativa do modelo W4.* 54

6.3 *Valores testados na análise quantitativa do modelo GMM.* 55

6.4 *Comparativo entre os modelos de segmentação testados em ambientes interno.* . . . 58

6.5 *Comparativo entre os modelos de segmentação testados em ambientes externo.* . . . 58

6.6 *Estados de um objeto rastreado.* 64

6.7 *Resultados obtidos pelo Algoritmo 1.* 66

6.8 *Resultados obtidos pelo algoritmo 2.* 72

6.9 *Resultados obtidos pelo módulo de detecção de eventos.* 83

A.1 *Parâmetros do modelo W4.* 92

B.1 *Parâmetros do modelo GMM.* 94

D.1 Definições das métricas utilizadas para análise *ROC*. 101

E.1 *Estados de um objeto rastreado.* 106

Capítulo 1

Introdução

O uso de vídeo digital é algo cada vez mais presente na vida das pessoas, principalmente, devido a fatores técnicos como o barateamento dos equipamentos para geração de vídeo, o aumento na qualidade das imagens geradas, o baixo consumo de energia desses equipamentos e o desenvolvimento dos meios de transmissão (Utsumi et al., 1998). Além de fatores técnicos, fatores sociais como violência e ações criminosas fazem com que cada vez mais sejam utilizados sistemas viglilância através de vídeo com objetivo de evitar agressões, roubos ou atos de vandalismo, bem como obter provas contra tais ações.

Uma consequência imediata do uso de vídeo em larga escala é a grande quantidade de vídeos gerados que, em sua grande maioria, não serão assistidos na íntegra, pois o conteúdo relavante é restrito a alguns poucos segundos desses vídeos. Dessa maneira, a grande quantidade de vídeos gerados leva ao disperdício de recursos de armazenamento. Além disso, o uso de sistemas convencionais de monitoramento através de vídeo não permite (por si só) a intervenção imediata no ambiente (para evitar assaltos, por exemplo).

Diante desse contexto, a análise automática de movimento humano através de vídeo vem sendo um dos principais temas de pesquisa em visão computacional. Segundo Gavrila (1999), este tema faz parte de um novo domínio de aplicações que emergiu através dos últimos anos, chamado *looking at people* (olhando as pessoas, em português) e cujo objetivo geral é descobrir onde estão os humanos, quem são eles e quais as atividades que estão desempenhando no ambiente. Dentre as aplicações promissoras, destacam-se:

- O rastreamento e classificação do comportamento de pessoas através de múltiplas câmeras, permitindo o monitoramento automático de atividades suspeitas em estacionamentos ou lojas de departamentos (Gavrila, 1999);

- A análise de desempenho de atletas pela observação das habilidades dos alunos e atuando com sugestões para melhorar os movimentos (Aggarwal and Cai, 1999; Ribeiro and Santos-Victor, 2005)

- Sistemas de diálogo homem-máquina em que, por exemplo, a interação com usuário pode ser iniciada pela detecção da presença do usuário (Gavrila, 1999);

- Armazenamento e Recuperação de imagens baseadas em conteúdo (Aggarwal and Cai, 1999; Gavrila, 1999)

Em sua discussão sobre a análise de movimento humano, Aggarwal and Cai (1999) dividem o tema em três áreas: análise de movimento, rastreamento de movimento humano (*tracking human motion*) e a detecção de eventos ou classificação/reconhecimento de atividades humanas (*recognizing human activities*). O reconhecimento de ações humanas consiste em identificar o que as pessoas estão fazendo (dentro de um conjunto restrito de possibilidades) através do vídeo registrado por uma ou mais câmeras.

Nesse contexto, os sistemas de vigilância automática fazendo uso de imagens de sistemas de monitoramento convencionais ganham cada vez mais espaço devido à grande disseminação desses sistemas tradicionais, instalados em locais públicos, como shopping centers, praças esportivas, aeroportos e vias públicas (Ribeiro and Santos-Victor, 2005). Um sistema de vigilância automática, geralmente é composto por módulos que envolvem as três áreas citadas por Aggarwal and Cai (1999): detecção de movimento, rastreamento de pessoas, e classificação da ação de acordo com as necessidades da aplicação.

Do ponto de vista de abstração, esses módulos estão relacionados conforme ilustrado na Figura 1.1, sendo o de detecção de movimento o módulo de mais baixo nível, enquanto que o de classificação o de mais alto nível. O objetivo principal desse módulo é realizar a separação entre os objetos que se movimentam na cena (*foreground*) e a parte estática ou fundo da cena (*background*) a partir de uma sequência de quadros de entrada.

Ao módulo de rastreamento cabe a função de estabelecer relações temporais entre os objetos alvo de quadros de vídeo consecutivos, isto é, identificar, no quadro corrente de um vídeo, a localização dos objetos alvo detectados no quadro anterior. Uma vez que os elementos de cena (*não-estáticos*) sejam detectados e rastreados, é de responsabilidade do módulo de classificação de ações atingir os objetivos finais da aplicação, isto é, as deteção de eventos primitivos ou classificação das ações dos objetos que estão na cena.

O grande número de publicações sobre sistemas de vigilância automática em periódicos e confe-

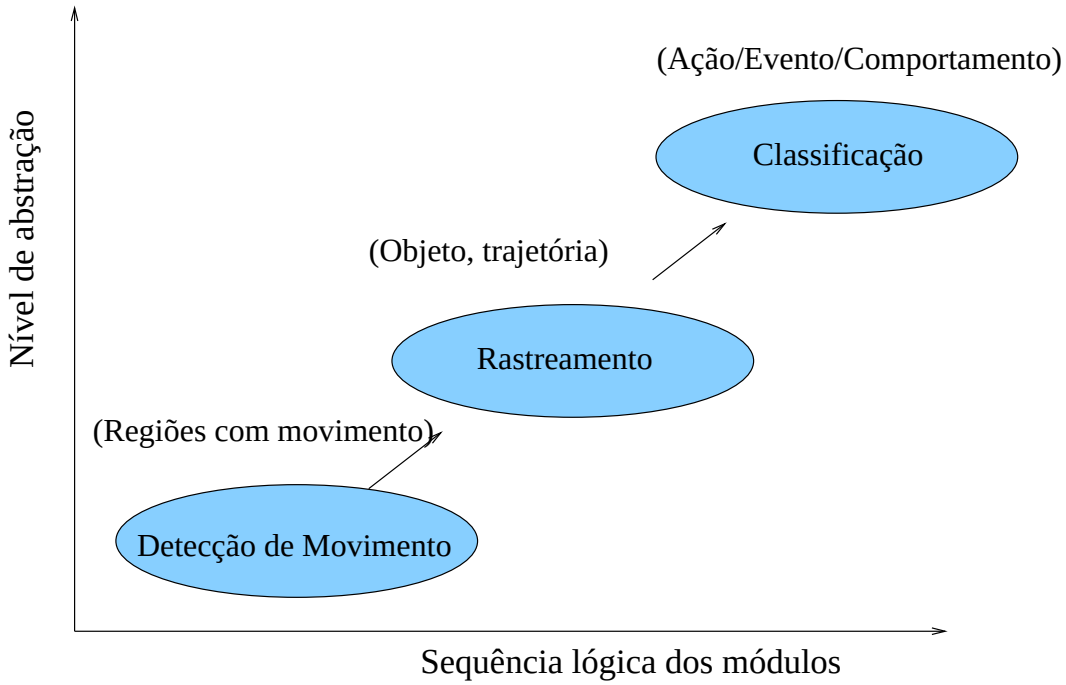


Figura 1.1: Níveis de abstração de um sistema típico de classificação/reconhecimento de ações humanas.

rências internacionais e nacionais, como os periódicos *IEEE Transactions*, a *International Conference on Computer Vision*, o *Brazilian Symposium on Computer Graphics and Image Processing* (Haritaoglu et al., 2000; Amer, 2005; Stauffer and Grimson, 1999; Elgammal et al., 2000; Gavrila, 1999; Nascimento and Marques, 2006), dentre outros, fez com que algumas das aplicações, antes consideradas apenas como promissoras, começassem a se tornar realidade.

Nesta direção, podem-se citar os projetos *VSAM* (Collins et al., 2000), do Departamento de Defesa Americana, O projeto CAVIAR (*Context Aware Vision using Image-based Active Recognition*), da universidade de Edinburgh, no Reino Unido, e o Sistema *W4*, proposto por Haritaoglu et al. (2000).

1.1 Objetivos

O principal objetivo deste trabalho é viabilizar soluções para os problemas de sistemas de monitoramento convencionais citados no início do capítulo, o seja, soluções para armezenar apenas o conteúdo relevante dos vídeos produzidos por esses sistemas e permitir a notificação dos acontecimentos nos ambientes monitorados de maneira automática e em tempo hábil.

A metodologia empregada para alcançar os objetivos deste trabalho compreende uma revisão das técnicas tradicionalmente utilizadas em sistemas de vigilância automática, destacando as principais

dificuldades e tendências no desenvolvimento de cada um dos módulos desses sistemas, assim como o desenvolvimento de um protótipo de sistema que, a partir da combinação de informações de rastreamento de objetos, e heurísticas simples de detecção e classificação de eventos, seja capaz de detectar ações primitivas como a remoção ou abandono de objetos em ambientes monitorados.

1.2 Relevância

Existe uma necessidade crescente por sistemas de vigilância automática de ambientes, devido ao grande número de sistemas de vigilância convencionais instalados. O uso desses sistemas convencionais, por sua vez, é consequência do aumento da violência em grandes cidades e de ações criminosas em geral como assaltos, atos de vandalismo ao patrimônio público, atendados ao pudor, atos de terrorismo, dentre outros.

Ao contrário dos sistemas convencionais de monitoramento, sistenas automáticos devem requerer a atenção humana somente quando necessário, pois o monitoramento manual (realizado por um operador humano) contínuo de fluxo de vídeo além de estar sujeito a falhas, consome tempo valioso.

Neste contexto, a habilidade para detecção de objetos removidos ou abandonados em cena é de grande importância em sistemas de vigilância automática. Detectar objetos abandonados como bagagens em aeroportos é importante, visto que esta atividade pode representar riscos de atentados terroristas prejudicando pessoas inocentes. Por outro lado, proteger objetos contra remoção sem a devida permissão também tem grande importância para sistemas de monitoramento em museus, galerias de arte ou lojas de departamento, dentre outros contextos semelhantes.

1.3 Estrutura da Dissertação

Nesta dissertação, é apresentado um levantamento das principais técnicas utilizadas em cada um dos módulos de um sistema de vigilância automática como aqueles citados no início do capítulo. Posteriormente, é apresentada uma proposta de arquitetura, incluindo os módulos e as tecnologias necessárias para implementação de tais sistemas. Por fim, alguns dos elementos da arquitetura proposta são selecionados com objetivo de implementar um protótipo que valide algumas das técnicas estudadas e outras propostas. Os demais capítulos desta dissertação estão organizados da seguinte maneira:

- Capítulo 2: este capítulo apresenta as atividades e desafios envolvidos na detecção de movimento: segmentação de movimento, remoção de ruído e classificação de objetos. Na primeira subseção, são descritas as principais abordagens utilizadas para segmentação de movimento,

isto é, técnicas para separar regiões estáticas de regiões em movimento a partir de seqüências de quadros de vídeo. Em seguida, são apresentadas técnicas utilizadas para remoção de ruído remanescentes da etapa de segmentação. Ao final do capítulo são apresentadas formas de classificar os objetos segmentados em diferentes classes (pessoas, veículos, animais).

- Capítulo 3: apresenta os principais desafios relacionados à concepção de um módulo de rastreamento. Esse capítulo também caracteriza dois tipos de algoritmos de rastreamento utilizados na literatura: um baseado no casamento de características e outro baseado na predição de movimento utilizando Filtro de Kalman. Na última seção do Capítulo são discutidas mecanismos de rastreamento de objetos através de múltiplas câmeras.

- Capítulo 4: discutem-se algumas das abordagens tradicionais utilizadas no processo de classificação de eventos e ações humanas em vídeo digital. Posteriormente, menciona o uso de heurísticas simples propostas por trabalhos recentes com objetivo de detectar eventos primitivos como abandono e remoção de objetos no ambiente monitorado.

- Capítulo 5: inicialmente (Seção 5.1), é apresentada a proposta de arquitetura distribuída e escalável para um sistema de vigilância automática . Em seguida (Seção 5.2), é feito um detalhamento da implementação dos principais módulos da arquitura proposta.

- Capítulo 6: na Seção 6.1 descrevem-se as bases de dados utilizadas nos experimentos de validação dos módulos implementados. Nas demais seções (6.2, 6.3, 6.4), são apresentados os resultados experimentais obtidos após a implementação dos principais módulos da arquitetura proposta.

- Capítulo 7: discutem-se os resultados apresentados no Capítulo 6 bem como apresenta sugestões para trabalhos futuros.

Capítulo 2

Detecção de Movimento

Sistemas de análise de movimento humano, como os de vigilância automática, possuem em comum objetos alvo (Dedeoglu, 2004), os quais estão em movimento no ambiente observado. Desta forma, tais sistemas necessitam separar e classificar esses objetos alvo para serem processados em etapas subsequentes como rastreamento e detecção de ações. Assim, a primeira das etapas de um sistema de análise de movimento humano é a detecção de movimento.

Normalmente, a detecção de movimento é representada por um módulo (detector de movimento) para o qual são enviadas as imagens capturadas pelos sensores instalados no ambiente. O objetivo principal deste módulo é segmentar (separar) os objetos alvo da aplicação do resto da imagem. Entretanto, algumas etapas auxiliares também são realizadas por este módulo: detecção de sombras e remoção de ruído. Opcionalmente, neste módulo pode ser feita a classificação de diferentes tipos objetos alvo como pessoas, veículos ou animais com objetivo eliminar objetos que não são alvos da aplicação principal. Neste capítulo, serão discutidas técnicas utilizadas tanto na segmentação de movimento (Seção 2.1) como em cada uma dessas etapas auxiliares (Seções 2.2 e 2.3).

2.1 Segmentação de Movimento

A segmentação de movimento é a primeira das subetapas da detecção de movimento. Consiste na separação entre objetos que se movimentam na cena (*foreground*) e a parte estática ou fundo da cena (*background*), a partir de uma sequência de quadros de entrada. A entrada da segmentação de movimento são as sequências de imagens recebidas pelo módulo detector de movimento. Como saída, são produzidas imagens binarizadas. Nestas imagens, destacam-se as regiões (*blobs*) em movimento das regiões estáticas do ambiente. Na Figura 2.1, é apresentado um exemplo de segmentação de

movimento.

(a) (b)

Figura 2.1: Exemplo de segmentação de movimento. (a) Quadro do vídeo original utilizado como referência para segmentação do movimento. (b) Quadro resultante do processo de segmentação.

Para realizar a segmentação de movimento a partir de um quadro de entrada, o módulo detector de movimento normalmente mantém um modelo de fundo da cena (*background model*), isto é, um modelo da aparência do ambiente sem a presença de pessoas ou objetos em movimento (Siebel, 2003). A esse modelo é aplicado um algoritmo para classificar cada *pixel* do quadro de entrada como um *pixel* que pertence a um objeto alvo (em movimento) ou não.

Problemas como mudanças climáticas, variações na iluminação do ambiente, sombras, superfícies espelhadas, objetos com movimentos oscilatórios com baixa amplitude (como galhos e folhas de vegetação sob efeito de ventos), dentre outros, fazem da segmentação de movimento uma tarefa difícil, podendo comprometer a confiabilidade e o desempenho do algoritmo utilizado nessa segmentação. Segundo Lara (2007), não existe um algoritmo robusto o suficiente para se adaptar a todas as situações apresentadas pelo ambiente. No entanto, quanto menos suposições a respeito de um dado ambiente não controlado o algoritmo necessitar, maior será sua robustez. Assim, deve-se ter em mente o tipo de aplicação final para a qual a segmentação de movimento será utilizada. Nas próximas subseções, serão apresentados os tipos de modelos de detecção de movimento mais utilizados de acordo com a literatura pesquisada.

2.1.1 Modelos de Subtração de Imagem de Fundo

Este é o tipo mais simples de detecção de movimento. Indicado para ambientes bastante controlados. O modelo de subtração de fundo de imagem (ou *background subtraction*) mais simples que

existe consiste em utilizar uma imagem estática do ambiente observada no momento em que não haja objetos alvo se movimentando pelo mesmo. A partir deste ponto, de cada quadro do vídeo subtrai-se a imagem estática, previamente definida. Este modelo é denominado não-adaptativo, pois não é tolerante a possíveis mudanças no ambiente.

Modelos não-adaptativos praticamente não são utilizados em sistemas de análise de movimento humano. Entretanto, podem ser utilizados com sucesso em sistemas de ambiente controlado como em Gonçalves and Monteiro (2007), em que a subtração de imagem estática foi utilizada na segmentação de imagens contendo ratos de laboratório utilizados em experimentos com medicamentos.

No trabalho de Siebel (2003), a detecção de movimento é realizada fornecendo uma imagem estática do ambiente sem objetos alvo. Entretanto, os valores dos *pixels* que representam o modelo podem ser atualizados caso haja variação crescente ou decrescente das respectivas intensidades em quadros consecutivos. Métodos de detecção de movimento que atualizam seu modelo de tempos em tempos são chamados de adaptativos. Em Heikkilä and Silvén (1999), cada quadro de entrada foi subtraído do modelo de fundo de imagem e ao resultado aplicado um limiar de acordo com a Equação 2.1.

$$|I_t(i,j) - B_t(i,j)| > \tau \tag{2.1}$$

em que $I_t(i,j)$ é o quadro de entrada corrente, $B_t(i,j)$ é a imagem correspondente ao modelo corrente de fundo de imagem e τ é uma constante definida experimentalmente.

Os valores de $B_t(i,j)$ são atualizados a cada segmentação utilizando a soma ponderada dada pela Equação 2.2.

$$B_{t+1}(i,j) = \alpha * I_t(i,j) + (\alpha - 1) * B_t(i,j) \tag{2.2}$$

em que α define os pesos do modelo atual e do quadro de entrada na equação de atualização.

Outra técnica comum de subtração de fundo de imagem é representar cada *pixel* do modelo pela média ou mediana de um conjunto inicial de quadros capturados do ambiente. Posteriormente, é aplicado um limiar aos pixels dos quadros de entrada, de forma a determinar se esses pixels pertencem ou não à imagem que representa o ambiente monitorado.

Os modelos baseados em subtração de fundo de imagem foram muito utilizados quando, no passado, barreiras computacionais impostas pelo hardware disponível limitavam a complexidade dos sistemas de processamento de vídeo em tempo real (Stauffer and Grimson, 1999). Diante do maior poder computacional oferecido pelos computadores atuais, a maioria das pesquisas em análise de

movimento humano faz uso de modelos mais robustos e complexos, como os métodos de detecção de movimento baseado em modelos adaptativos estatísticos.

2.1.2 Modelos Estatísticos

Métodos de segmentação de movimento que utilizam modelos estatísticos são uma evolução dos métodos que utilizam subtração de fundo de imagem. O princípio comum entre os modelos estatísticos utilizados é computar uma ou mais grandezas estatísticas para cada *pixel* e utilizá-las posteriormente para classificar os *pixels* dos quadros de entrada em *background* ou *foreground*. As medidas estatísticas de cada *pixel* do modelo podem ser atualizadas a cada nova imagem capturada ou a cada n quadros, em que n é um dos parâmetros do modelo.

Métodos de detecção de movimento com modelos estatísticos se tornaram bastante populares pois possuem a capacidade de lidar com imagens de ambientes externos contendo ruído, sombras e mudanças de iluminação (Wang et al., 2003).

Stauffer and Grimson (1999) utilizaram mistura adaptativa de curvas Gaussianas para modelar os *pixels* pertencentes ao *background* da cena. Segundo os autores, se o valor de um *pixel* for resultado da imagem de uma única superfície sob iluminação constante, uma simples função Gaussiana seria suficiente para modelar o valor deste *pixel*, considerando o ruído causado durante a aquisição da imagem. Se apenas a luz incidente sobre essa superfície fosse variável, seria necessária uma função Gaussiana adaptativa para tal modelagem. Entretanto, na prática, o valor de um *pixel* é resultado da reflexão de várias intensidades de luz em várias superfícies, por isso a opção pela mistura adaptativa de curvas Gaussianas.

Neste modelo, cada *pixel* é representado por k funções Gaussianas (tipicamente com k entre 3 e 5). Ao receber um quadro de entrada, as funções Gaussianas são utilizadas para calcular a probabilidade de que cada *pixel* pertença ou não ao *background*. *Pixels* que possuem pequena probabilidade, isto é, que não se ajustam às respectivas Gaussianas que modelam o *background*, são considerados parte do *foreground*.

A atualização do modelo é feita após o processamento de cada quadro de entrada através da avaliação da Gaussiana mais representativa dentre as k Gaussianas do modelo para o *pixel*, isto é, atualizando os parâmetros da Gaussiana mais representativa. Caso nenhuma das k funções Gaussianas represente suficientemente o *pixel* avaliado, uma nova função é adicionada à mistura em substituição à função menos representativa.

Elgammal et al. (2000) também evidenciam, através de histogramas de intensidade, o comportamento multimodal dos valores de *pixels* em uma cena de ambiente externo. Segundo Elgammal et al.

(2000), modelar essa extensa variação de valores com um pequeno número de Gaussianas torna o modelo impreciso. Por outro lado, uma quantidade excessiva de Gaussianas, levaria a um modelo que engloba quase toda a variação de tons (na escala de cinza, por exemplo).

Para superar esse problema, Elgammal et al. (2000) propuseram o uso de um modelo estatístico não-paramérico de *background* baseado na estimativa da função densidade de probabilidade dos valores de *pixels*. A função densidade é re-estimada a cada quadro de entrada processado utilizando-se a distribuição normal como função núcleo (*Normal Kernel Estimator*) e o histórico recente de valores para cada *pixel*.

A estimação da função densidade utilizando uma distribuição normal como função núcleo é uma generalização do modelo de mistura de Gaussianas, uma vez que a função estimada é resultado da soma de várias Gaussianas centradas nos valores amostrados do *pixel*. Isto permite estimar a função densidade de modo mais preciso dependendo apenas das informações mais recentes.

O sistema *W4* proposto por Haritaoglu et al. (2000), utiliza um modelo de fundo de imagem denominado bimodal, pois se baseia nos valores máximo e mínimo assumidos por cada posição de *pixel* nas imagens de entrada. O modelo de *background* é representado por uma estrutura contendo três valores para cada posição (x, y) da imagem: $m(x, y)$, $n(x, y)$ e $d(x, y)$ que correspondem, respectivamente, aos valores máximo, mínimo e a diferença máxima entre duas imagens consecutivas.

O processo completo de detecção de movimento nesse modelo consiste de três etapas: iniciação do modelo, classificação dos *pixels* e atualização do modelo (parte adaptativa do processo). A etapa de iniciação consiste em computar os valores do modelo para cada *pixel*, utilizando para tanto, imagens de alguns segundos de vídeo. A etapa de classificação dos *pixels* das imagens de entrada é feita verificando se o valor do *pixel* em classificação encontra-se entre os valores máximo e mínimo do modelo corrente, como apresentado na Equação 2.3. Esta equação é um aperfeiçoamento proposto por Jacques et al. (2005) em relação à equação original proposta no sistema *W4*.

$$M^t(x,y) = \begin{cases} 1 & \text{se } I^t(x,y) \leq |m(x,y) - kd_\mu| \text{ ou} \\ & \quad I^t(x,y) \geq |n(x,y) - kd_\mu| \\ 0 & \text{caso contrário} \end{cases} \tag{2.3}$$

em que

- $M^t(x, y)$ indica se o *pixel* (x, y) pertence ao *background* (1) ou não (0) no tempo *t*;
- $I^t(x, y)$ é a intensidade de um *pixel* na posição (x, y) no tempo t;
- $n(x, y)$ é o valor máximo do *pixel* dado pelo modelo;

- k é uma constante;
- d_μ é a mediana da diferença máxima entre imagens dada pelo modelo;
- $m(x, y)$ é o valor mínimo do *pixel* dado pelo modelo na posição (x, y).

A atualização do modelo de *background* é feita com o auxílio de mapas denominados mapas de mudança. Os valores de cada mapa são atualizados toda vez que um quadro de entrada é processado durante a fase de classificação dos *pixels*. A cada N quadros classificados, os mapas são utilizados na substituição do modelo de *background* corrente (quando necessário) pelo modelo de *foreground* ou *background* computados nos últimos N quadros. Estes dois últimos modelos são computados em paralelo ao modelo corrente (durante a fase de classificação). A substituição do modelo na fase de atualização corresponde à adaptação do modelo às mudanças do ambiente.

Existem duas formas de atualização no modelo proposto por Haritaoglu et al. (2000): baseada em *pixel* (*pixel based*) e baseada em objeto (*object based*). A primeira atualiza o modelo de *background* com relação às mudanças de iluminação no ambiente. A segunda atualiza o modelo pela incorporação de objetos que permaneçam algum tempo parados no ambiente.

Por ser baseado apenas nos valores máximo e mínimo assumidos pelos *pixels* das imagens recebidas, Haritaoglu et al. (2000) afirma que o modelo bimodal é menos complexo e mais eficiente do que os modelos propostos por Stauffer and Grimson (1999) e Elgammal et al. (2000). Entretanto, na prática, o processo de atualização do modelo bimodal a cada N imagens pode degradar o desempenho do módulo de detecção de movimento além de causar um aumento na utilização de memória disponível, pois a cada atualização é necessário aplicar um filtro da mediana temporal nos últimos N quadros armazenados com objetivo de eliminar ruído adquirido durante a aquisição das imagens. No Capítulo 5 são apresentados resultados comparativos entre dois modelos de detecção de movimento: bimodal e mistura de funções Gaussianas.

2.1.3 Modelos de Diferença Temporal

Da mesma forma que o modelo de subtração de fundo de imagem, a Diferença Temporal também é um tipo de modelo bastante simples. Consiste da subtração *pixel* a *pixel* de duas ou mais imagens de entrada consecutivas aplicadas a um limiar de classificação previamente definido. Wang and Brandstein (1998) utilizaram a diferença temporal entre quadros para detectar movimento em seu sistema e rastreamento de faces em imagens de vídeo conferência. Após a classificação dos *pixels* em *background* ou *foreground*, aplicou-se um algoritmo de checagem de vizinhança para eliminar ruídos resultantes da classificação.

Lipton et al. (1998) utilizaram diferença temporal em seu sistema de rastreamento de pessoas e veículos em ambiente externo. A detecção de movimento foi feita segundo a Equação 2.4. Nesta equação, δ_n e I_n representam, respectivamente, a diferença entre as imagens no tempo n e I_n uma imagem no tempo n.

$$\delta_n = |I_n - I_{n-1}| \tag{2.4}$$

Após a diferença temporal entre os dois últimos quadros recebidos, aplica-se a equação de limiarização para separar os objetos alvo do *background* do ambiente como apresentado na Equação 2.5, em que M representa um mapa de movimento e (u, v) representa uma posição nesse mapa.

$$M(u,v) = \begin{cases} I_n(u,v) & \text{caso } \delta_n(u,v) \geq T \\ 0 & \text{caso } \delta_n(u,v) < T \end{cases} \tag{2.5}$$

Collins et al. (2000) afirmam que modelos baseados em diferença temporal são eficazes na detecção de pequenas variações de cena. Entretanto, a segmentação dos objetos alvo com esse modelo é incompleta, uma vez que, na maioria dos casos, as imagens segmentadas possuem objetos alvo contendo espaços vazios no interior dos mesmos (ver Figura 2.2). Esse problema se deve ao fato de que os *pixels* pertencentes ao interior do objeto alvo não possuem variação de intensidade suficiente para classificá-los como *pixels* de movimento (ou *foreground*). Por esse motivo, Collins et al. (2000) propôs um método híbrido para detecção de movimento que inicialmente aplica a diferença temporal para detectar as regiões de movimento e, numa segunda etapa aplica a subtração de fundo de imagem para preencher os espaços vazios gerados pela deficiência do modelo de diferenças temporais. A Figura 2.2 apresenta um exemplo de segmentação de movimento utlizando diferenciação temporal.

Apesar do problema apresentado pelos métodos baseados em diferença temporal, sua simplicidade garante rapidez no processamento das imagens de entrada, fazendo com que este tipo de modelo seja comumente combinado com outros a fim de obter melhores resultados.

2.1.4 Modelos de Fluxo Óptico

O fluxo óptico é uma estimativa de movimento calculada entre duas ou mais imagens de entrada representada por vetores de movimento contendo intensidade e direção. Uma vez computados os vetores de movimento, uma maneira simples de implementar a detecção de movimento é aplicar um limiar sobre a norma desses vetores para classificar os *pixels* pertencentes à região em que se encontra o vetor em *background* ou *foreground*. Outra forma de conseguir detecção de movimento utilizando

fluxo óptico é através do agrupamento dos vetores de movimento como descrito em Smith and Brady (1995).

Atualmente, existem várias técnicas para computar fluxo óptico. Uma discussão sobre essa diversidade de técnicas e comparativo de desempenho é feita em Barron et al. (1994). A grande vantagem desse tipo de modelo é que o mesmo é invariante a movimento de câmera como aproximação e panorâmicas. Entretanto, modelos baseados em fluxo óptico possuem um alto consumo de recursos computacionais e ainda, um desempenho inferior se comparado aos modelos estatísticos e de subtração de fundo de imagem (Wang et al., 2003).

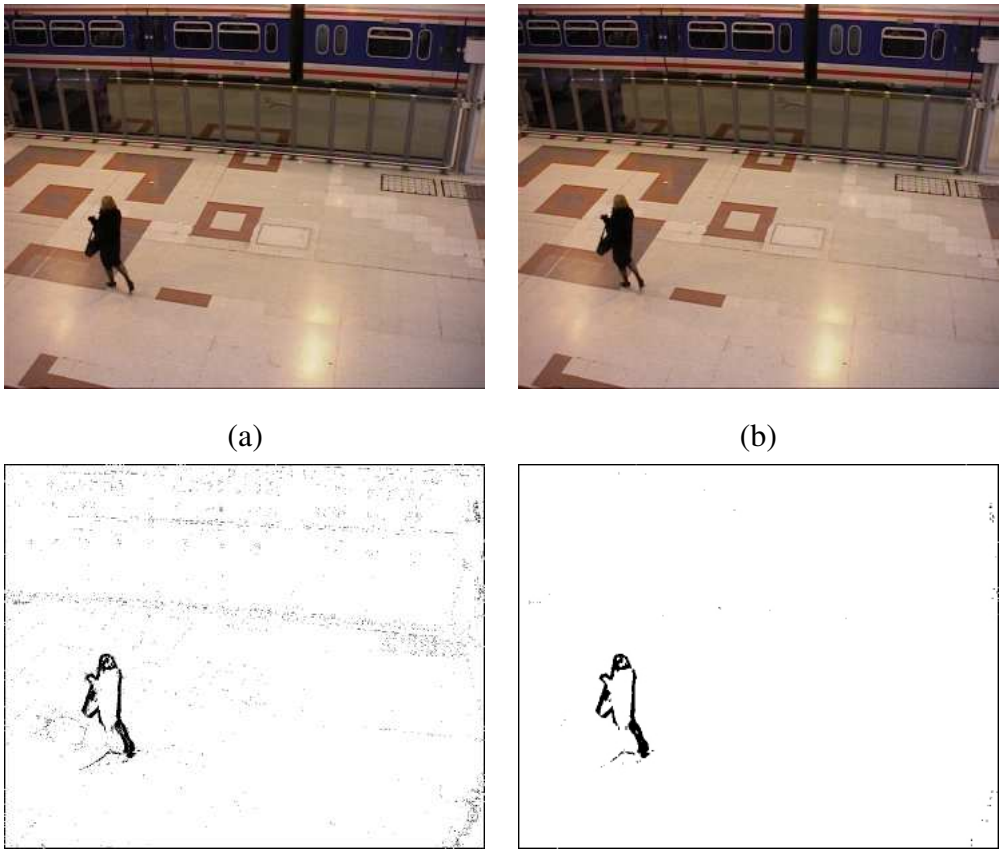


Figura 2.2: Exemplo de segmentação de movimento: (a) Quadro de vídeo no tempo t. (b) Quadro de vídeo no tempo t+1. (c) Diferença temporal. (d) Resultado da segmentação após eliminação de ruído.

2.2 Processamento Pós-Segmentação

Conforme discutido anteriormente, modelos de segmentação de movimento, por mais robustos que sejam, normalmente estão sujeitos a falhas devido a existência de sombras, ruído proveniente da aquisição das imagens pelas câmeras ou pequenos movimentos de objetos pertencentes ao ambiente como galhos de árvores balançando (no caso de imagens de ambiente externo). Por isso é comum, após a etapa de segmentação, as imagens binarizadas resultantes apresentarem falsos positivos (*pixels* que pertencem ao *foreground*, mas que foram classificados como *background*) e falsos negativos (*pixels* que pertencem ao *background*, mas que foram classificados como *foreground*). Na Figura 2.3, apresenta-se um exemplo de imagem contendo falsos positivos correspondentes à sombra apresentada na imagem original.

(a) (b)

Figura 2.3: Exemplo de problema de segmentação causado por sombras. (a) Imagem original. (b) Imagem segmentada.

Para que as etapas subsequentes à detecção de movimento (rastreamento, por exemplo) sejam bem sucedidas, problemas como os ilustrados nas figuras anteriores devem ser minimizados através de uma etapa de processamento pós-segmentação de movimento. Na Subseção 2.2.1 serão apresentadas abordagens para remoção de sombras, enquanto que a Subseção 2.2.2 exemplifica a obtenção dos grupos de *pixels* pertencentes aos objetos alvo os quais serão utilizados durante a etapa de rastreamento. Por último, na Subseção 2.2.3 são discutidos alguns artifícios utilizados na remoção de ruído proveniente da aquisição das imagens ou de falhas do modelo de segmentação.

2.2.1 Remoção de Sombras

Um dos maiores desafios da etapa de detecção de movimento é a identificação de sombras associadas aos objetos alvo. A existência de sombras causa sérios problemas às imagens geradas pelo módulo detector de movimento como, como por exemplo, distorção da forma dos objetos alvo (prejudicando a etapa de classificação desses objetos), união de objetos alvo diferentes e perda de objetos.

Segundo Prati et al. (2003), a remoção de sombras torna-se uma tarefa difícil por dois motivos: 1) regiões sombreadas são classificadas como *foreground* pois possuem *pixels* contendo características significativamente diferentes das capturadas pelo modelo de fundo do ambiente e 2) regiões sombreadas possuem movimento semelhante aos verdadeiros objetos alvo encontrados na cena. Por esses motivos, e também por motivos de desempenho, a remoção de sombras é realizada em conjunto com a etapa de segmentação de movimento, isto é, tão logo um *pixel* tenha sido classificado como *background* ou *foreground* (ou até mesmo como parte do processo de classificação).

Horprasert et al. (1999) criaram um modelo de segmentação de movimento estatístico apto a identificar falsos positivos causados pela ocorrência de regiões sombreadas. O modelo, inspirado na visão humana, utiliza estatísticas de distorção de brilho e cromaticidade para classificar os *pixels* do quadro de entrada em quatro classes:

- ***Background original*** (B)**:** se o *pixel* possui brilho e cor semelhantes ao modelo de *background*;
- ***Background sombreado*** (S)**:** se o *pixel* possui cor semelhante ao modelo de *background*, mas possui brilho menor que o modelo de *background*;
- ***Background iluminado*** (H)**:** se o *pixel* possui cor semelhante ao modelo de *background*, mas possui valor de brilho maior que o modelo de *background*;
- ***Objeto de foreground em movimento*** (F)**:** se o *pixel* possui cor e brilho diferentes do modelo de *background*.

Cucchiara et al. (2001) desenvolveram um método para remoção de sombra utilizando um espaço cromático em que as componentes de iluminação e cromaticidade estão separadas (*HSV*). Considerando que o espaço *HSV* é uma representação plausível à visão dos seres humanos (Gonzalez and Woods, 2002) , os autores constataram que áreas sombreadas possuem o componente V dos *pixels* das imagens menor que em áreas não sombreadas, enquanto que os outros componentes (H e S) permanecem quase que inalterados. Esse princípio deu origem à Equação 2.6 reproduzida abaixo.

$$SP_k(x,y) = \begin{cases} 1 & \text{se } \alpha \leq \frac{I_k^V(x,y)}{B_k^V(x,y)} \leq \beta \text{ e} \\ & |B_k^S(x,y) - I_k^S(x,y) \leq \tau_S| \text{ e } |B_k^H(x,y) - I_k^H(x,y)| \leq \tau_H \\ 0 & \text{caso contrário} \end{cases} \tag{2.6}$$

em que H,S e V são as componentes do espaço de cores, I e B são o quadro de entrada corrente e a imagem do modelo de *background*, respectivamente, α e β são limiares entre 0 e 1 que definem quão sensível à sombra será a função $SP_k(x,y)$. Caso a função assuma o valor 1, o *pixel* será classificado como sombra. Do contrário, o *pixel* será classificado como não-sombra.

Jacques et al. (2005) desenvolveram um método para remoção de sombras para o modelo bimodal de segmentação de movimento proposto por Haritaoglu et al. (2000). O método proposto não utiliza informação de cor, uma vez que o modelo bimodal de segmentação de imagens proposto por Haritaoglu et al. (2000) utiliza apenas imagens em tons de cinza. Entretanto, o princípio utilizado é parecido com as abordagens anteriormente citadas: um *pixel* pertencente a uma região sombreada difere de um *pixel* de *background* não sombreado apenas por uma fração de luz bloqueada.

A remoção de sombras em Jacques et al. (2005) é feita logo após a classificação do *pixel* em *background* ou *foreground* realizada na etapa de classificação do modelo bimodal. Pixels classificados como parte do *foreground* passam por um novo teste o qual verifica se o *pixel* em questão pertence ou não a uma região sombreada. O teste consiste em verificar se a correlação cruzada normalizada (*NCC - normal cross-correlation*) está de acordo com a inequação 2.7.

$$NCC(i,j) \geq L_{ncc} \text{ e } E_{T_(i,j)} < E_B(i,j) \tag{2.7}$$

em que

$$NCC(i,j) = \frac{ER(i,j)}{E_B(i,j)E_T(i,j)} \tag{2.8}$$

$$ER(i,j) = \sum_{n=-N}^{N} \sum_{m=-N}^{N} B(i+n, j+m) T_{i,j}(m,n) \tag{2.9}$$

$$E_B(i,j) = \sqrt{\sum_{n=-N}^{N} \sum_{m=-N}^{N} B(i+n, j+m)^2} \tag{2.10}$$

$$E_{T_(i,j)} = \sqrt{\sum_{n=-N}^{N} \sum_{m=-N}^{N} T_{i,j}(m,n)^2} \tag{2.11}$$

ER, E_B e $E_T(i,j)$ são chamadas de energias. Essas energias são calculadas na vizinhança (chamada de *template* $T(m,n)$) do *pixel* (i,j). $B(i,j)$ refere-se à imagem de *background* mantida pelo modelo e L_{ncc} é um limiar previamente definido (usualmente 0.95). Na Figura 2.4 estão destacadas as regiões sombreadas encontradas na Figura 2.3 utilizando o método proposto por Jacques et al. (2005).

(a) (b)

Figura 2.4: Exemplo de remoção de sombras. (a) Imagem original. (b) Imagem segmentada com *foreground* na cor preta e sombras na cor cinza.

Conforme mencionado anteriormente, a maioria dos métodos de remoção de sombras se baseia no princípio de que a região sombreada apresenta as mesmas características de cor (ou tom de cinza) a menos de uma atenuação de brilho. Dos métodos apresentados acima, o de Horprasert et al. (1999) apresenta maior fidelidade ao modelo biológico da visão humana, porém é o mais complexo dentre os métodos citados por utilizar informações estatísticas calculadas a partir do histórico de imagens anteriores.

O método proposto por Cucchiara et al. (2001) é bastante simples do ponto de vista computacional, visto que não requer cálculo estatístico complexo e pode ser aplicado diretamente ao *pixel* no espaço de cores HSV. O método proposto por Jacques et al. (2005) é uma opção alternativa a sistemas em que apenas imagens em tons de cinza estão disponíveis. Entretanto pode haver diminuição no desempenho em relação ao método de Cucchiara et al. (2001) devido ao cálculo das energias em cada *pixel* utilizar os valores de $2N + 1$ *pixels* vizinhos. Uma discussão aprofundada sobre métodos de remoção de sombras pode ser encontrada em Prati et al. (2003).

2.2.2 Detecção de Regiões Conectadas

Após a segmentação de movimento, devem-se separar os objetos alvo através de algoritmos de detecção de regiões conectadas para avançar nas etapas de análise de movimento humano (rastreamento e

detecção de eventos). Inicialmente, são atribuídos identificadores (*labels*) e computados retângulos mínimos contendo os objetos. Posteriormente, são computadas características que permitem a identificação desses objetos alvo em quadros posteriores. Na Figura 2.5, são ilustrados objetos alvo com seus respectivos identificadores e retângulos mínimos.

Figura 2.5: Exemplo detecção de regiões conectadas com respectivos rótulos e retângulos mínimos.

2.2.3 Remoção de Ruído

Após a remoção de sombras, as imagens resultantes da segmentação de movimento ainda podem conter falsos positivos provenientes de ruído adicionado no processo de aquisição das imagens pelas câmeras do ambiente bem como de falhas de segmentação do próprio modelo adotado. Falsos positivos dessa natureza podem ser removidos utilizando operações primitivas de processamento de imagem como filtro da mediana, erosão e dilatação (Gonzalez and Woods, 2002) .

Em alguns casos, as imagens segmentadas apresentam falsos objetos alvo, isto é, grupos de *pixels* que formam regiões conectadas, em geral pequenas, se comparadas com os verdadeiros objetos alvo. Nesse caso, uma solução simples consiste na eliminação de regiões consideradas pequenas. Essa eliminação só poderá ser feita após a etapa de detecção de regiões conectadas como descrito na Subseção 2.2.2.

Outro problema comum durante a segmentação de movimento acontece quando partes de um mesmo objeto (pé ou cabeça de uma pessoa, por exemplo) aparecem separados na imagem de saída. Siebel (2003) aplicou um algoritmo de junção (*merge*) de regiões muito próximas para contornar esse problema, porém sem muito sucesso. Dedeoglu (2004) aplicou com êxito sucessivas operações de dilatação e erosão para eliminar falhas de segmentação provenientes de ruído, fechar regiões abertas e unir partes separadas.

Apesar da variedade de estratégias adotadas para minimizar ruído de segmentação, a utilização das mesmas não trará, necessariamente, bons resultados em qualquer situação. A eliminação de regiões pequenas, por exemplo, depende da definição de um limiar para exclusão de tais regiões. Um certo limiar poderá eliminar grande parte dos falsos positivos de imagens de ambiente interno, mas poderá também eliminar objetos alvo verdadeiros em cenas de ambientes externos, em que a tomada da cena geralmente abrange uma área maior. Algoritmos de junção, por sua vez podem também atuar negativamente se mal parametrizados, pois podem realizar a junção de partes de regiões diferentes.

2.3 Classificação de Objetos

Após o processamento das etapas discutidas nas seções anteriores, o módulo de detecção de movimento possui um conjunto de objetos alvo. Entretanto, dependendo do ambiente observado, não há garantia de que esses objetos alvo correspondam realmente àqueles de interesse da aplicação. Cenas de ambientes externos podem conter seres humanos, diferentes tipos de veículos e animais. Assim, a menos que se tenha certeza de que não haverá deferentes tipos de objetos em cena, faz-se necessário a classificação dos objetos alvo em classes.

No contexto de detecção de movimento em vídeo, a maioria dos trabalhos em classificação de objetos tem como objetivo separá-los em pessoas, grupos de pessoas, veículos ou pessoas carregando objetos. De maneira geral, a classificação é feita pela extração de características dos objetos alvo. A escolha de tais características é fundamental para o sucesso da classificação. Devem-se escolher características representativas de cada classe de objetos.

Existem dois grupos de características comumente utilizadas para a etapa de classificação: as baseadas na forma do objeto (*shape based features*) e aquelas que se baseiam no movimento dos objetos (*motion based features*).

Collins et al. (2000) treinaram uma rede neural com três camadas para classificar objetos alvo em seres humanos e veículos. Foram utilizadas características baseadas na forma como dispersão (*dispersedness*), área, razão altura/largura (*aspect ratio*) do retângulo mínimo do objeto e aproximação da câmera (*zoom*). Lipton et al. (1998) também utilizou uma medida de dispersão para separar humanos de veículos. Ambas as abordagens utilizaram processo supervisionado de treinamento através de amostras de objetos previamente segmentados.

Dedeoglu (2004) também utilizou características baseadas na forma do objeto. O autor criou uma base de dados contendo informações das silhuetas de três grupos de objetos: pessoas, grupos de pessoas e veículos. Na base de dados foi armazenado o que o autor chamou de sinal de distância

(*distance signal*) da silhueta. Trata-se de um vetor contendo a distância de cada ponto da silhueta para o centro de massa do objeto. Para saber qual a classe de um novo objeto recém segmentado, calcula-se a distância entre o vetor do novo objeto e os vetores da base de dados. O novo objeto assumirá o tipo da base que apresenta menor distância em relação a esse novo objeto.

Em outro trabalho, Lipton et al. (1998) utilizaram o fluxo óptico residual (exemplo de *motion based features*) nas bordas de cada objeto para separas objetos não rígidos (seres humanos e animais) de objetos rígidos (veículos, por exemplo). Os autores basearam-se no fato de que objetos não-rígidos possuem alto fluxo óptico residual se comparado com objetos rígidos.

Haritaoglu et al. (2000) utilizaram uma combinação de características baseadas em forma e movimento. Inicialmente para cada objeto são calculados histogramas de projeção vertical e horizontal ao longo do eixo principal dos mesmos. A separação dos objetos em pessoas e outros objetos é feita verificando-se a periodicidade dos movimentos do objeto. Essa verificação é feita através da correlação de imagens. Posteriormente, objetos classificados como pessoa(s) passam pela verificação dos histogramas de projeção para classificá-los nas classes pessoa ou grupo de pessoas.

O uso de características baseadas em movimento na classificação de objetos, como periodicidade, é mais indicada para separar objetos não-rígidos, como seres humanos, de objetos não rígidos. Entretanto tais abordagens, por serem mais complexas, necessitam de maiores recursos computacionais.

O uso de características baseadas na forma é eficiente, mas possui alguns pontos que merecem atenção: o treinamento dos classificadores deve ser feito previamente à classificação dos objetos (treinamento *off-line*); pode ser necessário treinar um classificador para cada câmera (como em Collins et al. (2000)); as características utilizadas são dependentes do ponto de vista das câmeras.

Capítulo 3

Rastreamento

Após a etapa de detecção de movimento, os objetos alvo da aplicação estão separados dos elementos estáticos da cena e estarão disponíveis para o processamento das etapas de mais alto nível, como rastreamento e detecção de eventos ou reconhecimento de ações. O rastreamento consiste em estabelecer relações temporais entre objetos alvo de quadros de vídeo consecutivos, isto é, identificar, no quadro corrente de um vídeo, a localização dos objetos alvo detectados no quadro anterior.

Seguindo a idéia de que sistemas de análise de movimento possuem processamento encadeado, em que o sistema é composto por módulos e a saída de um módulo será utilizada como entrada do módulo seguinte, o rastreamento é representado por um módulo de software que tem como entrada a saída produzida pelo módulo de detecção de movimento.

Um módulo de rastreamento é baseado em um algoritmo de rastreamento. Existem diversos critérios e algumas taxonomias para classificar algoritmos de rastreamento em sistemas de análise de movimento humano. Discussões detalhadas sobre os tipos de algoritmos de rastreamento podem ser encontradas em Moeslund and Hilton (2006), Wang et al. (2003) e Elgammal et al. (2000).

Neste trabalho, por questões de simplicidade, adota-se uma classificação semelhante à utilizada por Amer (2005) e Elgammal et al. (2000), na qual algoritmos de rastreamento podem ser divididos em algoritmos que rastreiam partes do corpo das pessoas (cabeça, tronco e membros) e algoritmos que rastreiam os objetos como um todo.

Os algoritmos que rastreiam partes do corpo em sua grande maioria utilizam modelos do corpo humano (2D ou 3D) como em Haritaoglu et al. (2000). Algoritmos que rastreiam objetos como um todo podem ser subdivididos entre os que utilizam casamento de características (*feature matching*) e os que utilizam predição de posição ou estimativa de movimento. A Figura 3.1 apresenta a divisão dos algoritmos de rastreamento considerados neste trabalho.

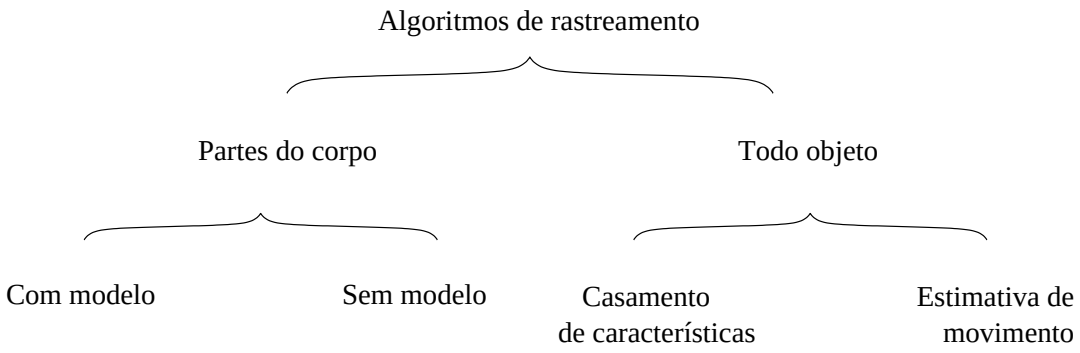


Figura 3.1: Tipos de algoritmos de rastreamento.

Os algoritmos baseados em casamento de características extraem características das regiões fornecidas pelo módulo detector de movimento. Tais características podem ser pontos, linhas, forma, textura ou cor. Uma vez que essas características sejam computadas, cabe ao algoritmo de rastreamento fazer a correspondência entre as mesmas em quadros de vídeo consecutivos.

Os algoritmos baseados em predição de posição ou estimativa de movimento vêm sendo bastante utilizados em sistemas de vigilância automática. Tais algoritmos utilizam ferramentas estatísticas como Filtros de Kalman ou Redes Bayesianas dinâmicas para estimar a posição dos objetos alvo no quadro seguinte de um vídeo.

Outro critério utilizado para classificar algoritmos de rastreamento é quanto ao número de câmeras do sistema. Um sistema de rastreamento pode lidar com objetos detectados em apenas uma ou em múltiplas câmeras. Sistemas multi-câmera proporcionam imagens de diversos pontos de vista do ambiente monitorado facilitando, portanto, o tratamento de oclusão. A oclusão ocorre quando dois objetos ocupam o mesmo lugar no espaço da imagem capturada por uma câmera. Em contra partida, algoritmos de rastreamento multi-câmera são mais complexos que os algoritmos para sistemas com uma única câmera.

Segundo Elgammal et al. (2000), o uso de modelos do corpo humano, tanto 2D quanto 3D, para rastreamento de pessoas bem como das partes do corpo humano, não é uma tarefa simples, pois a associação de cada imagem real com os modelos utilizados pode ser complexo devido ao número de parâmetros do modelo. Para o autor, rastrear objetos utilizando características visuais de baixo nível como pontos, linhas, forma e cor é computacionalmente mais eficiente.

Pelas razões mencionadas acima, as seções que seguem neste capítulo tratam de algoritmos de rastreamento baseados em similaridade de características extraídas de objetos e de predição ou estimativa de movimento. Na Seção 3.1, apresentam-se os tipos de características que podem ser extraídas dos *pixels* dos objetos de entrada do módulo bem como técnicas de correspondência dessas caracte-

rísticas. Na Seção 3.2 é apresentado o Filtro de Kalman como ferramenta matemática utilizada para fazer predição de posição ou estimativa de movimento das regiões rastreadas. Por último, nas Seções 3.3 e 3.4 descrevem-se técnicas utilizadas para rastreamento multi-câmera com objetivo de solucionar o tratamento de oclusão.

3.1 Rastreamento por Similaridade de Características

Estabelecer as devidas relações temporais entre objetos detectados em quadros de vídeo consecutivos não é uma tarefa simples, principalmente em ambientes contendo muitos objetos.

Como mencionado na seção anterior, características de baixo nível são facilmente extraídas dos objetos-alvo previamente segmentados pelo módulo de detecção de objetos, pois a computação dessas características não requer grande esforço computacional. Tais características, mantidas ao longo de quadros consecutivos, permitem identificar unicamente os objetos rastreados até mesmo em situações de oclusão (Wang et al., 2003).

Amer (2005) propôs um método de rastreamento de objetos baseado no casamento de características de tamanho, forma e deslocamento. A Tabela 3.1 apresenta algumas das características utilizadas. Segundo Amer (2005), algoritmos baseados em predição de posição ou estimativa de movimento não são adaptáveis a cenas complexas, com grande número de objetos e, consequentemente, maior número de oclusões.

O método de Amer (2005) consiste de três etapas: extração de características (*feature extration*), casamento de características (*feature matching*) e monitoramento de características (*feature monitoring*). Após a etapa de extração de características, existem duas listas de objetos: O_c (quadro corrente) e O_p (quadro anterior). A fase de casamento de características consiste de um esquema de votação dividido em dois estágios. No primeiro, é feito o casamento de cada objeto O_{p_l} pertencente ao quadro $I(n-1)$ com pelo menos um objeto O_{c_i} pertencente ao quadro corrente, $I(n)$.

A correspondência, ou casamento entre características de objetos, na maioria das vezes é feita utilizando equações de restrição, distância ou similaridade. Isto é, pela comparação das medidas obtidas através das características computadas com limiares previamente estabelecidos. A Equação 3.1 reproduz, como exemplo, a correspondência feita por Amer (2005) no primeiro estágio da fase de correspondência.

$$\begin{aligned} M_{li}: \quad & (d_{li} < t_r) \wedge (\zeta_{li} > t_v) \wedge \\ & (|w_{xic}| < w_max) \wedge (|w_{yic}| < w_max) \ . \\ \overline{M_{li}}: \quad & \text{caso contrário.} \end{aligned} \tag{3.1}$$

Na equação acima, M_{li} é uma função booleana que retorna verdadeiro quando há uma correspondência entre um objeto O_{p_l} e outro O_{c_i}, pertencentes ao quadro anterior ao quadro corrente, respectivamente; t_r, t_v, w_{max} são os limiares aplicados; d_{li} é a distância calculada entre O_{p_l} e O_{c_i}; Por último, ζ_{li} representa um valor resultante das funções de votação utilizas.

Tabela 3.1: *Características utilizadas por Amer (2005).*

Nome	**Definição**	**Descrição**
Altura	$w = y_{max} - y_{min}$	y_{max}: maior valor de y assumido por um *pixel* do objeto; y_{min}: menor valor de y assumido por um *pixel* do objeto.
Largura	$h = x_{max} - x_{min}$	x_{max}: maior valor de x assumido por um *pixel* do objeto; x_{min}: menor valor de x assumido por um *pixel* do objeto.
Área	a	Número de *pixels* do objeto.
Perímetro	p	Número de *pixels* da borda do objeto.
Retângulo mínimo	$MBB = (x_{min}, x_{max}, y_{min}, y_{max})$	Idem descrições de Altura e Largura.
Relação de extensão	$e = \frac{h}{w}$ se $h < w$ ou $e = \frac{h}{w}$ se $w < h$	h: altura; w: largura.
Compactação	$c = \frac{a}{hw}$	a: área; h: altura; w: largura.
Irregularidade	$r = \frac{p^2}{4\pi a}$	p: perímetro; π: 3,14; a: área

No segundo estágio do processo de votação proposto por Amer (2005), os objetos O_{p_l} pertencentes a $I(n-1)$ que possuem mais de uma correspondência passam por um processo de voto majoritário para eliminar ambiguidades. Objetos O_{c_i} que não possuem correspondência são considerados objetos que acabaram de entrar na cena. Objetos O_{lp} que não possuem correspondência são considerados objetos que saíram de cena.

Com inspiração nas idéias de Amer (2005), Dedeoglu (2004) apresentou um algoritmo de rastreamento bastante simples, baseado apenas no tamanho e no centro de massa dos objetos. A Tabela 3.2

apresenta as características utilizadas por Dedeoglu (2004) e suas respectivas descrições.

Tabela 3.2: *Características utilizadas por Dedeoglu (2004).*

Nome	**Definição**	**Descrição**
Tamanho	$a = w * h$	w: largura; h: altura
Centro de massa	$x_c = \frac{\sum_i^n x_i}{n} y_c = \frac{\sum_i^n y_i}{n}$	x_c: coordenada x do centro de massa do objeto; y_c: coordenada Y do centro de massa do objeto; n: número de *pixels* do objeto; x_i: coordenada x do i-ésimo *pixel* do objeto; y_i: coordenada y do i-ésimo pixel do objeto;

De maneira análoga, armazenaram-se os objetos segmentados no quadro anterior e no quadro corrente em duas listas. Para cada objeto da lista correspondente ao quadro anterior O_{p_l}, faz-se uma iteração sobre os objetos O_{c_i} da lista correspondente ao quadro corrente. Nessa iteração, calcula-se a distância Euclidiana entre os centros de massa dos objetos O_{p_l} e O_{c_i}. Ao objeto O_{c_i} com menor distância é feita uma nova verificação que leva em consideração a razão entre o tamanho dos objetos comparados, pois dois objetos muito próximos um do outro não representam, necessariamente o mesmo objeto. Esta segunda verificação está representada na Equação 3.2. Nesta Equação, s_i e s_p são os tamanhos dos objetos O_{c_i} e O_{p_l}, respectivamente.

$$\frac{s_p}{s_i} < \mu \vee \frac{s_i}{s_p} < \mu \tag{3.2}$$

No trabalho de Humphreys (2004) foi utilizada uma função de custo para relacionar os objetos de dois quadros adjacentes de vídeo. Foram utilizadas características de área, altura, largura e um histograma com 16 níveis de cinza normalizado. A função custo utilizada (reproduzida na Equação 3.3) para associar dois objetos Q e S de dois quadros consecutivos é constituída pela soma das diferenças entre as características dos dois objetos. A Tabela 3.3 apresenta o cálculo das diferenças entre as características dos objetos Q e S. Todas as diferenças foram normalizadas em relação ao objeto Q.

$$C_o(Q, S) = pArea + pHeight + pWidth + dHist \tag{3.3}$$

Uma decisão bastante relevante e inerente ao uso de algoritmos de rastreamento na similaridade de características diz respeito à quantidade e a qualidade das características utilizadas. Algoritmos

Tabela 3.3: *diferenças entre características de dois objetos Q e S utilizadas por Humphreys (2004).*

Nome	Definição	Descrição
Percentual de área	$pArea = \frac{a_Q - a_S}{a_Q}$	a_Q: área do objeto Q; a_S: área do objeto S.
Percentual de altura	$pHeight = \frac{h_Q - h_S}{h_Q}$	h_Q: altura do objeto Q; h_S: altura do objeto S.
Percentual de largura	$pWidth = \frac{w_Q - w_S}{w_Q}$	w_Q: largura do objeto Q; w_S: largura do objeto S.

com poucas características (Dedeoglu, 2004) são fáceis de implementar e possuem bom desempenho. Por outro lado, algoritmos com maior número de características e mecanismos de casamento mais seletivos (Amer, 2005) proporcionam melhores resultados. No Capítulo 5 são apresentados resultados comparativos quanto a quantidade e qualidade das características utilizadas.

3.2 Rastreamento Utilizando Filtro de Kalman

Nesta seção será apresentado o filtro de Kalman como ferramenta para rastreamento baseado na predição da posição ou velocidade dos objetos alvo da aplicação. Algoritmos baseados em modelos de movimento de primeira (Siebel, 2003) e segunda ordem (Haritaoglu et al., 2000) ou fluxo óptico (Polana and Nelson, 1994) também podem ser utilizadas com essa finalidade.

A principal diferença entre o rastreamento por similaridade de características daqueles que utilizam predição de características diz respeito ao mecanismo de associação entre objetos de quadros consecutivos. No primeiro, a associação é feita através da comparação entre cada objeto do quadro anterior com objetos segmentados no quadro corrente. As comparações são feitas através de equações de restrição que verificam as diferenças entre as características dos objetos em comparação. No segundo, o rastreamento consiste de duas etapas: predição e correção (ou atualização).

Na predição, as características do objeto são projetadas, segundo o modelo utilizado, para o quadro corrente. Na etapa de correção, as características projetadas na etapa de predição são corrigidas de acordo com a segmentação feita para o quadro corrente.

O Filtro de Kalman é definido por conjuntos de equações matemáticas que propiciam uma implementação computacional efeciente para estimar o estado de um processo, de maneira tal que o erro inerente a essa estimativa seja minimizado (Welch and Bishop, 1995). No caso do rastreamento de objetos em vídeo, o processo em questão é a dinâmica dos objetos previamente segmentados, enquanto que o estado do processo é representado pelo conjunto de características que serão estimadas

(geralmente, posição e/ou velocidade).

O Filtro de Kalman é dito ser ótimo e recursivo. Um dos aspectos que o caracteriza como ótimo é o fato do mesmo incorporar todas as informações disponíveis a respeito do processo, isto é, as diversas variáveis medidas pelo sistema bem como estatísticas sobre a acurácia dessas medições. O filtro de Kalman é dito recursivo porque projeta o estado de um processo dinâmico sem a necessidade de se manter um histórico de todos os estados anteriores.

Matematicamente, o filtro de Kalman é definido por dois conjuntos de equações. O primeiro conjunto contém as equações cuja função é projetar em um passo, isto é, do tempo $t-1$ para o tempo t as variáveis do processo. As Equações 3.4 e 3.5 reproduzidas abaixo representam as equações da etapa de predição. Nessas equações, a indicação superscrita "$^-$"ao lado de uma variável indica a projeção daquela variável.

$$x_t^- = Ax_{t-1} + Bu_{t-1} \tag{3.4}$$

$$P_t^- = AP_{t-1}A^T + Q \tag{3.5}$$

em que

- x_t^- representa a predição do vetor de estado do sistema no tempo t;
- A é a matriz de transição de estado do sistema;
- B é a matriz de controle do sistema;
- u representa o vetor de controle do sistema;
- P_t^- é a matriz de covariância do erro de x. Diz respeito a acurácia da estimativa do sistema.

O segundo conjunto de equações corresponde a etapa de correção, na qual, de posse das medições (z_t)realizadas no passo t, as projeções feitas no passo $t-1$ são corrigidas. As Equações 3.6, 3.7 e 3.8 abaixo representam as equações da etapa de correção.

$$K_t = P_t^- H^T (HP_t^- H^T + R)^{-1} \tag{3.6}$$

$$x_t = x_t^- + K_t(z_t - Hx_t^-) \tag{3.7}$$

$$P_t = (I - K_t H)P_t^- \tag{3.8}$$

em que

- K_t é a matriz de ganho de Kalman (*Kalman gain*), também chamada de fator de combinação;
- H^T é a matriz de observação do modelo, a qual mapeia o espaço de estados do processo no espaço de estados observados;
- R é a matriz de covariância que representa o ruído ocasionado durante a medição.

O fator de ganho K é o elemento que minimiza o erro no processo de correção. De maneira resumida, quando a acurácia do processo de medição (representada pela matriz R) for baixa, o valor de K também será baixo. Ou seja, deve-se confiar mais na predição do processo do que na medição utilizada para corrigir o vetor de estados. Uma descrição geral e bastante detalhada do Filtro de Kalman pode ser encontrada no trabalho de Maybeck et al. (1980).

A forma final assumida pelas equações, tanto da fase de predição como da fase de correção, depende das equações que regem a dinâmica do processo observado. São essas equações que determinam os elementos do vetor de estado x_t bem como as matrizes A, B e H. Para o rastreamento de objetos segmentados a partir de sequências de imagens, o vetor de estado dos objetos deve ser composto de uma ou mais características como posição, deslocamento, velocidade ou aceleração. A matriz de transição A de estado deverá então refletir as equações que calculam essas características, como por exemplo, as equações que regem o movimento retilíneo uniformemente variado (*MRUV*).

Uma vez definidos vetores e matrizes que compõem as equações do filtro de Kalman, o processo de rastreamento se resume a três etapas: predição, busca, e correção. A fase de busca corresponde à localização de cada objeto de um quadro no tempo $t - 1$ no quadro seguinte, de tempo t. A busca é feita de acordo com a fase de predição, isto é, a localização de um objeto é feita de acordo com a projeção de suas características, e não dentre todas as opções de objetos possíveis como realizado no rastreamento por similaridade de características.

Exemplos de como definir o vetor de estados e as demais matrizes utilizadas pelo filtro de Kalman podem ser encontradas em Russell and Norvig (2003) e Utsumi et al. (1998). No Capítulo 5 será apresentada a modelagem do Filtro de Kalman utilizada durante a fase de implementação.

3.3 Calibração de Câmera

As técnicas de rastreamento apresentadas nas seções anteriores funcionam suficientemente bem quando aplicadas a sequências de imagens fornecidas por uma única câmera. No entanto, uma das

maiores dificuldades enfrentadas pelo módulo de rastreamento é o tratamento de oclusão. A oclusão ocorre quando duas ou mais regiões de objetos rastreados ocupam a mesma posição nas imagens capturadas.

Embora existam técnicas de tratamento de oclusão para algoritmos de rastreamento de única câmera, a maneira ideal para superar o problema da oclusão é pela inclusão de mais câmeras no ambiente, isto é, adicionar redundância espacial pela captura de várias imagens do mesmo ambiente, porém, sob diferentes pontos de vista.

Na Figura 3.2 são exibidas duas imagens do mesmo ambiente sob diferentes pontos de vista. Os objetos que se movem na Figura 3.2(a) formariam, após a etapa de detecção de movimento, um único objeto. Contudo, ao consultar a imagem redundante do ambiente (Figura 3.2(b)) proveniente de uma segunda câmera, o algoritmo de rastreamento poderia utilizá-la para que continuasse rastreando corretamente dois objetos e não apenas um.

(a) (b)

Figura 3.2: Objetos vistos de diferentes pontos do mesmo ambiente: (a) Com oclusão . (b) Sem oclusão.

O principal requisito para um rastreamento multi-câmera é que a localização dos objetos segmentados nas imagens geradas pelas várias câmeras estejam sob o mesmo referencial. Para que todas as câmeras estejam em um mesmo referencial faz-se necessário o uso de transformações de perspectiva, isto é, transformações que mapeiam pontos (x, y) no plano das imagens em pontos (X, Y, Z) de um plano tridimensional com a origem no ambiente observado.

É importante lembrar que a primeira transformação de perspectiva ocorre no momento da aquisição das imagens pelas câmeras do ambiente. Por se tratar de um mapeamento entre espaços vetoriais de diferentes dimensões $(R^3 \rightarrow R^2)$, uma transformação direta no sentido inverso, isto é, de $(R^2 \rightarrow R^3)$ não alcançaria os resultados esperados, pois muitos pontos do plano (x, y) da imagem

seriam mapeados em um mesmo ponto (X, Y, Z) do ambiente.

No entanto, a partir de um conjunto pequeno de pontos cujas coordenadas, tanto do mundo real (ambiente) quanto do plano das imagens, sejam conhecidos, é possível encontrar os coeficientes de uma matriz de transformação entre dois espaços vetoriais tridimensionais: um com origem (X_0, Y_0, Z_0) no ambiente monitorado e outro com origem (x_o, y_o, z_o) no plano das imagens. Neste último, a coordenada z será desprezada. Esse procedimento no qual os coeficientes da matriz de transformação são encontrados através de um conjunto de pontos cujas coordenadas são conhecidas é chamado de calibração de câmera. Informações mais detalhadas sobre calibração de câmera são descritas em Gonzalez and Woods (2002).

Tsai (1987) desenvolveu um modelo de transformação de perspectiva capaz de mapear pontos 3D (X, Y, Z) do ambiente monitorado em pontos 2D (x, y) do plano das imagens capturadas. O modelo possui onze parâmetros, dos quais cinco deles são considerados intrínsecos, pois descrevem o modo como a câmera forma a imagem. Os outros seis parâmetros, chamados de extrínsecos, descrevem a posição e orientação da câmera em relação ao sistema de coordenadas do ambiente.

A calibração de câmera no modelo de Tsai Tsai (1987) consiste em computar tanto parâmetros intrínsecos quanto extrínsecos baseados em um número de pontos cujas coordenadas sejam conhecidas. O modelo provê dois tipos de calibração: coplanar e não-coplanar. No primeiro, os pontos amostrados do ambiente estão todos em um mesmo plano. De maneira contrária, na calibração não-coplanar os pontos amostrados estão distribuídos no espaço do ambiente.

Na próxima seção serão discutidas técnicas de tratamento de oclusão assumindo, quando necessário, um esquema de calibração de câmera e transformação entre perspectivas.

3.4 Rastreamento Multi-Câmera

Técnicas de rastreamento utilizando uma única câmera fixa são limitadas a uma pequena área do ambiente restrita ao ângulo de visão da câmera. Câmeras com capacidade de rotação e aproximação ampliam a área de cobertura do sistema, contudo exigem técnicas de segmentação mais robustas do que as apresentadas no Capítulo 2 para compensar os movimentos da câmera.

Outra limitação imposta pelo uso de uma única câmera diz respeito ao tratamento de oclusão, como mencionado no início do capítulo. Nesta seção serão discutidas técnicas para o tratamento de oclusão pela adição de mais câmeras ao ambiente monitorado.

O tratamento de oclusão, seja em sistemas de única ou múltiplas câmeras, envolve duas etapas. Inicialmente, o algoritmo de rastreamento deve ser capaz de detectar os objetos entrando ou saindo

de situações de oclusão. Posteriormente, o algoritmo deve estar apto ao rastreamento dos objetos enquanto os mesmos estiverem em situação de oclusão.

De maneira geral, os algoritmos de rastreamento atribuem possíveis estados aos objetos de cena como novo, perdido, rastreado, junção, separação. Estes dois últimos tratam das situações em que objetos entram e saem da situação de oclusão, respectivamente.

A detecção de oclusão em sistemas de câmera única ocorre durante a fase de correspondência entre os objetos do quadro anterior e do quadro corrente. A junção ocorre quando dois ou mais objetos do quadro anterior são associados a um único objeto do quadro corrente. Desse ponto em diante, um grupo de objetos é criado e atribuído um rótulo ao mesmo. A separação ocorre quando a um grupo de objetos do quadro anterior se transforma em mais de um objeto no quadro corrente.

O rastreamento dos objetos durante a oclusão em sistemas de câmera única geralmente é feito baseado na predição de alguma característica, como posição, velocidade ou deslocamento. No caso de algoritmos que utilizam filtro de Kalman, deve-se utilizar as estimativas obtidas durante a fase de predição. Detalhes de técnicas para tratamento de oclusão em sistemas de câmera única podem ser encontrados em Amer (2005), Humphreys (2004) e Dedeoglu (2004).

Para o caso de sistemas com múltiplas câmeras, o rastreamento é inicialmente realizado como em sistemas de câmera única mediante a escolha de uma câmera principal. A principal diferença ocorre quando da detecção da oclusão, a qual se dá de maneira análoga à descrição feita anteriormente.

Ao se detectar uma oclusão, o algoritmo de rastreamento tem como opção a consulta a outras câmeras vizinhas, através das quais fará a verificação efetiva de um ou mais objetos no ambiente. Desta maneira, o principal desafio imposto aos algoritmos de rastreamento em sistemas de múltiplas câmeras consiste na associação entre objetos entre quadros de vídeo gerados por diferentes câmeras.

Segundo Chang et al. (2000), existem duas abordagens para fazer a associação entre objetos de diferentes câmeras: baseados em geometria e baseado no mapeamento (ou reconhecimento de características entre câmeras. Na primeira, as características geométricas dos objetos são transformadas em um mesmo referencial antes de compará-las as demais características. Na segunda abordagem, características de objetos são extraídas dos diferentes pontos de vista das várias câmeras do ambiente para formar um conjunto de treinamento. Posteriormente este conjunto será utilizado para treinar um classificador, o qual durante a fase de testes, será utilizado para associar características de uma câmera a outra.

Os primeiros trabalhos a associar informações de diferentes câmeras baseados em geometria propuseram métodos que necessitam, previamente, de uma etapa de calibração para que o algoritmo possa mapear as coordenadas dos planos das imagens nas coordenadas do mundo real (ambiente).

Kelly et al. (1995) apresentaram uma abordagem para rastrear pessoas entre câmeras baseada em um ambiente 3D utilizando calibração de câmera. No momento em que um objeto sai do campo de visão [1] de uma câmera, apenas as coordenadas 3D do ambiente são utilizadas para localizá-lo no campo de visão das demais câmeras espalhadas pelo ambiente.

Posteriormente, alguns trabalhos (Khan et al. (2001); Chang et al. (2000)) apresentaram abordagens em que a associação dos objetos entre câmeras é feita através de geometria epipolar e de marcações feitas no ambiente. A geometria epipolar refere-se à geometria da visão estéreo entre duas câmeras.

A associação entre objetos entre câmeras diferentes com base em geometria é, na maioria das vezes, complementada por uma correspondência de características. No entanto, deve-se ter em mente que as características baseadas na forma e cor dos objetos podem mudar drasticamente dependendo do posicionamento das câmeras (Agarwall 96).

De maneira geral, sistemas de rastreamento baseados em múltiplas câmeras precisam decidir, a cada instante, a partir de qual câmera serão utilizadas imagens para rastrear os objetos ou contornar problemas de oclusão. Em outras palavras, o desafio para o êxito do rastreamento multi-câmera está em como selecionar, associar e tratar dados entre câmeras.

[1] É comum o uso da sigla FOV, do inglês *Field of View*.

Capítulo 4

Detecção de Ações

Após a etapa de rastreamento dos objetos que se movem entre quadros consecutivos de vídeo ocorrem, naturalmente, os problemas de compreensão de comportamento em cena, reconhecimento de ações, atividades e eventos. Segundo Wang et al. (2003), o desenvolvimento de técnicas de visão computacional para entendimento do comportamento humano em vídeo deve guiar o surgimento de muitas aplicações em análise de movimento.

Segundo Moeslund and Hilton (2006), o campo da representação e do reconhecimento de ações através de vídeo não é relativamente novo, porém ainda imaturo. Prova disso, seria o grande número de estudos com diferentes abordagens para os problemas inerentes ao tema. Ainda segundo o autor, não há um consenso a respeito da terminologia do assunto. Muitos autores utilizam os termos *ações*, *atividades*, *eventos* e *comportamento* indiferentemente em seus trabalhos. Algumas tentativas de padronização ou hierarquia de termos foram feitas por Mohan (2001), Bobick (1997) e Aggarwal and Park (2004).

Na presente dissertação, adotou-se uma hierarquia de termos semelhante à adotada por Moeslund and Hilton (2006) em que *atividades* são compostas de *ações*. *Ações*, são compostas de outras ações mais simples, denominadas de *primitivas*. O conceito de ação primitiva depende dos objetivos da aplicação. Em Yamato et al. (1992), por exemplo, foi apresentada uma técnica para reconhecimento da atividade *jogar tênis* na qual possíveis primitivas como *forehand*, *backhand*, *run left*, e *run right* estavam associadas à ação *retornar a bola*. Os termos, *compreensão de cena* ou de *comportamento em cena* são considerados como atividades de mais alto nível de abstração e grau de complexidade não abordados neste trabalho.

Segundo Wang et al. (2003), o problema da detecção de ações deve ser conceituado como um problema de classificação de dados com características que variam no tempo, isto é, um problema

de correspondência entre uma sequência desconhecida de dados e grupos de sequências previamente rotuladas representando ações humanas de determinado interesse. Deste ponto em diante, os termos reconhecimento e classificação de ações serão utilizados com o mesmo significado.

4.1 Técnicas Tradicionais

Os primeiros esforços (Polana and Nelson, 1994) (Yamato et al., 1992) na área de classificação de ações utilizaram técnicas de reconhecimento de padrões bem sucedidas como Cadeias de Markov Escondidas (*Hidden Markov Model - HMM*) e Casamento de Padrões *(Template Matching)*. Essas técnicas foram utilizadas com sucesso em problemas como reconhecimento de voz, reconhecimento de manuscritos ou modelagem dos movimentos de olhos.

De maneira resumida, uma *HMM* é um conjunto de estados conectados por arestas que representam as transições entre esses estados. Matematicamente uma *HMM* é uma tripla $\lambda = \{A, B, \pi\}$ composta por três matrizes: matriz de probabilidade de transição de estados (A), matriz de probabilidade dos símbolos de saída (B) e a matriz inicial de probabilidades π. Na Figura 4.1 está ilustrada uma representação gráfica de uma *HMM*.

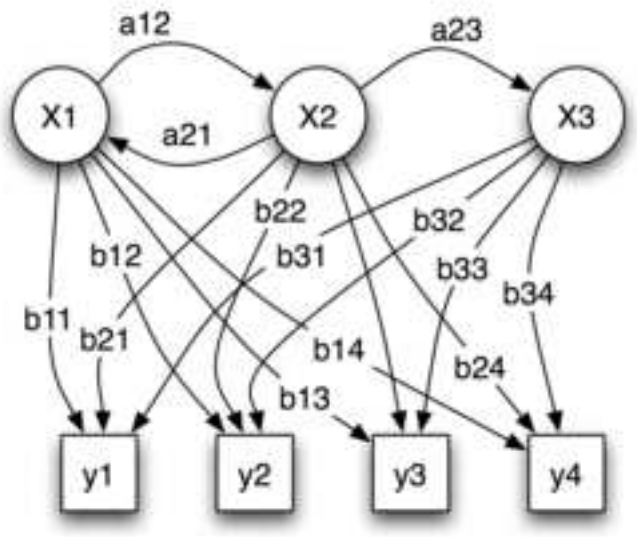


Figura 4.1: Representação gráfica de uma Cadeia de Markov Escondida.

Yamato et al. (1992) utilizaram *HMM* para classificar ações primitivas de uma partida de tênis através de vetores de características de baixo nível extraídos de sequências de imagens. As primitivas

em questão foram movimentos característicos do esporte: *forehand stroke*, *backhand stroke*, *forehand volley*, *backhand volley*, *smash*, and *service*.

Em seu estudo, Yamato et al. (1992) representou cada primitiva por uma *HMM*. O processo de reconhecimento consistiu então em descobrir qual das *HMMs* melhor representava uma sequência de vetores de características extraídos das imagens de entrada. Para compor o vetor de características, imagens de entrada binarizadas foram decompostas em células de $M \times N$ *pixels*. O número de *pixels* de *foreground* de cada célula representaram o vetor de características, que posteriormente, foram mapeados em símbolos através de quantização vetorial. Por fim, os símbolos obtidos foram utilizados como entrada tanto na fase de treinamento quanto na fase de testes junto as *HMMs*. A Figura 4.2 apresenta o esquema proposto por Yamato et al. (1992).

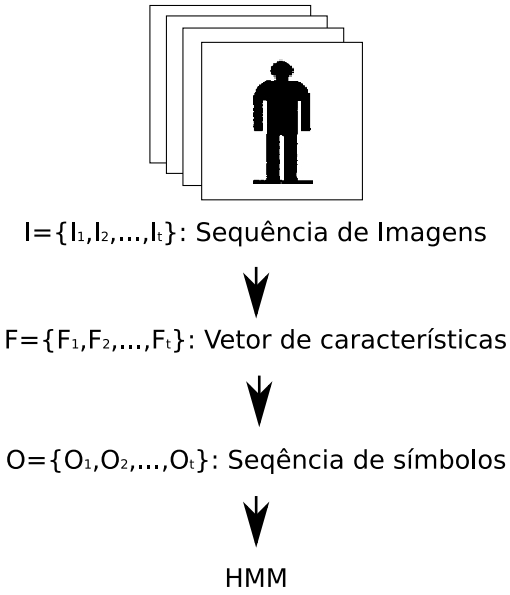


Figura 4.2: Fluxo do processo proposto por Yamato et al. (1992).

A técnica de Casamento de Padrões (*Template Matching*) consiste em compor um vetor de características para comparação com um conjunto de vetores de características ditos representantes das classes de ações que se deseja reconhecer.

Polana and Nelson (1994) também dividiram cada região da imagem binarizada de entrada em células. As magnitudes de movimento (fluxo óptico) de cada célula foram somadas ao longo do tempo e comparadas com limiares de periodicidade, a fim de verificar se o movimento de cada região é ou não periódico. A frequência dos ciclos de movimento dos objetos foi então utilizada para o reconhecimento de primitivas como *andar* ou *correr*.

Ribeiro and Santos-Victor (2005) apresentaram um trabalho que abordou o problema de reco-

nhecer ações humanas primitivas como *ativo*, *inativo*, *andando*, *correndo* e *lutando* com ênfase na seleção de características de baixo nível (velocidade e fluxo óptico), modelagem dos dados e estrutura de classificador. Os autores utilizaram um classificador Bayesiano através do qual foram calculadas as probabilidades de cada ação dado um vetor de características extraído de regiões previamente segmentadas (Equação 4.1).

$$P(A_j|F(t)) = \frac{P(F(t)|A_j)P(A_j)}{P(F(t))} \tag{4.1}$$

em que

- $P(A_j|F(t))$ é a probabilidade de uma determinada ação dado um vetor de características;
- $P(F(t)|A_j)$ é a probabilidade de um vetor de características dada uma determinada ação;
- $P(A_j)$ é a probabilidade de uma determinada ação;
- $P(F(t))$ é a probabilidade da ocorrência de um determinado vetor de características;

As funções de probabilidade de cada ação foram modeladas através de mistura de funções Gaussianas como definido na Equação 4.2. Nesta equação, η representa a *j*-ésima função Gaussiana com parâmetros μ_j (média) e σ_j (variância) multiplicada pelo respectivo peso ω_j.

A fase de treinamento, isto é, obtenção das funções probabilidade de cada ação em através do vetor de características, utilizou bases de dados rotuladas, das quais foram extraídas as características de baixo nível separadas por classe de ações. O algoritmo de *Expectation Maximization* foi utilizado na modelagem dos dados. O autor destaca que a complexidade da estimativa das funções de probabilidade está diretamente relacionada com a quantidade de características utilizadas para cada classe de atividade.

$$P(F(t)|A_j) \approx \sum_{j=1}^{N} \omega_j \eta(x, \mu_j, \sigma_j) \tag{4.2}$$

Outras técnicas comumente mencionadas em levantamentos na área de análise de movimento para o problema da classificação de ações são: *DTW* (*Dynamic Time Warping*) (Bobick and Wilson, 1995), Redes Neurais (Lin et al., 1999) e *PCA* (*Principal Component Analysis*).

4.2 Técnicas Baseadas em Simples Heurísticas

Os trabalhos citados na seção anterior utilizaram características de baixo nível (algumas citadas no Capítulo 3) extraídas das sequências de imagens de vídeo, em conjunto com técnicas tradicionais de

modelagem de dados em problemas da análise de movimento (incluindo rastreamento e reconhecimento de ações). Há ainda, trabalhos recentes que abordaram o problema de reconhecimento de ações primitivas através de simples heurísticas.

O encontro *PETS*[1] (*Performance Evaluation of Tracking Systems*) realizado em 2006, teve como tema principal o abandono de bagagens e objetos suspeitos em locais públicos. Tais sistemas são de grande utilidade na política de prevenção a atos terroristas (ver Figura 4.3).

Auvinet et al. (2006) apresentaram uma heurística baseada em três componentes: o rastreamento dos objetos, formando entidades espaço-temporais; a detecção espaço-temporal de divisões entre as regiões rastreadas; e a detecção de objetos imóveis. Baseado nas saídas desses componentes, um alarme é gerado toda vez que uma divisão é detectada e um dos objetos resultantes permanece imóvel.

(a) (b)

Figura 4.3: Quadros extraídos do conjunto de dados *PETS* 2006 exibindo abandono de bagagem em uma estação de trem.

Rincón et al. (2006) desenvolveram um modelo de subtração de *background* duplo no qual potenciais objetos estáticos são detectados previamente às etapas de rastreamento e reconhecimento de ações. Posteriormente algumas restrições configuráveis de tamanho e tempo de abandono são aplicadas para evitar falsos positivos. O módulo de rastreamento neste trabalho foi utilizado para identificação de pessoas próximas a esses objetos como potencias responsáveis pelo abandono.

Dedeoglu (2004) também utilizou o módulo de subtração de *background* em conjunto com informação de cor dos objetos rastreados para discriminar entre objetos abandonados e objetos removidos do ambiente monitorado. Segundo a estratégia utilizada, se a trajetória recente de um objeto informa que o mesmo não se move por algum período, o sistema decide que a região do objeto é possivelmente um candidato a objeto removido ou abandonado. O sistema então decide o tipo do objeto pelo confronto de histogramas de cores entre o objeto e a região do modelo de *background* ao redor do objeto. Ao contrário dos outros trabalhos citados, em Dedeoglu (2004) o sistema não rastreia pessoas

[1] http://www.pets2006.net

responsáveis pelo abandono ou remoção dos objetos (ver Figura 4.4).

(a) (b)

Figura 4.4: Exemplo de abandono de bagagem apresentado em Dedeoglu (2004). (a) Imagem original. (b) Destaque para mochila abandonada e regiões *S* e *R* avaliadas para decidir o tipo de objeto detectado (abandonado ou removido).

O desafio básico no problema de reconhecimento de ações através de técnicas de classificação está em modelar o mecanismo de aprendizagem a partir de amostras de treinamento, e ainda, relacionar treinamento e classificação (correspondência) de forma a se adequar às variações de tempo e escala inerentes aos padrões de movimento de cada classe. O uso de heurísticas simples pode ser eficaz em alguns casos, como o abandono e a remoção de objetos, porém limitado se aplicado a problemas de complexidade maior. No Capítulo 5, será apresentado um algoritmo para detecção de abandono e remoção de objetos pela combinação de algumas das idéias pesquisadas na literatura.

Capítulo 5

Proposta de um Sistema de Vigilância Automática a partir de Análise de Vídeo

Após uma revisão (Capítulos 2, 3 e 4) sobre técnicas comumente utilizadas pelos módulos que compõem um sistema de vigilância automática, este capítulo inicia-se na Seção 5.1 com a apresentação de uma proposta de arquitetura que vislumbra a integração dos módulos anteriormente discutidos. A Seção 5.2 traz uma descrição mais detalhada de como alguns dos módulos da arquitetura proposta foram implementados.

5.1 Arquitetura do Sistema Proposto

Antes que sejam apresentados os detalhes de implementação e os resultados obtidos por cada módulo implementado, faz-se necessário apresentar os conceitos envolvidos em um sistema de vigilância automática, assim como um esboço da arquitetura na qual esses conceitos estão inseridos.

O modelo conceitual apresentado na Figura 5.1 serve de referência para entender o relacionamento entre os elementos no domínio de um sistema de vigilância. Os principais conceitos apresentados são o detector de movimento, o rastreador e o detector de eventos.

Como descrito no Capítulo 2, o detector de movimento tem como entrada seqüências de quadros captados no ambiente monitorado. A maneira como essa seqüência de quadros de vídeo é obtida pode variar de três formas: fluxo contínuo, arquivo de vídeo ou um conjunto de arquivos em que cada arquivo representa um quadro de entrada para o detector de movimento.

O fluxo contínuo (*streaming* em inglês) é recomendado para sistemas de vigilância em tempo real. O fluxo pode ser obtido, por exemplo, através de uma conexão de rede com um servidor de fluxo ou

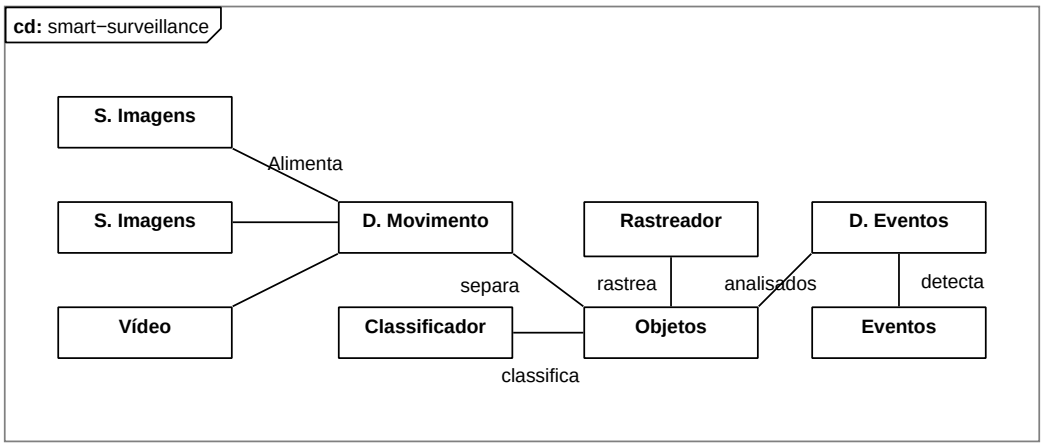


Figura 5.1: Modelo conceitual para um sistema de vigilância automática.

até mesmo diretamente da câmera. Arquivo de vídeo e conjunto de arquivos de imagens geralmente são utilizados na fase de testes do sistema, pois permitem que os resultados obtidos sejam validados pelas respectivas saídas desejadas de cada entrada. Na Seção 5.2 serão descritas formas de validação considerando as saídas desejadas.

De acordo com a Figura 5.1, como saída do detector de movimento são criados os objetos alvo da aplicação. Opcionalmente, a depender do ambiente e da aplicação, haverá necessidade de classificação desses objetos como discutido na Seção 2.3. Os objetos alvo da aplicação serão então acompanhados pelo rastreador de objetos, cuja responsabilidade é estabelecer correspondência temporal e manter o histórico dos objetos. Por último, o detector de eventos utiliza todas as informações disponíveis dos objetos para detectar os eventos de interesse da aplicação.

Apresentados os conceitos e relacionamentos pertinentes a um sistema de vigilância automática, pode-se ampliar a visão geral do sistema através de uma representação que vislumbre a interação entre os elementos no ambiente de execução, ou seja, uma representação que descreva a arquitetura do sistema.

Uma proposta de arquitetura distribuída para sistema de vigilância multi-câmera em ambiente fechado pode ser vista na Figura 5.2. A figura mostra câmeras espalhadas pelo ambiente contendo pessoas que se movimentam pelo ambiente. Como sugere a figura, as seqüências de imagens são obtidas pelas estações de observação através de fluxos contínuos lançados na rede local. Num outro cenário possível, porém menos flexível, cada câmera poderia ser diretamente conectada à estação de observação.

As estações de observação são responsáveis pela captura das imagens geradas pelas câmeras,

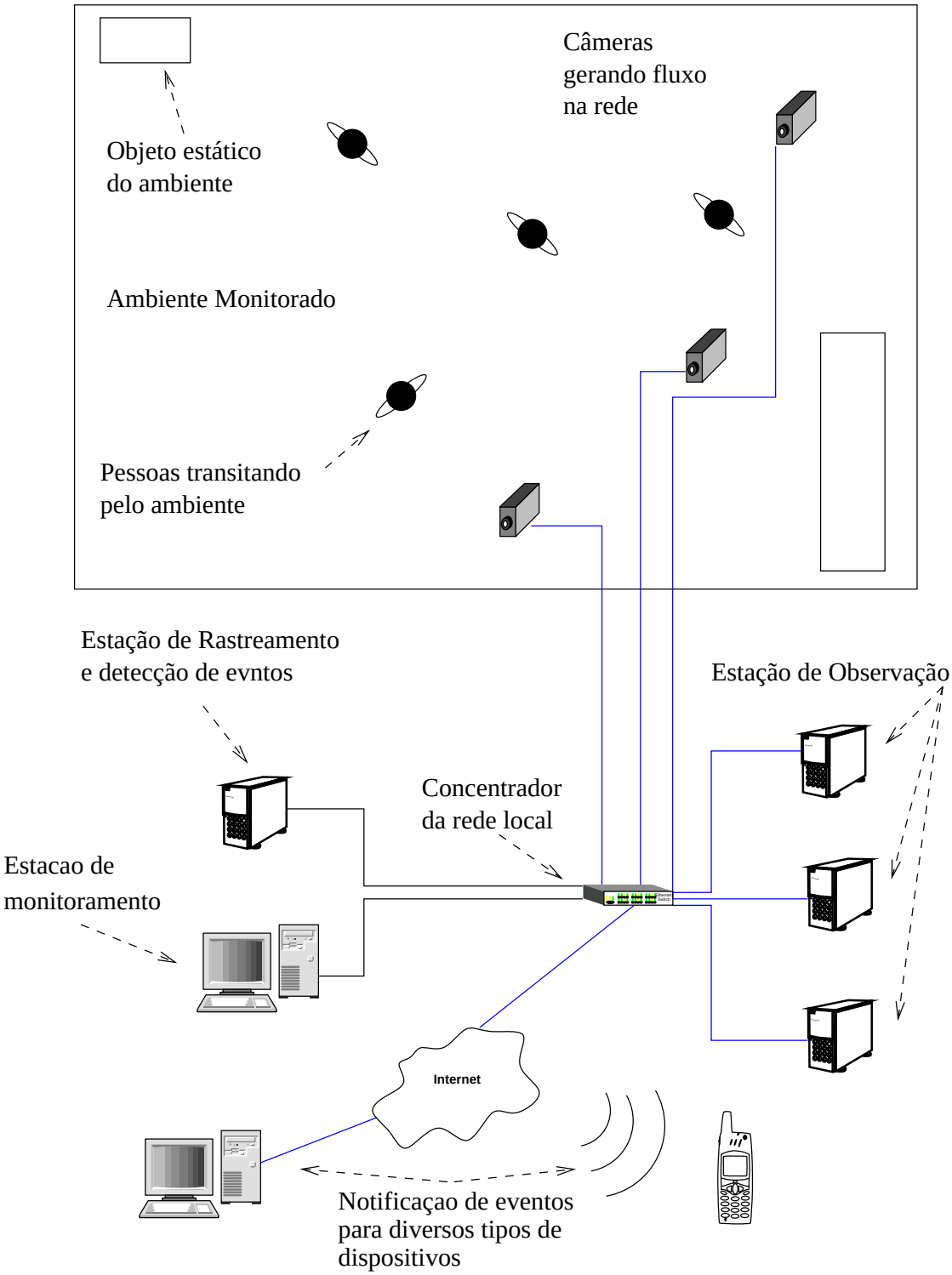


Figura 5.2: Proposta de arquitetura para um sistema de vigilância automática.

detecção de movimento e, opcionalmente, classificação dos objetos segmentados. Cada estação de observação processa as imagens geradas por uma das câmeras (Utsumi et al. (1998)).

O processamento distribuído das imagens de entrada, como mostrado na Figura 5.2, confere maior escalabilidade no caso de sistemas multi-câmera, pois a adição de uma nova câmera no ambiente requer a adição de nova estação de observação, não sobrecarregando a estação de rastreamento e detecção de eventos. Entretanto, como citado no Capítulo 3, sistemas multi-câmera exigem dos algoritmos de rastreamento a capacidade de relacionar os mesmos objetos entre câmeras distintas.

Após o processamento das imagens de entrada, as estações de observação enviam para a estação de rastreamento e detecção de Eventos apenas as características relevantes dos objetos detectados. Essas características serão utilizadas pelo algoritmo de rastreamento na correspondência entre objetos recém detectados e aqueles detectados no passado. É possível, entretanto, que essa correspondência seja feita pelas estações de observação como em Utsumi et al. (1998). Nesse caso, devem-se utilizar algoritmos de rastreamento que utilizam o mecanismo de estimativa da posição dos objetos. Assim, a estação de rastreamento envia as informações de predição para as estações de observação e estas realizam a fase de atualização, devolvendo posição e demais características atualizadas para a estação de rastreamento.

Seguindo a arquitetura proposta, a comunicação entre os processos que executam os módulos do sistema é feita através do barramento compartilhado na rede local. No caso em que as informações trafegam apenas das estações de observação para a estação de rastreamento, é necessário que as estações de observação conheçam o endereço da estação de rastreamento. Por outro lado, havendo necessidade das informações trafegarem no sentido contrário, isto é, da estação de rastreamento para as de observação, é também necessário que o endereço de cada uma das estações de observação seja conhecido pela estação de rastreamento.

Uma estratégia eficiente de comunicação na qual a estação de rastreamento envia informações para as demais estações se dá através do uso de infra-estrutura *multicast* [1], na qual uma entidade de rede envia informação para múltiplos destinos por meio de uma única transmissão (Deering and Cheriton, 1985). Dessa maneira, em vez de replicar a mesma informação para cada estação destino, a estação de rastreamento envia as informações apenas para o grupo *multicast* do qual as demais estações farão parte.

Outra vantagem que o uso da infra-estrutura de *multicast* pode oferecer está na captura das informações geradas do sistema pelas estações de monitoramento. Estas são estações que recebem informações de todos os níveis de abstração do sistema, ou seja, desde os fluxos de vídeo gerados

[1] http://www.ietf.org/rfc/rfc0966.txt?number=0966.

pelas câmeras, informação dos objetos rastreados e os alarmes gerados pelo módulo detector de eventos.

O uso de transmissão de rede tradicional (*unicast*) pode comprometer o desempenho da rede caso haja um número excessivo de estações de monitoramento, pois o fluxo de vídeo das câmeras seria replicado para cada uma das estações de monitoramento. Por outro lado, com uso de *multicast*, para cada câmera será gerado apenas um fluxo de vídeo endereçado por um grupo *multicast*.

Uma possível desvantagem no uso dessa tecnologia está na necessidade de que os equipamentos de interconexão de rede utilizados sejam capazes de lidar com esse tipo de transmissão.

Ainda de acordo com a Figura 5.2, após o rastreamento, um histórico de informações dos objetos alvo estará disponível para o detector de eventos, cuja função é detectar, registrar e notificar eventos de interesse da aplicação através das técnicas discutidas no Capítulo 4. A figura contempla ainda a possibilidade de notificação dos eventos para diversos tipos de dispositivos (locais ou remotos) como estações de monitoramento ou dispositivos móveis.

Na próxima seção será feita uma descrição geral da implementação das principais partes da arquitetura proposta para em seguida apresentar e discutir os resultados obtidos.

5.2 Detalhes de Implementação

Como fugiria ao escopo de um trabalho em nível de mestrado a implementação de toda a proposta de arquitetura para um sistema automático de monitoramento a partir de vídeos, apresentada na seção anterior, foram selecionados para implementação e validação apenas os módulos mais representativos em termos de requisitos funcionais, ou seja, aqueles indispensáveis para a solução dos problemas investigados neste trabalho. Os módulos escolhidos foram o detector de movimento, o módulo rastreador e o detector de acções primitivas.

Considerando o processamento sequencial entre os módulos implementados e, para facilitar a validação dos resultados deste trabalho, a comunicação entre os módulos de detecção de movimento, rastreamento e detecção de eventos foi feita através de sistemas de arquivos, ou seja, a saída do módulo detector de movimento gravada no sistema de arquivos serve de entrada para o módulo de rastreamento e assim por diante.

A linguagem *C* foi escolhida para as atividades de implementação por ser uma linguagem que atende aos requisistos de desempenho de aplicaçõesde em visão computacional devido a eficiência dos compiladores existentes na geração de código nativo. Como ambiente de desenvolvimento, foi

utilizada a plataforma *Eclipse*[2] juntamente com o pacote (*plugin*) *CDT* (*C Develpment Tool*) para desenvolvimento de aplicações em linguagem *C*.

Além da linguagem e do ambiente de desenvolvimento, foram utilizadas duas bibliotecas de funções também implementadas em linguagem *C*: *OpenCV* versão 1.0 e *FFMPEG* versão 0.49. A biblioteca *OpenCV*[3], desenvolvida originalmente pela Intel®, é uma biblioteca de código aberto voltada para aplicações de visão computacional que implementa tanto operações básicas de processamento digital de imagem como representação e acesso aos *pixels* de uma imagem, além de ferramentas matemáticas complexas como o Filtro de Kalman.

FFMPEG[4] é uma biblioteca de código aberto que permite a codificação e decodificação de arquivos, fluxos de vídeo e conjuntos de imagens. Atualmente, essa biblioteca serve de base para diversas aplicações em plataforma Linux como *MPLayer* [5], *xine*[6] e *VLC* [7]. Neste trabalho, a biblioteca *FFMPEG* foi utilizada na implementação de uma fonte genérica de imagens, pois, como mencionado na Seção 5.1, um módulo detector de movimento pode receber imagens de diferentes tipos de fontes.

As Figuras 5.3 e 5.4 apresentam, respectivamente, um diagrama de componentes e outro de atividades para o módulo detector de movimento. O diagrama de componentes permite visualizar as relações entre o detector de movimento e a fonte de imagens, denominada de *Seletor*.

Outra flexibilidade apresentada pela figura diz respeito a possibilidade de utilização de diferentes modelos de segmentação, uma vez que implementam as assinaturas da interface *BGModel*. O componente classificador, como mencionado anteriormente, é opcional.

O diagrama de atividades (Figura 5.4) ilustra as atividades realizadas pelo processo que representa o módulo detector de movimento. De acordo com a figura, o processo recebe os parâmetros de configuração a partir de um arquivo texto. Dentre o conjunto de opções disponíveis para o módulo, as que definem a fonte/destino das imagens e o modelo de segmentação a ser utilizado são as mais importantes. Realizada a leitura das configurações, cabe ao detector de movimento requisitar ao seletor as imagens de entrada e enviar, através do mesmo as imagens produzidas na saída.

Na Figura 5.5 apresenta-se, analogamente, o diagrama de atividades para o módulo rastreador de objetos. Nota-se pela Figura a mesma flexibilidade que a do algoritmo de rastreamento.

De acordo com a Figura 5.5, o processo de rastreamento recebe os parâmetros de configuração a

[2]http://www.eclipse.org.

[3]http://www.intel.com/technology/computing/opencv.

[4]http://ffmpeg.mplayerhq.hu.

[5]http://www.mplayerhq.hu.

[6]http://xinehq.de.

[7]http://www.videolan.org/vlc.

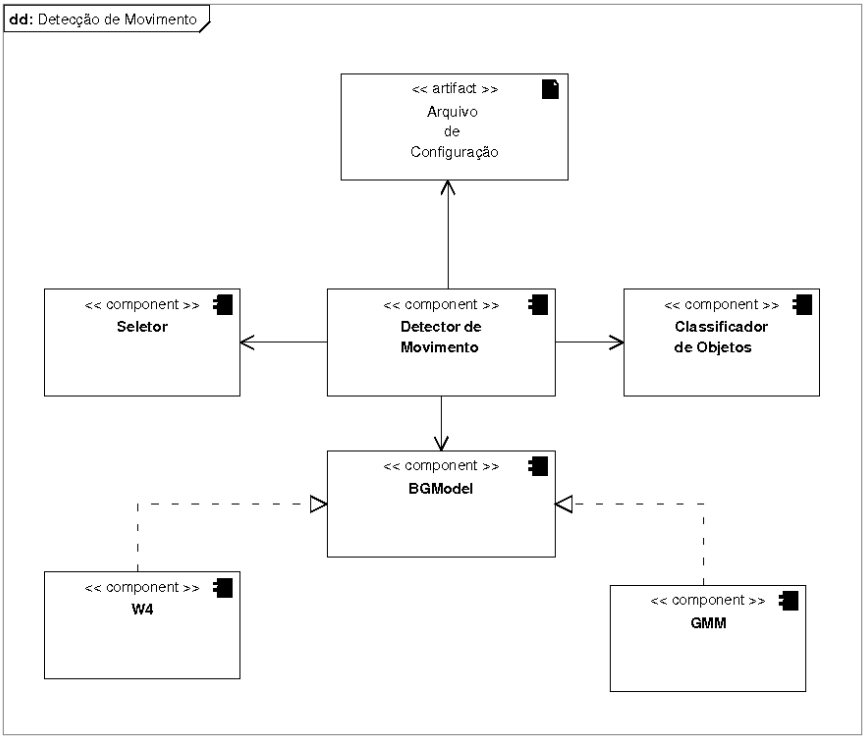


Figura 5.3: Diagrama de componentes com módulo detector de movimento e fonte genérica de imagens.

partir de um arquivo texto. Dentre o conjunto de opções disponíveis para esse módulo, estão as que definem a localização do arquivo com os objetos segmentados pelo módulo detector de movimento, o algoritmo de rastreamento a ser utilizado e as saídas produzidas pelo rastreamento.

O módulo de rastreamento permite dois tipos de saídas: em arquivo texto e seqüências de quadros marcados tomando como base os próprios quadros do vídeo de entrada. A saída em arquivo texto contém os objetos rastreados por quadro de entrada. Cada objeto rastreado possui além das características já adicionadas na segmentação de movimento, informação de identificação, estado e tipo de objeto. As saídas em imagem são os próprios quadros do vídeo de entrada, marcados com a identificação numérica e os retângulos mínimos de cada um dos objetos rastreados.

A cada iteração, o módulo de rastreamento carrega a lista dos objetos do próximo quadro, que estão armazenados no arquivo resultante da segmentação e os passa para o algoritmo de rastreamento selecionado, juntamente com a lista de objetos rastreados para que o algoritmo faça a correspondência entre essas duas listas de objetos.

Por último, na Figura 5.6 está ilustrado o diagrama de atividades para o módulo de detecção de eventos. O processo de detecção de eventos, assim como os demais módulos, também recebe

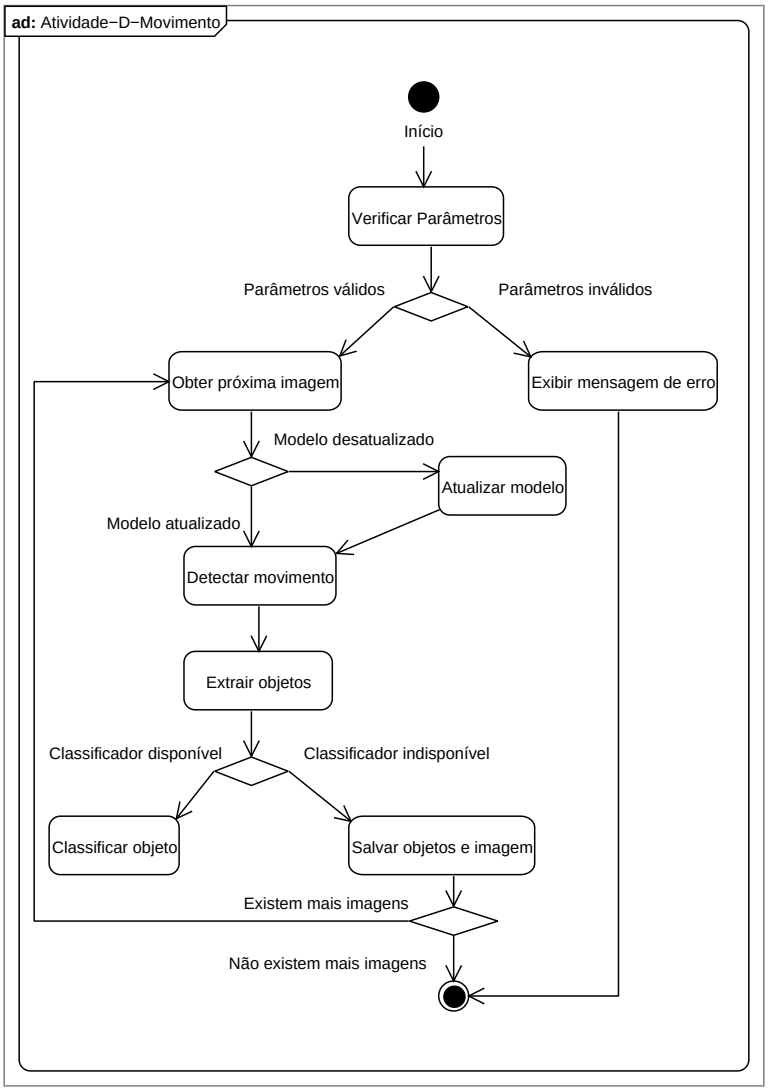


Figura 5.4: Diagrama de atividades do módulo detector de movimento.

os parâmetros de configuração a partir de um arquivo texto. Os principais parâmetros configuráveis desse módulo são a localização do arquivo com as informações dos objetos rastreados pelo módulo de rastreamento e a localização das saídas produzidas pelo módulo. Como mencionado no Capítulo 4, a detecção de eventos implementada neste trabalho baseou-se em heurísticas simples para detecção de abandono e subtração de objetos

O módulo de detecção de eventos permite dois tipos de saídas: em forma de texto e nas próprias imagens de entrada. As saídas em forma de texto, chamadas de alarmes, são geradas cada vez que for detectado um abandono ou subtração de um objeto no ambiente monitorado. O segundo tipo de saída consiste em marcar os retângulos mínimos dos objetos envolvidos em cada alarme, destacando

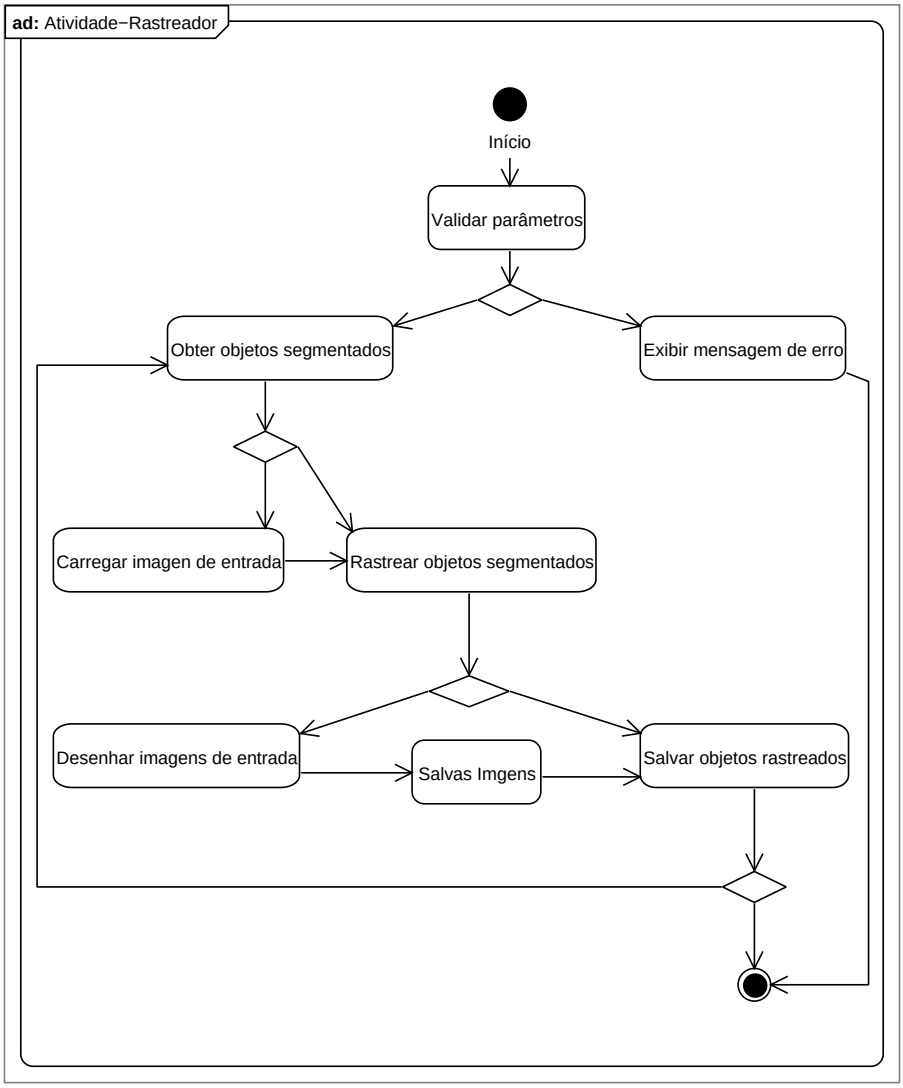


Figura 5.5: Diagrama de atividades do módulo rastreador de objetos.

o identificador numérico e tipo (abandono ou remoção).

Durante o processo de detecção de eventos, o módulo mantém duas listas: uma que mantém os alarmes (eventos) encontrados e outra contendo os objetos rastreados no quadro corrente. Um alarme deve conter pelo menos dois objetos. O primeiro corresponde ao objeto que abandona (ou remove) objetos do ambiente. O segundo diz respeito ao objeto que sofre a ação (abandono ou remoção).

A cada iteração, o módulo detector de eventos carrega a lista dos objetos rastreados no quadro corrente, que estão armazenados no arquivo resultante do rastreamento. Inicialmente, o módulo realiza dois tipos de buscas nesta lista de objetos. A primeira com objetivo de localizar os objetos de eventos (alarmes) já existentes. A segunda busca por objetos que representem novos alarmes que

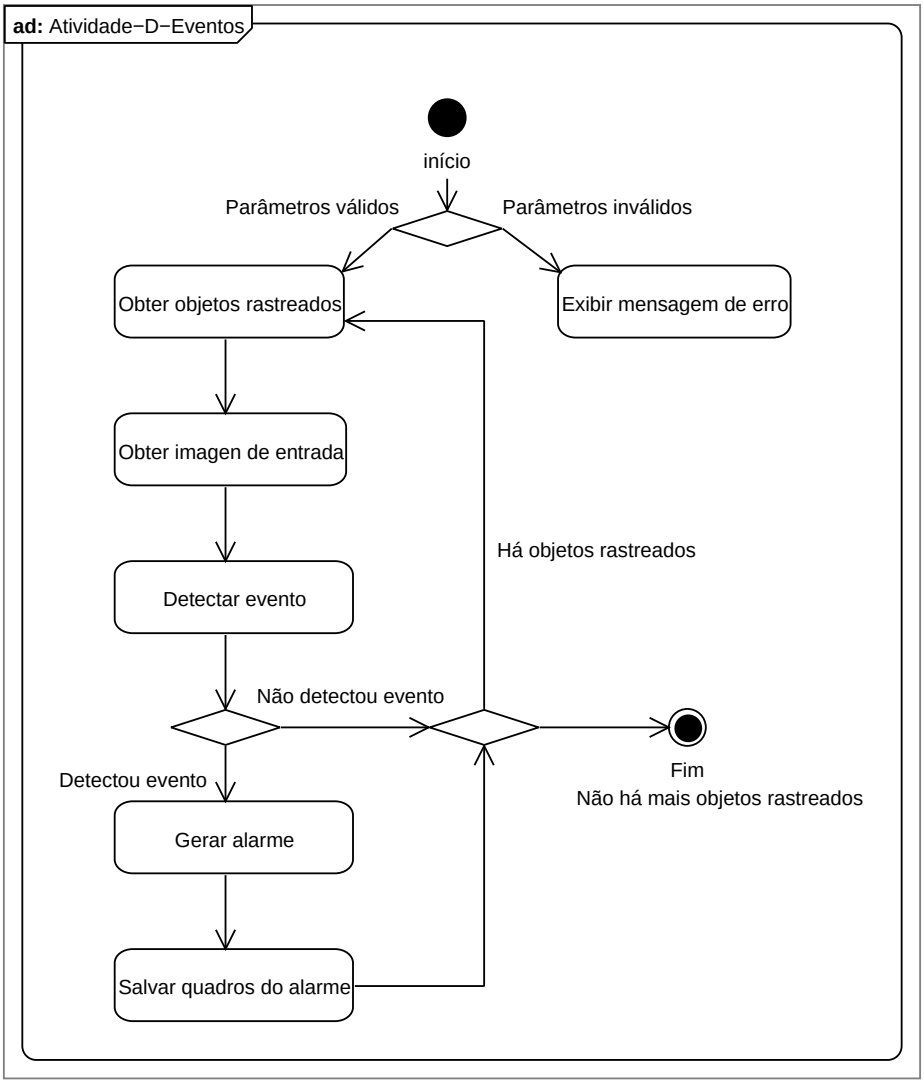


Figura 5.6: Diagrama de atividades do módulo detector de eventos.

podem ter sido gerados no quadro corrente. Posteriormente, o módulo itera sobre a lista de eventos existentes aplicando um conjunto de restrições. Para cada evento que atende ao conjunto de restrições aplicado é gerado um alarme na saída do módulo. Uma descrição mais detalhada das heurísticas empregadas na detecção de eventos é apresentada na Seção 6.4. O fluxograma do algoritmo de detecção de eventos implementado, é apresentado no Apêndice F.

Capítulo 6

Experimentos e Resultados

Neste capítulo, são apresentados e discutidos os resultados obtidos após a implementação dos principais módulos da arquitetura proposta. Inicialmente, a Seção 6.1 apresenta as bases de dados utilizadas nos experimentos e as estratégias de validação quantitativa dos resultados obtidos. Por último, as Seções 6.2, 6.3 e 6.4 apresentam, respectivamente, os resultados obtidos para cada módulo.

6.1 Bases de Dados e Avaliação Quantitativa

Os experimentos realizados neste trabalho utilizaram bases de vídeo públicas disponíveis na Internet juntamente com suas respectivas saídas desejadas, comumente denominadas conjunto verdade (*ground truth*). Uma das bases foi obtida do Projeto *CAVIAR* [1], sigla para *Context Aware Vision Using Image-based Active Recognition* em inglês. O projeto *CAVIAR* disponibiliza seus vídeos em arquivos codificados no formato *MPEG2*, ou simplesmente seqüências de quadros codificados no formato *JPEG* com resolução de 384×288 *pixels*.

Existem três tipos de cenários registrados pelos vídeos do projeto *CAVIAR*. O primeiro cenário contém imagens capturadas de cima do salão de entrada do Laboratório *INRIA*, na França. O segundo e terceiro cenários contém duas tomadas, uma frontal e outra lateral do corredor de um shopping center em Lisboa, Portugal.

A segunda base de dados utilizada foi obtida do encontro *PETS 2001* [2]. O encontro *PETS* é realizado anualmente desde o ano 2000 com o objetivo de testar diferentes aplicações em análise de movimento humano. O encontro realizado no ano de 2001 teve como tema o rastreamento de pessoas

[1] http://homepages.inf.ed.ac.uk/rbf/CAVIAR.

[2] ftp://ftp.pets.rdg.ac.uk/pub/PETS2001

e veículos em ambientes externos.

O encontro disponibilizou cinco cenários de vídeo, embora neste trabalho tenha sido utilizado apenas o primeiro cenário, por ser este o único cujo conjunto verdade está disponível. Os vídeos desta base encontram-se no formato *QuickTime* ou em arquivos *JPEG* com resolução de 768×576 *pixels*.

Para cada seqüência de imagens utilizada nos experimentos existe um ou mais arquivos do conjunto verdade, a depender de como as saídas desejadas estejam organizadas. Basicamente, existem dois tipos de conjunto verdade: os baseados em *pixel* e os baseados em objetos.

Os conjuntos verdade baseados em *pixel* são um conjunto de imagens binarizadas destacando os objetos que devem ser segmentados. Em outras palavras, é a saída perfeita de um módulo detector de movimento. Este tipo de conjunto verdade é mais utilizado em avaliações quantitativas também baseadas em *pixel* como será explicado adiante.

Os conjuntos verdade baseados em objetos consistem, geralmente, de arquivos texto ou em formato *XML* contendo informações sobre posicionamento e/ou características dos objetos de cena. Neste caso, os objetos são representados pelos retângulos mínimos nos quais estão contidos. Este tipo de conjunto verdade é mais utilizado em avaliações quantitativas que utilizam métricas baseadas em objetos. Tanto os conjuntos baseados em *pixel*, quanto os baseados em objetos, são construídos manualmente através de marcações nas imagens de entrada feitas por um operador humano.

De posse dos conjuntos verdade, pode-se então calcular métricas que avaliam quantitativamente a acurácia de cada módulo. Para o módulo de detecção de movimento existem métricas baseadas em *pixel* e métricas baseadas em objetos. O primeiro tipo é calculado pela comparação *pixel* a *pixel* entre resultados obtidos pelo modelo de segmentação testado e a saída desejada de cada vídeo.

Embora métricas calculadas *pixel* a *pixel* revelem uma visão geral da quantidade de acertos do modelo de segmentação em cada quadro de vídeo, esse tipo de métrica nada revela sobre a segmentação individual dos objetos (Lara (2007), Renno et al. (2006)). Por esse motivo, a validação dos resultados do módulo de detecção implementado neste trabalho utilizou as métricas baseadas em objetos definidas em Nascimento and Marques (2006). A forma como essas métricas são calculadas é apresentada na Subseção 6.2

Para os módulos de Rastreamento e Detecção de Eventos, foram utilizadas sequências de imagens de uma terceira base, obtida do encontro *PETS 2006* [3], cujo tema foi o abandono de bagagens em locais públicos. A base consiste de sete conjuntos de sequências de vídeos, capturadas por quatro câmeras numa estação de metrô em Londres, Inglaterra. Cada conjunto possui um grau de complexi-

[3] http://www.cvg.rdg.ac.uk/PETS2006/data.html

dade diferente. As sequências de imagens desta base estão disponíveis em arquivos no formato *JPEG* com resolução de 720 $\times$ 576 *pixels*.

6.2 Detecção de Movimento

Nesta seção, são discutidos os resultados dos modelos de segmentação testados pelo detector de movimento. Foram testados dois modelos: o modelo bimodal adaptativo *W4*, proposto por Haritaoglu et al. (2000) e o modelo de Mistura de Gaussianas [4] proposto inicialmente por Stauffer and Grimson (1999).

O modelo *W4* foi implementado seguindo sua descrição original, porém com algumas modificações na fase de classificação e na fase de adaptação às mudanças do ambiente. Para o modelo *GMM*, foi utilizada a implementação disponível na versão 1.0 da biblioteca *OpenCV*, a qual baseou-se no trabalho de Kaewtrakulpong and Bowden (2001). O Apêndice A contém detalhes do mecanismo de funcionamento desses dois modelos.

Como mencionado anteriormente, a validação dos resultados obtidos pelo módulo de deteção de movimento foi feita utilizando as métricas baseadas em objetos definidas no trabalho de Nascimento and Marques (2006). O cálculo dessas méstricas têm por base uma matriz ζ de correspondência entre os objetos segmentados pelo modelo e os objetos contidos no conjunto verdade como definido na Equação 6.1.

$$\zeta(i,j) = \begin{cases} 1 & \text{caso } \overline{R}_i \cap R_j \geq T, \\ & \forall i \in \{1,2,3,...,t\}, j \in \{1,2,3,...,v\}\,, \\ 0 & \text{caso contrário} \end{cases} \tag{6.1}$$

em que

- $\overline{R}_i$ é um objeto pertencente ao conjunto verdade;
- R_j é um objeto obtido pelo modelo;
- t e v são, respectivamente, o número de regiões de $\overline{R}$ e R;
- $\zeta(i,j)$ é a correspondência entre o objeto $\overline{R}_i$ e o objeto R_j;

De acordo com a matriz ζ, existe uma linha para cada objeto do conjunto verdade e uma coluna para cada objeto obtido pelo modelo. Caso a interseção entre um objeto $\overline{R}_i$ e outro R_j seja maior que um determinado limiar T, o valor $\zeta(i,j)$ será ajustado para 1.

[4] Deste ponto em diante do trabalho, será utilizada a sigla *GMM*, do inglês *Gaussian Mixture Model*

Visto que os conjuntos verdade utilizados neste trabalho possuem a marcação dos retângulos mínimos de cada objeto, a interseção utilizada para o preenchimento das células da matriz ζ foi feita como ilustrado na Figura 6.1. Neste trabalho foi definido o limiar mínimo de 50% necessário para relacionar um objeto do conjunto verdade a um objeto detectado.

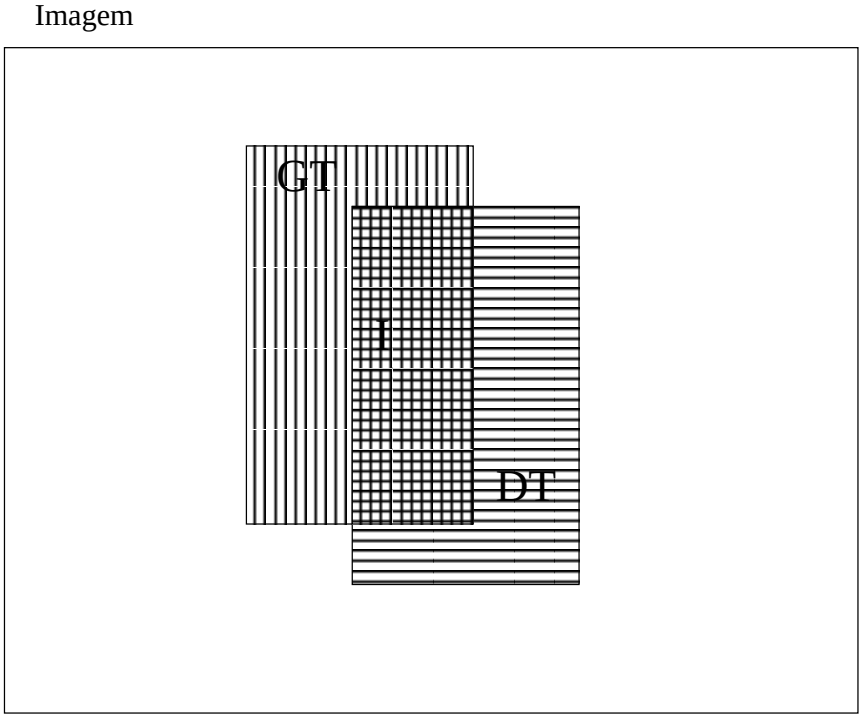


Figura 6.1: Exemplo de interseção (I) entre um objeto do conjunto verdade, representado por *GT*, e um objeto detectado pelo modelo de segmentação, representado por *DT*.

Calculada a matriz de correspondência ζ, pode-se inferir alguma semântica nas relações entre os objetos $\overline{R}_i$ e R_j. Seja $L(i)$ a soma dos elementos da matriz ζ para a linha i e $C(j)$ a soma dos elementos da matriz ζ para coluna j. De acordo com os valores de $\zeta(i,j)$, $L(i)$ e $C(j)$, uma relação entre regiões $\overline{R}_i$ e R_j pode ser classificada em um dos casos da Tabela 6.1.

Por último, ainda pode ser derivada a taxa de *detecção total* utilizada na análise quantitativa baseada em objetos. A equação 6.2 define essa métrica.

$$DR = \frac{CD + SR + JR + JSR}{t} \tag{6.2}$$

Os modelos de segmentação avaliados neste trabalho possuem um conjunto de parâmetros os quais influenciam diretamente os resultados obtidos. Com objetivo de investigar os melhores valores (dentre um conjunto de valores testados) para cada parâmetro, foi feita uma análise quantitativa baseada no conceito de curvas *ROC* (Renno et al., 2006).

Tabela 6.1: *Possiveis relações definidas por Nascimento and Marques (2006), entre objetos obtidos pelo modelo e objetos do conjunto verdade.*

Medida	**Descrição**	**Condição**
Detecção Correta (CD)	Indica que houve correspondência de 1 para 1 entre uma região obtida pelo modelo e uma região do conjunto verdade.	$\zeta(i,j) = 1, L(i) = C(j) = 1$
Ruído (N)	Não existe correspondência entre um objeto detectado pelo modelo e o conjunto verdade.	$C(j) = 0$
Não Detecção (ND)	O objeto do conjunto verdade não possui correspondente no resultado do algoritmo.	$L(j) = 0$
Região Conjunta (JR)	Um objeto do resultado corresponde a vários objetos do conjunto verdade.	$\zeta(i,j) = 1, C(j) > 1$ e $L(i) = 1$
Região Dividida (SR)	Vários objetos do resultado correspondem a um objeto do conjunto verdade.	$\zeta(i,j) = 1, C(j) = 1$ e $L(i) > 1$
Região Conjunta Dividida (JSR)	Quando os casos JR e SR ocorrem simultaneamente.	$\zeta(i,j) = 1, C(j) > 1$ e $L(i) > 1$

Uma curva ROC (*Receiver Operating Characteristic*) é um gráfico representando plota valores de sensibilidade versus especificidade de um classificador binário pela variação de seu limiar de classificação. Uma curva ROC pode ser representada ao se plotar frações de positivos verdadeiros e falsos positivos. Quando definida dessa forma, a sigla ROC é conhecida como (*Relative Operating Characteristic Curve*), pois se trata de uma comparação de duas características (positivos verdadeiros e falsos positvos) através da variação dos critérios de classificação do classificador.

Na análise quantitativa realizada neste trabalho, foram avaliadas dez variações de cada parâmetro, tanto para o modelo *W4* quanto para o *GMM*. As Tabelas 6.2 e 6.3 resumem os valores testados, os valores padrão, conforme descrição original de cada modelo, e os melhores valores obtidos pelas curvas *ROC*. Devido a grande quantidade de valores testados, e para evitar o excesso de combinações de configurações, cada configuração consistiu da variação de um parâmetro enquanto que os demais assumiram valor padrão. Uma descrição detalhada de cada parâmetro dos modelos pode ser encontrada

nos Apêndices A e B.

Para a análise quantitativa, cada modelo foi testado com dois vídeos, um em ambiente interno, da base de videos do projeto *CAVIAR*, e outro em ambiente externo, pertencente à base de dados do *PETS* 2001. Para uma comparação justa entre os modelos de segmentação, foram eliminados os 100 primeiros quadros de cada vídeo, uma vez que um dos parâmetros do modelo *W4* é a quantidade de quadros utilizados para iniciar o modelo.

Tabela 6.2: *Valores testados na análise quantitativa do modelo W4.*

Parâmetro	**Valor padrão**	**Valores testados**	**Melhores valores obtidos para ambiente interno**	**Melhores valores obtidos para ambiente externo**
n_0	50	10, 20, 30, 40, 60, 70, 80, 90, 100	60	80
n	50	10, 20, 30, 40, 60, 70, 80, 90, 100	30	40
k	0,8	0,8, 0,5, 0,6, 0,7, 0,9, 1,0, 1,1, 1,2, 1,3, 1,4	0,7	0,7
r	0,01	0,01, 0,001, 0,005, 0,02, 0,03, 0,04, 0,05, 0,06, 0,07, 0,08	0,08	0,07
t	2	3, 4, 5, 6, 7, 8, 9, 10, 11	2	3

Os gráficos da análise quantitativa dos modelos avaliados estão ilustrados no Apêndice C.

Como mencionado no ínicio desta seção, as métricas utilizadas na avaliação quantitativa deste trabalho são baseadas no conceito de curva *ROC*, mas que, por também se basearem na sensibilidade do algoritmo de detecção de objetos como um todo (e não apenas na classificação individual dos pixels), podem apresentar comportamento diferente de uma curva ROC convencional. No Apêndice D é apresentada uma outra abordagem que esboça curvas ROC representando a relação entre positivos verdadeiros versus falsos positivos (*Relative Operating Characteristic Curve*).

A análise dos parâmetros do modelo *W4*, como pode ser observado pelas Figuras no Apêndice C, sugere os seguintes comentários:

- n_0: não tem muita influência na coleta de métricas, pois atua apenas na primeira atualização do *background*;

Tabela 6.3: *Valores testados na análise quantitativa do modelo GMM.*

Parâmetro	**Valor padrão**	**Valores testados**	**Melhores valores obtidos para ambiente interno**	**Melhores valores obtidos para ambiente externo**
T	0,7	0,3, 0,4, 0,5, 0,6, 0,8, 0,9, 1,0, 1,1, 1,2	0,8	0,9
σ_{th}	2,5	1,25, 1,5, 1,75, 2,0, 2,25, 2,75, 3,0, 3,25, 3,5	3,5	3,5
w	200	50, 100, 150, 250, 300, 350, 400, 450, 500	Sem influência.	Sem influência.
ng	5	3, 4, 6, 7, 8, 9, 10, 11, 12	Sem influência.	Sem influência.
σ_0	30	0,01, 25, 26, 27, 28, 29, 31, 32, 33, 34	Sem influência.	Sem influência.
w_0	0,05	0,02, 0,03, 0,04, 0,06, 0,07, 0,08, 0,09, 0,1, 0,11, 0,12	Sem influência.	Sem influência.

- n: no vídeo testado, os melhores valores ficaram entre 20 e 40, entretanto testes com demais vídeos do projeto *CAVIAR*, os quais contém situações diversas, mostraram que objetos que se movem lentamente são rapidamente incorporados ao fundo da cena, ocasionando um aumento no número de falha de detecções. Por outro lado, valores muito altos podem retardar por demais a incorporação de objetos que parem de se mover;

- k: apresentou pouca variação entre valores de 0,5 a 0,8, mas aumentando muito a taxa de falhas de detecção para valores acima dessa faixa;

- t: comportou-se como descrito na proposição original do modelo, isto é, valores abaixo de 2 diminuem a restrição da equação de classificação do modelo e aumentam a taxa de ruído. Valores acima de 2 aumentam a restrição e classificação, aumentando a taxa de falhas de detecção.

- r: em geral, valores altos de r (entre 0,5 e 1) retardam a incorporação de objetos ao fundo da cena, deixando rastros de falsos positivos nas imagens da saída. Os melhores valores desse parâmetro podem depender do tipo de vídeo utilizado, mas geralmente encontram-se entre 0 e

0,3.

De maneira análoga para o modelo *GMM* do *OpenCV*, tem-se os seguintes comentários:

- T: valores baixos de T, entre (0,3 e 0,5) tendem a aumentar o número de falsos positivos, pois diminuem a restrição da etapa de classificação. Por outro lado, valores próximos a 100% aumentam o número de falhas de detecção por deixar a equação mais restritiva.

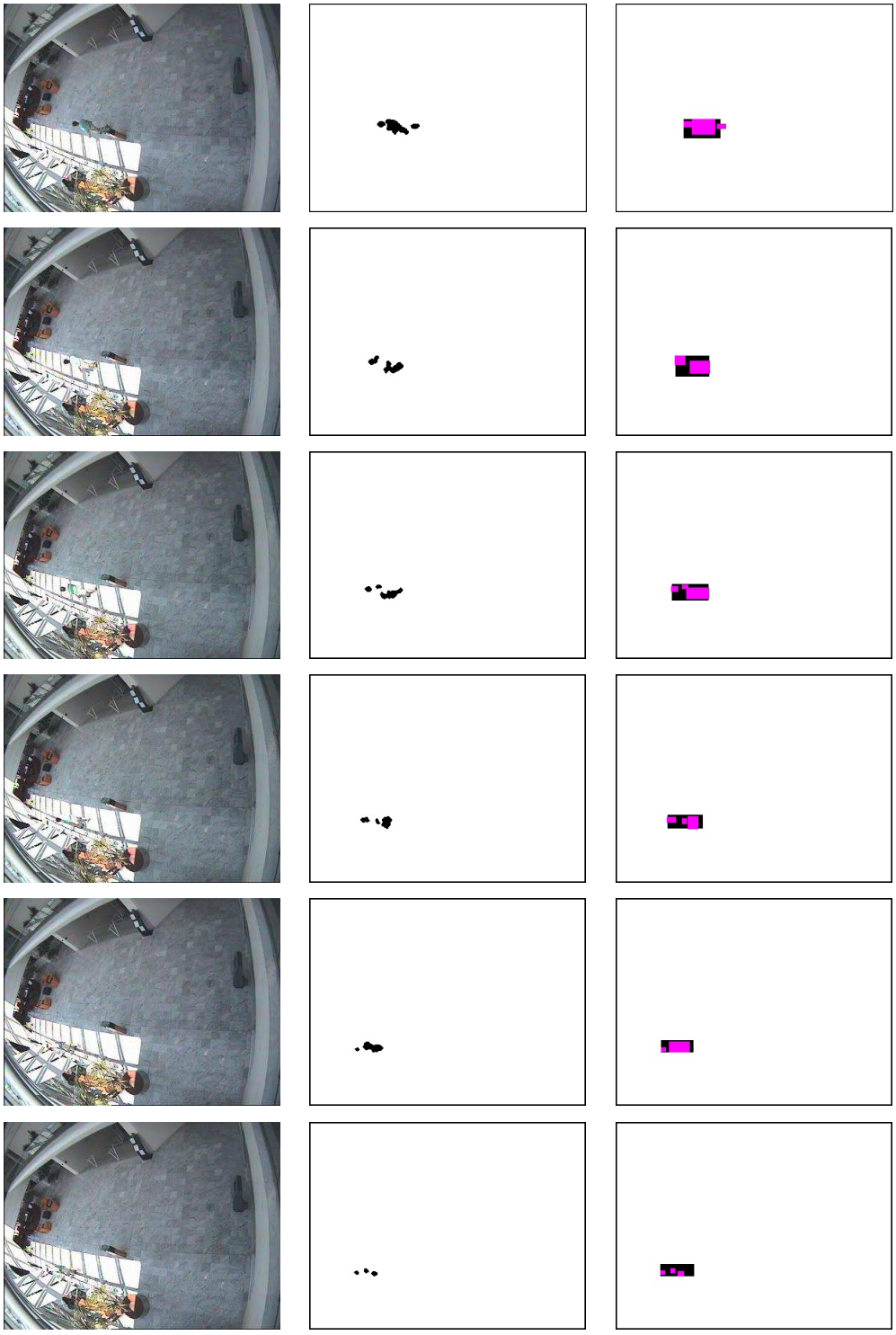
- σ_{th}: valores mais baixos desse parâmetro (abaixo de 2,5) fazem com que mais Gaussianas passem a representar o *background* da cena, levando a baixas taxas de detecção correta, enquanto que valores mais altos (acima de 3) aumentam a precisão do modelo e diminuem o número de falsos positivos.

- ng: surpreendendemente, esse parâmetro em nada influenciou nos resultados da análise em ambiente interno e muito pouco em ambiente externo. Esperava-se a não variação dos resultados em vídeo interno pela pequena faixa de valores assumidos pelos pixels nas imagens, visto que se trata de um ambiente mais controlado. Entretanto, após a análise com vídeo em ambiente externo, constatou-se que a não influência dos resultados pela variação do número de Gaussianas se deve a maneira como os parâmetros destas funções são iniciados: uma das funções Gaussianas é iniciada com peso 1 e média igual ao valor do pixel da primeira imagem recebida, enquanto as demais são iniciadas com todos os parâmetros iguais a zero. O modelo original proposto por Stauffer and Grimson (1999) sugere o uso de algoritmos de *Expectation Maximization*.

Nas Tabelas 6.4 e 6.5, são apresentados comparativos entre as métricas calculadas para os modelos com os melhores parâmetros obtidos pela análise quantitativa.

Com relação às taxas de detecção, ambos os algoritmos apresentaram resultados satisfatórios, quando comparados às taxas de detecção total apresentadas por Lara (2007). O destaque da Tabela 6.4 são os altos valores de ruído apresentados pelo modelo *W4*. Boa parte desse ruído é ocasionado pelos falsos positivos gerados pelo algoritmo de detecção de sombras implementado. Esses falsos positivos fragmentaram os objetos alvo do vídeo testado de maneira que alguns dos fragmentos resultantes não conseguiram alcançar o percentual mínimo de interseção (50%) exigido pelos testes. Dessa forma, os fragmentos são contabilizados como ruído e não como região dividida. Na Figura 6.2 estão ilustrados alguns desses fragmentos.

Na Tabela 6.5, nota-se uma diminuição dos valores de detecção correta e um aumento considerável nas taxas de região conjunta. Esse comportamento pode ser explicado pela presença no vídeo de



(a) (b) (c)

Figura 6.2: Exemplos de segmentação fragmentada pelo modelo *W4*. (a) Imagem de entrada. (b) Imagem segmentada. (c) Comparativo entre conjunto verdade (cor preta) e objetos segmentados (cor vermelha).

Tabela 6.4: *Comparativo entre os modelos de segmentação testados em ambientes interno.*

Métrica	***W4* (%)**	**GMM (%)**
Detecção Correta	90,51	94,72
Falha de Detecção	3,50	1,84
Região Dividida	1,09	0,72
Região Conjunta	1,43	1,36
Região Conjunta e Dividida	3,47	1,36
Detecção Total	96,50	98,16
Ruído	39,22	11,13

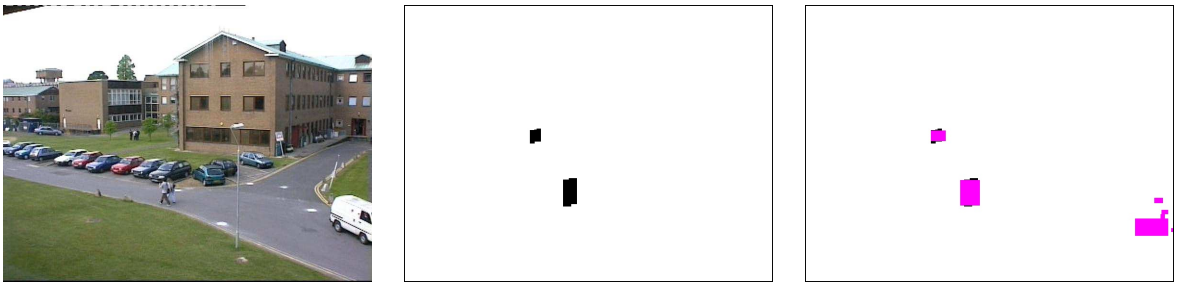
Tabela 6.5: *Comparativo entre os modelos de segmentação testados em ambientes externo.*

Métrica	***W4* (%)**	**GMM (%)**
Detecção Correta	63,10	60,00
Falha de Detecção	6,65	10,91
Região Dividida	0,8	0,01
Região Conjunta	25,59	28,36
Região Conjunta e Dividida	3,86	0,11
Detecção Total	93,35	89,09
Ruído	61,74	38,78

grupos de pessoas que se movimentam muito próximas umas as outras, fazendo com que o modelo utilizado agrupasse os *pixels* em apenas um único objeto (Ver Figura 6.3).

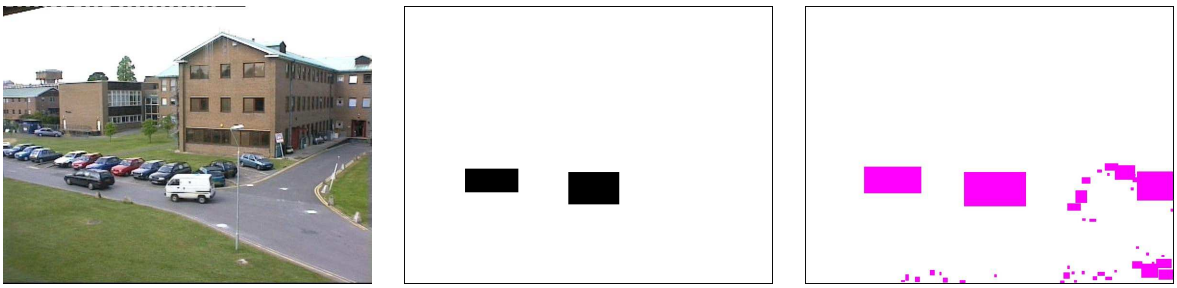
Outro destaque da Tabela 6.5 é o aumento excessivo das taxas de ruído em relação à tabela anterior. Além dos problemas já citados para o ambiente interno, a maior variação da faixa de valores assumidos pelos pixels, devido a variação de iluminação, faz com que pequenos falsos positivos ocorram durante a fase de classificação em ambientes externos. Falsos positivos também ocorrem quando um objeto é fragmentado por não ter sido totalmente incorporado pelo modelo ao *background*. A Figura 6.4 contém exemplos de ruído produzido na fase de classificação. A coluna (c) da Figura 6.4 exibe um comparativo entre os objetos do conjunto verdade (na cor preta) e os objetos da imagem segmentada (cor vermelha).

Como mencionado na Seção 2.2, existem algumas técnicas de processamento pós-segmentação que objetivam a remoção de ruído remanescente da fase de segmentação de movimento. A mais simples dessas técnicas consiste na remoção de regiões consideradas muito pequenas em relação à



(a) (b) (c)

Figura 6.3: Exemplo de segmentação com região conjunta. (a) Imagem de entrada. (b) conjunto verdade. (c) Comparativo entre conjunto verdade (cor preta) e imagem segmentada (cor vermelha).



(a) (b) (c)

Figura 6.4: Exemplo de segmentação com ruído. (a) Imagem de entrada. (b) Conjunto verdade. (c) Comparativo entre os objetos do conjunto verdade (cor preta) e objetos da imagem segmentada (cor vermelha).

maioria dos objetos alvo presentes na aplicação.

Quanto ao ruído provocado por falsos positivos de métodos de remoção de sombras, pode-se utilizar operações de erosão e dilatação com objetivo de unir fragmentos de objetos. No entanto, a abordagem mais correta para o problema é a busca por métodos de remoção de sombras que minimizem esse tipo de problema.

Além dos testes feitos com vídeo escolhido para análise quantitativa, foram feitos testes com outros 14 vídeos do primeiro conjunto de vídeos do projeto *CAVIAR*. Os vídeos testados incluem desde vídeos mais simples, com uma única pessoa transitando pelo ambiente, até vídeos contendo situações mais complexas incluindo encontro de grupos de pessoas, simulações de briga e abandono de objetos. As Figuras 6.5 e 6.6 apresentam gráficos do tipo Box-plot que resumem as métricas calculadas sobre os demais vídeos para os modelos *W4* e *GMM*.

O gráfico Box-plot (Figuras 6.5 e 6.6) foi escolhido por apresentar várias informações em um único gráfico. Nos gráficos abaixo são apresentados, além do valor mediano de cada métrica (representado pelo traço no inteiror dos retângulos), os valores mínimos (extremidade inferior do gráfico), máximos (extremidade superior do gráfico), primeiro e terceiro quartis (extremidades inferior e superior dos retângulos).

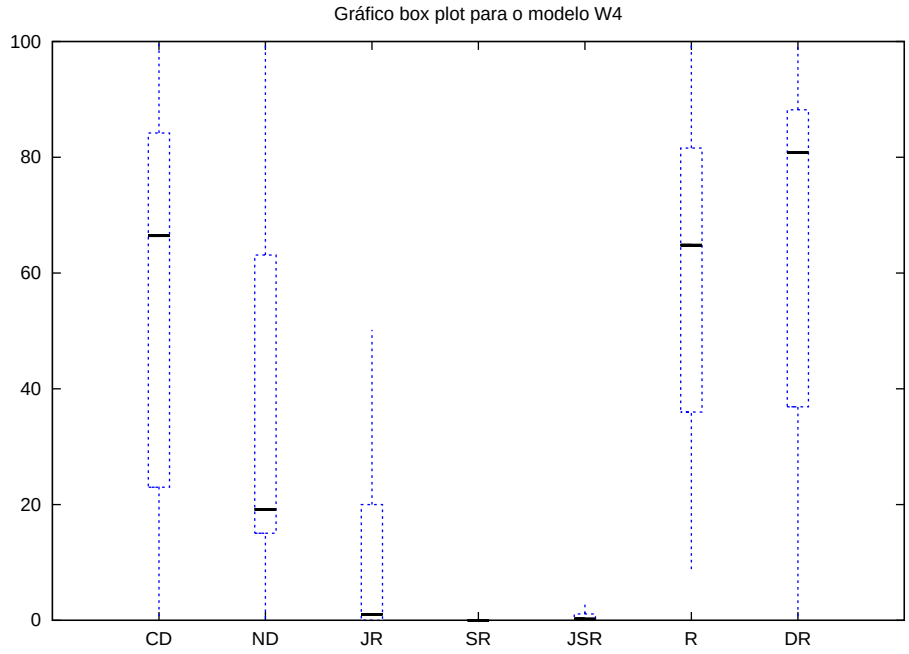


Figura 6.5: Gráfico Box-plot com porcentagem (eixo vertical) das métricas coletadas (eixo horizontal) *W4*.

Nos gráficos acima destacam-se as grandes variações nas taxas de detecção correta e ruído. Essa

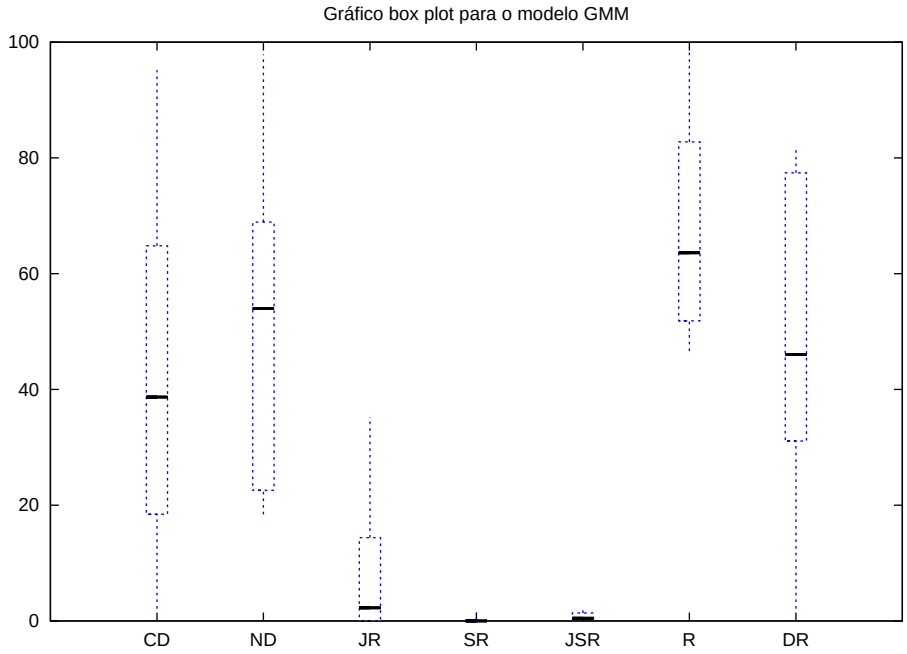


Figura 6.6: Gráfico Box-plot com porcentagem (eixo vertical) das métricas coletadas (eixo horizontal) para o modelo *GMM*.

variação excessiva deve-se, essencialmente a dois fatores. O mais evidente é que vídeos contendo situações diferentes, como os aqui testados, requerem diferentes conjuntos de parâmetros. Em outras palavras, a configuração obtida através da análise quantitativa previamente realizada, obteve altas taxas de detecção correta e baixas taxas de ruído apenas nos vídeos semelhantes àquele utilizado na análise quantitativa.

Outro fator que influencia o cálculo das taxas de detecção correta e falha de detecção é a interpretação da segmentação dada no momento de criação dos conjuntos verdade de cada vídeo. Diferentes bases de vídeo podem gerar diferentes interpretações.

No caso dos vídeos disponibilizados pelo projeto *CAVIAR*, objetos que pararam de se mover, continuam sendo segmentados, não importando o tempo que esses objetos permaneçam parados. Dessa forma, objetos que são incorporados por modelos adaptativos de segmentação (como os testados neste trabalho), passam a ser contabilizados como falhas de detecção de acordo com os conjuntos verdade. Este tipo de interpretação não ocorre nos conjuntos verdade disponibilizados pelo PETS 2001. De maneira contrária, objetos que parem de se mover, só deverão ser novamente segmentados nos casos em que esses mesmos objetos retomem a movimentação.

De maneira geral, ambos os modelos de segmentação apresentaram resultados satisfatórios se comparados aos trabalhos de Lara (2007) e Nascimento and Marques (2006). Entretanto, a principal

limitação no uso de modelos de segmentação adaptativos diz respeito ao momento em que as adaptações são feitas. Adaptações muito rápidas podem prejudicar etapas posteriores, como rastreamento e detecção de eventos, enquanto que adaptações tardias podem gerar um número excessivo de objetos em cena.

Por ter um modelo baseado no valor mediano assumido por cada *pixel*, o modelo *W4* necessita armazenar todos os valores assumidos pelos *pixels* das imagens até que o *n*-ésimo quadro seja alcançado. No momento do processo de atualização, esses valores são ordenados para que seja encontrada a nova mediana de cada *pixel*. Todo esse processo tem como consequência, maior consumo de memória e tempo de processamento, se comparado com modelos como o *GMM*, os quais atualizam o modelo de *background* a cada quadro.

Uma alternativa aos métodos que atualizam o modelo a cada n quadros é considerar a taxa de quadros por segundo na qual o vídeo foi ou está sendo gerado, pois para altas taxas de quadro por segundo deve-se ter maiores valores de n, a fim de que haja mudança considerável nos valores dos pixels até o momento da atualização do *background*. Baixas taxas de quadros por segundo sugerem atualizações de *background* mais frequentes, pois as diferenças de posicionamento dos objetos entre dois quadros consecutivos são maiores.

Outra maneira de obter melhoramento nos resultados produzidos pelos métodos de segmentação baseados em estatísticas extraídas de valores de *pixels* é através do refinamento desses resultados através do uso de técnicas de processamento espacial de imagem como detecção das bordas dos objetos ou técnicas de morfologia matemática como as utilizadas por Lara (2007). Vale lembrar que níveis adicionais de processamento diminuem o desempenho dos algoritmos utilizados, pois há necessidade de mais recursos de tempo de processamento e memória.

6.3 Rastreamento

Nesta seção, são apresentados e discutidos os resultados dos algoritmos de rastreamento testados. Foram implementados dois algoritmos: o primeiro utiliza apenas similaridade/dissimilaridade de características de objetos. Ao segundo algoritmo, além da correspondência de características, foi adicionada a cada objeto rastreado uma instância de um Filtro de Kalman com objetivo de verificar sua influência nos resultados, principalmente em relação ao tratamento de oclusão. No Apêndice E econtram-se os fluxogramas dos dois algoritmos.

A estratégia de correspondência entre os objetos rastreados e aqueles segmentados no quadro corrente é baseada no trabalho de Amer (2005). O vetor de características utilizado é composto pelas

mesmas características citadas na Tabela 3.1 do Capítulo 3.

O módulo de rastreamento possui duas listas de objetos, uma contendo objetos rastreados e a outra com os objetos segmentados no quadro corrente. No algoritmo de correspondência por similaridade, essas duas listas de objetos dão origem a um grafo bipartido. O algoritmo itera sobre os vértices da partição que contém os objetos rastreados aplicando um esquema de votação que verifica a similaridade/dissimilaridade entre os candidatos a associação (vértices da outra partição). Caso os objetos sejam similares, uma aresta entre os vértices é criada.

Ao término das iterações, o algoritmo aplica sucessivas verificações com objetivo de detectar casos de oclusão (*occlusion* ou *merge*) e divisão (*split*) de objetos entre quadros consecutivos. No Apêndice E, são apresentados detalhes dessas verificações. Ao final, as propriedades (estado, deslocamento, vetor de características, etc.) de cada objeto são atualizadas de acordo com as associações previamente estabelecidas. Objetos do quadro corrente que, ao final das verificações, não possuem arestas de correspondência recebem estado de *Novo*, isto é, acabaram de entrar em cena. Objetos previamente rastreados que não possuam correspondência alguma com objetos do quadro corrente (exceto por oclusão) recebem o estado de *Perdido*, cabendo ao módulo de rastreamento eliminá-los ou não. Na Tabela E.1 do Apêndice E estão listados os possíveis estados assumidos por um objeto durante o rastreamento.

O segundo algoritmo de rastreamento implementado (com Filtro de Kalman) possui estratégia de correspondência entre objetos semelhante às do primeiro algoritmo, porém com algumas diferenças. Antes de cada iteração para associação entre objetos das duas listas, o algoritmo projeta a posição do centro de massa de cada objeto rastreado para o quadro corrente utilizando as equações de predição do Filtro de Kalman. As posições projetadas no quadro atual são utilizadas nas restrições de associação de objetos.

A maior diferença em relação ao primeiro algoritmo está na etapa de atualização dos objetos. Para objetos normalmente rastreados (os que possuem correspondência), o segundo algoritmo utiliza a posição real do objeto no quadro corrente para aplicação das equações de correção do Filtro de kalman. Para objetos em oclusão, utiliza-se a projeção da posição feita previamente. O primeiro algoritmo limitou-se apenas à detecção da oclusão, isto é, ao detectá-la, o algoritmo continua o rastreamento apenas com o objeto resultante da junção dos objetos previamente rastreados. Na Tabela 6.6, apresenta-se um resumo das capacidades de cada algoritmo.

Com objetivo de depurar os algoritmos implementados, antes de realizar testes com algumas das bases públicas citadas no inicio do capítulo, foram realizados testes com sequências de imagens sintéticas contendo figuras geométricas que se deslocam ao longo dos quadros. Na Figura 6.7 estão

Tabela 6.6: *Estados de um objeto rastreado.*

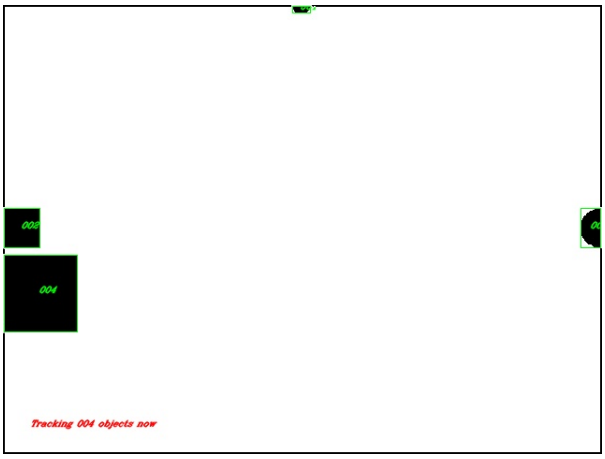
Algoritmo	**Estratégia**	**Detecção de oclusão/junção**	**Tratamento de oclusão/junção**	**Detecção de divisão**
Algoritmo 1	Similaridade de vetor características	Sim	Não. Atribui ao objeto resultante o rótulo do maior objeto em oclusão	Sim
Algoritmo 2	Similaridade de vetor de características e Filtro de Kalman	Sim	Sim. Utiliza a predição da posição	Sim

ilustrados alguns resultados obtidos por esses testes preliminares.

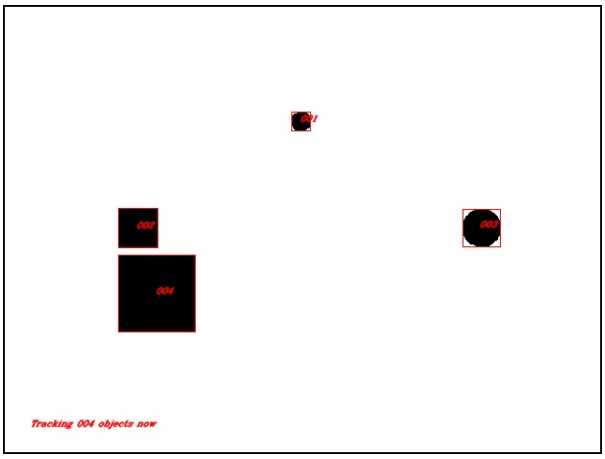
Como mencionado na Seção 6.1, os testes para os módulos de rastreamento e detecção de eventos utilizaram a base de dados do encontro *PETS* 2006. Para o módulo de rastreamento foi utilizada a sequência de imagens da câmera 3 do conjunto de dados 1. Para cada algoritmo foram escolhidos alguns objetos da sequência com objetivo de verificar a eficácia do rastreamento. Ambos os algoritmos utilizaram como entrada os objetos segmentados pelo modelo *W4* combinado com algoritmo de detecção de sombras proposto por Jacques et al. (2005).

Na Tabela 6.7, apresentam-se os resultados obtidos no rastreamento de seis objetos com baixo grau de complexidade para rastreamento. A coluna **Quadros** informa os números dos quadros inicial e final de cada objeto na sequência utilizada. A coluna **Num. de Rótulos** indica a quantidade de rótulos diferentes assumidos pelo objeto ao longo do rastreamento. A coluna **Quadros Rastreados** indica a sequência mais longa de quadros rastreados com um mesmo rótulo. As duas últimas colunas da tabela informam os números de **Divisões** e **Junções** detectadas ao longo da sequência avaliada.

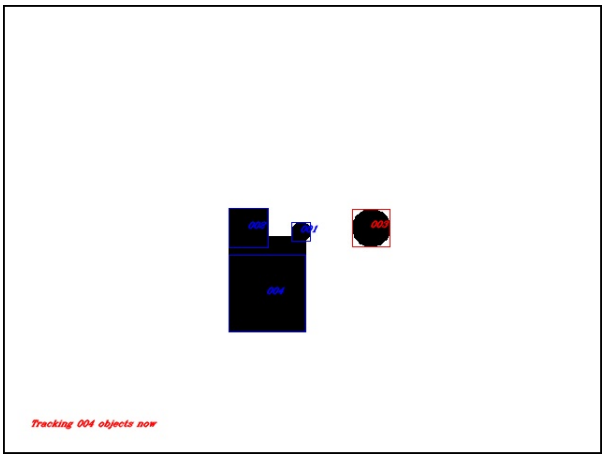
Na Tabela 6.7, observam-se bons resultados considerando o baixo número de rótulos assumidos por cada objeto durante o experimento. Embora eficiente com relação à detecção de sombras, os algoritmos de segmentação utilizados apresentam considerável segmentação interna dos objetos, causando excessivo número (Objetos 4 e 5) de divisões. No entanto, o número de rótulos assumidos pelos objetos permaneceu baixo devido à estratégia de atribuir o rótulo do objeto dividido ao objeto filho com maior tamanho (ver Figura 6.9). A mesma estratégia foi adotada em caso de junção, isto é, quando dois objetos diferentes de juntam, permanece o rótulo do objeto com maior tamanho.



(a)



(b)



(c)

Figura 6.7: Resultados obtidos de imagens sintéticas. (a) Objetos em estado *Novo* na cor verde. (b) Objetos em estado *Rastreando* na cor vermelha. (c) Objetos em estado *Oclusão* em azul.

Tabela 6.7: *Resultados obtidos pelo Algoritmo 1.*

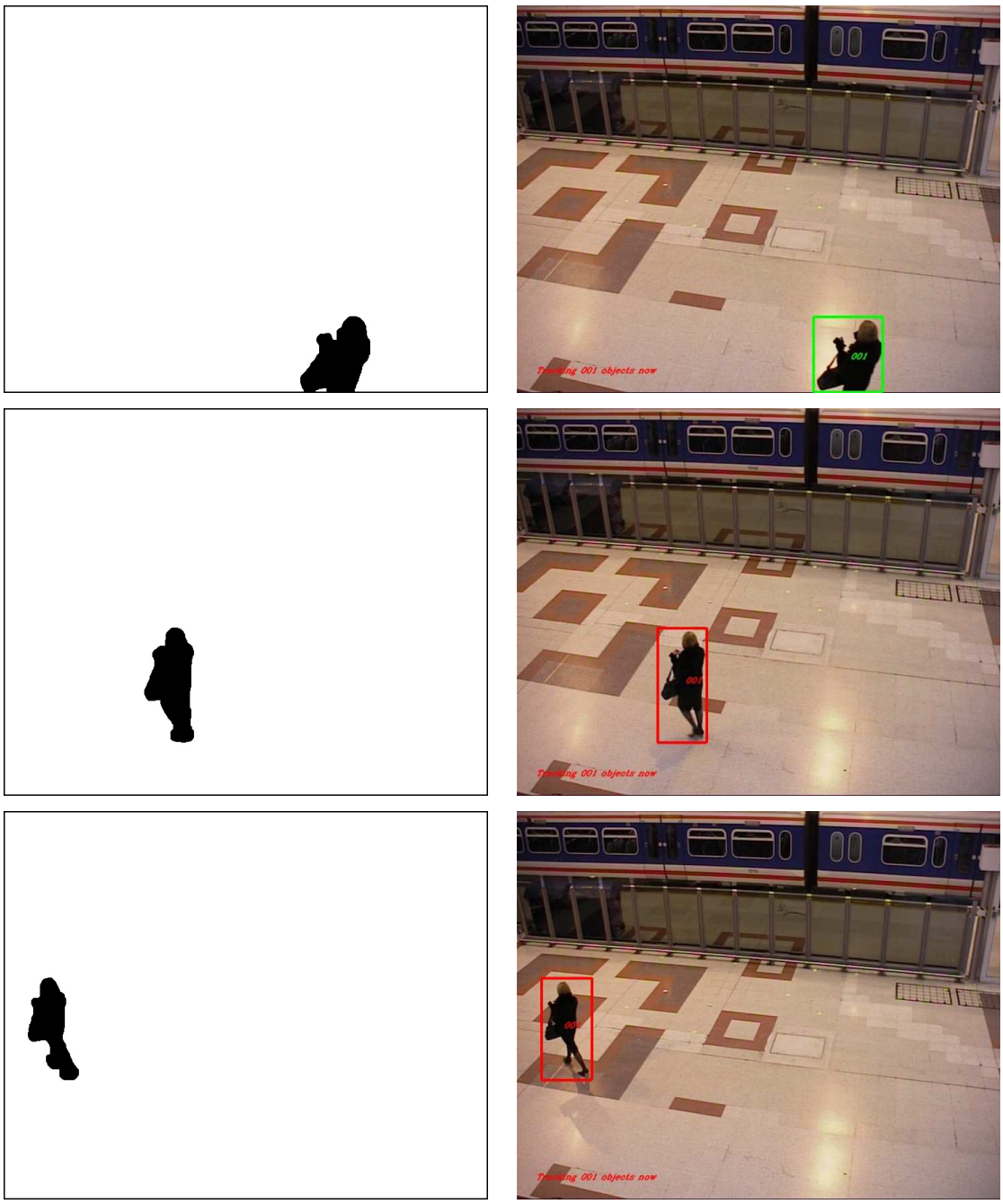
Objeto	**Quadros**	**Núm. de rótulos**	**Quadros rastreados**	**Divisões**	**Junções**
Objeto 1	195	1	195	0	0
Objeto 2	132	1	132	2	2
Objeto 3	110	1	110	1	0
Objeto 4	153	1	153	8	4
Objeto 5	1140	2	671	27	15
Objeto 6	167	1	167	0	0
Total	1897	7	1428	38	21

Há casos, no entanto, em que a estratégia adotada pelo Algoritmo 1 falha, pois quando a segmentação apresenta sucessivas junções e divisões de objetos diferentes mas com tamanhos parecidos, o algoritmo tende a trocar os rótulos de objetos diferentes, perdendo assim o objetivo principal do rastreamento, que é atribuir o mesmo rótulo ao mesmo objeto ao longo de uma sequência de quadros. As Figuras 6.8, 6.9, 6.10, 6.11 apresentam alguns dos quadros resultantes do rastreamento dos objetos da Tabela 6.7.

Na Tabela 6.8, apresentam-se os resultados obtidos no rastreamento de cinco objetos com complexidade de rastreamento maior do que aqueles apresentados na Tabela 6.7, pois envolvem situações de oclusão em que pessoas caminham lado a lado (objetos que se deslocam no mesmo sentido) e pessoas passando umas pelas outras (objetos que se deslocam em sentidos opostos). Estes objetos foram utilizados nos testes com o Algoritmo 2.

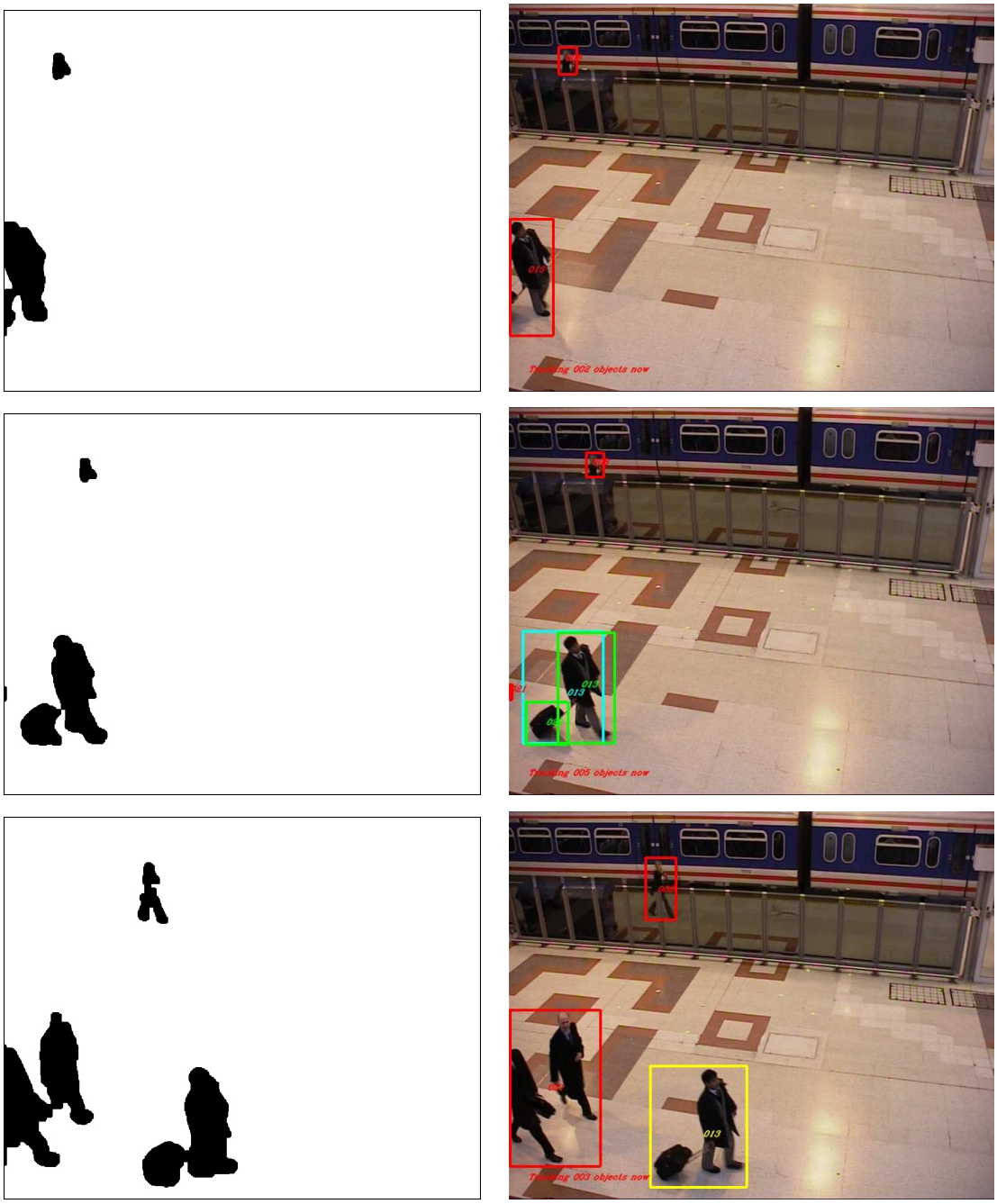
Na Tabela 6.8, a coluna **Quadros** contém os números dos quadros inicial e final de cada objeto na sequência utilizada. Na coluna **Num. de rótulos**, é indicada a quantidade rótulos diferentes assumidos pelo objeto ao longo do rastreamento. Na coluna **Oclusões**, indica-se a quantidade de vezes em que o objeto entra em oclusão com outros objetos. Na coluna **Oclusões detectadas**, indicam-se quantas vezes o algoritmo detectou oclusão do objeto experimentado com outros da cena. Por último, na coluna **Rec. de Rótulo**, são indicadas quantas vezes o algoritmo conseguiu recuperar um objeto em estado de oclusão, isto é, passar do estado *Oclusão* para o estado *Rastreando*.

Na Tabela 6.8, há um considerável aumento no número de rótulos assumidos. Notadamente em situações nas quais pessoas caminham lado a lado e permanecem por um longo período em estado de oclusão. Nos experimentos realizados, configurou-se uma tolerância máxima (de 15 quadros) para objetos em oclusão. Ao atingir a tolerância máxima sem recuperação, um objeto é removido em definitivo da lista de objetos rastreados. O algoritmo no entanto, apresenta melhor resultado em



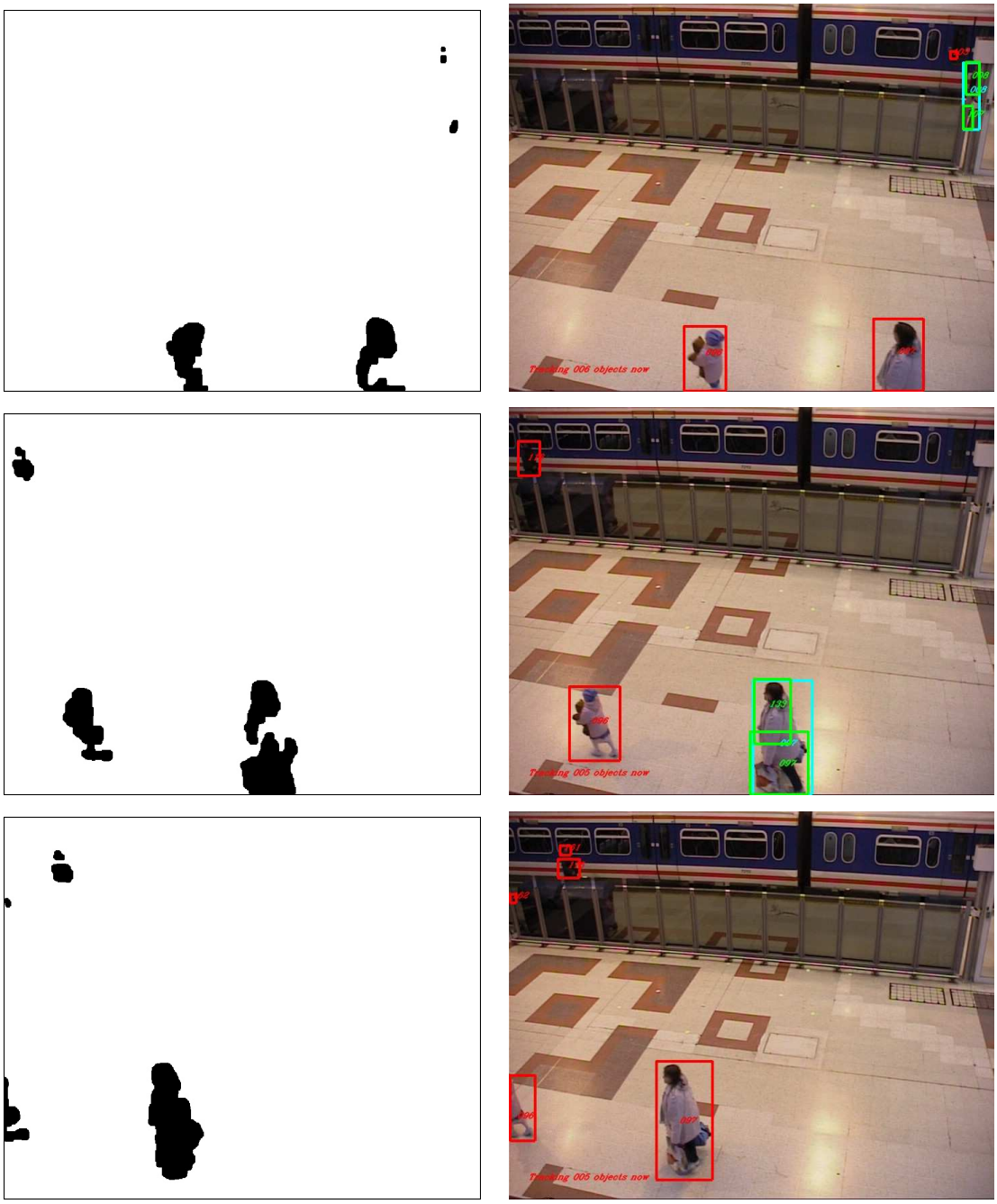
(a) (b)

Figura 6.8: Resultados obtidos pelo Algoritmo 1 para o Objeto 1. (a) Saída do módulo detector de movimento. (b) Objeto 1 rastreado, em estado *Novo* (verde) e *Rastreando* (vermelho).



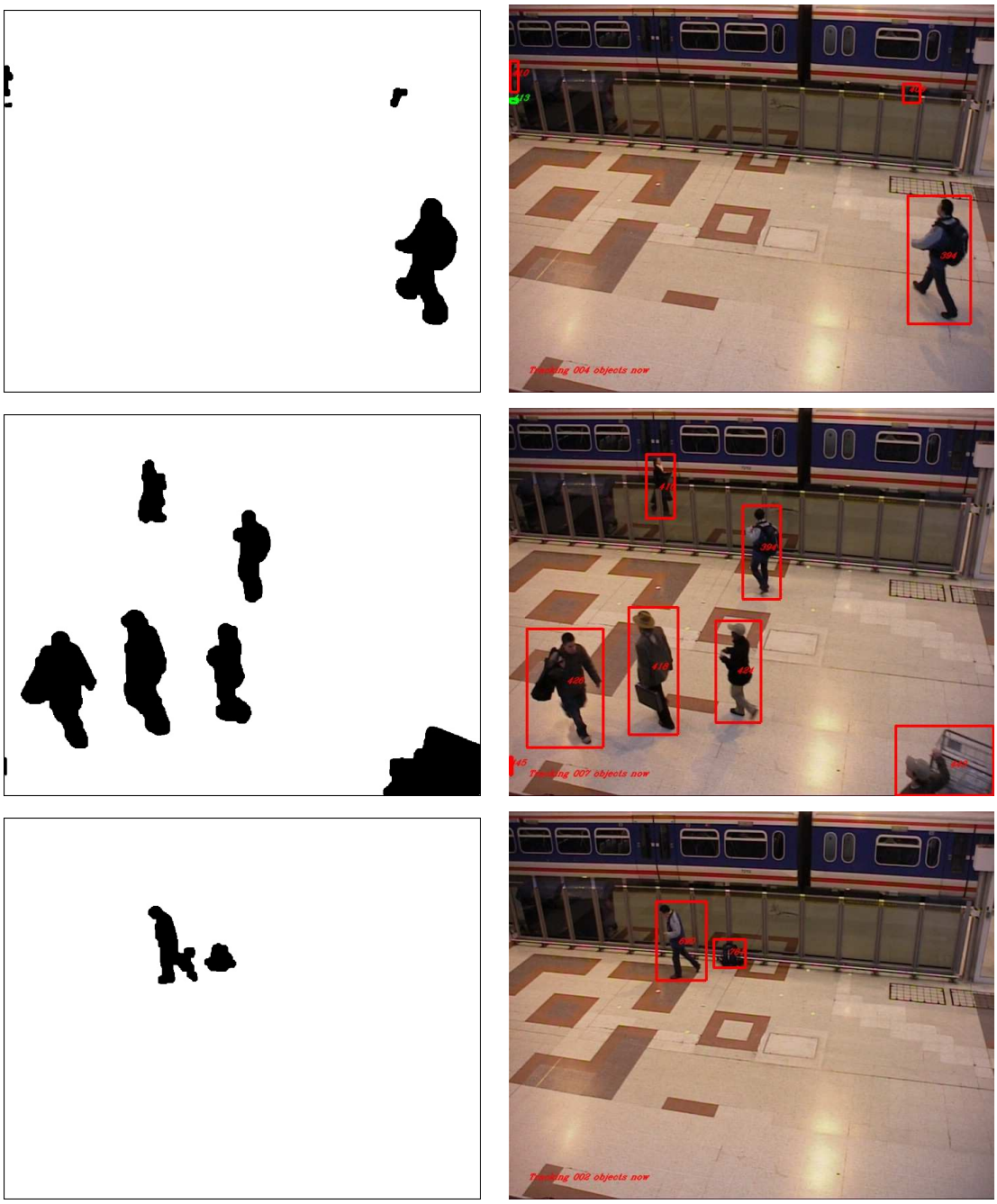
(a) (b)

Figura 6.9: Resultados obtidos pelo Algoritmo 1 para o Objeto 2. (a) Saída do módulo detector de movimento. (b) Objeto 2 (identificador 013) rastreado, em estado *Reastreando* (vermelho), *Divisão* (azul claro) e *Junção* (em amarelo).



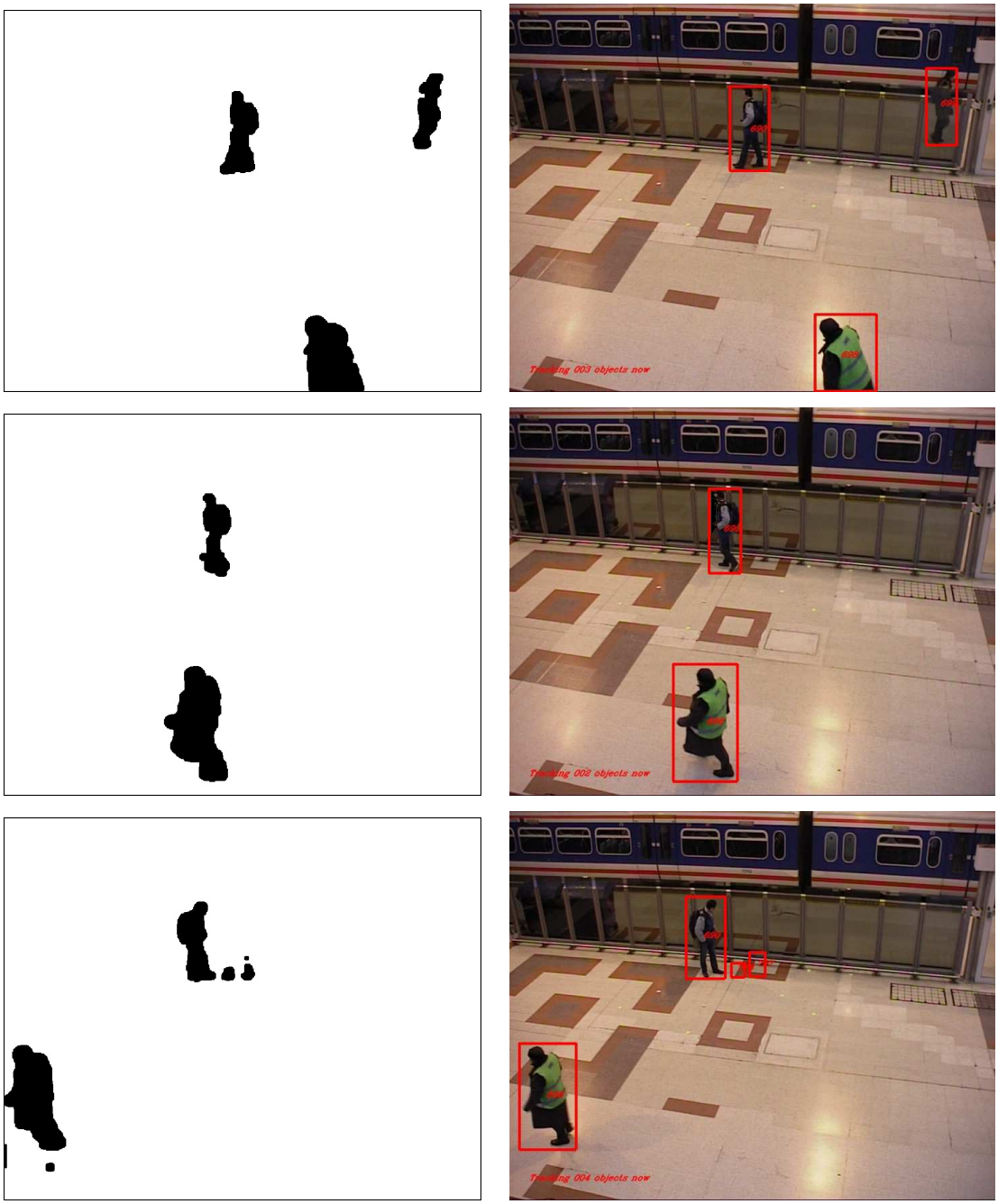
(a) (b)

Figura 6.10: Resultados obtidos pelo Algoritmo 1 para os Objetos 3 e 4. (a) Saída do módulo detector de movimento. (b) Objeto 3 (identificador 096) e 4 (identificador 97) em estado *Rastreando* (vermelho), *Divisão* (azul claro) e *Junção* (em amarelo).



(a) (b)

Figura 6.11: Resultados obtidos pelo Algoritmo 1 para o Objeto 5. (a) Saída do módulo detector de movimento. (b) Objeto 5 (identificadores 394 e 690) em estado *Rastreando* (vermelho).



(a) (b)

Figura 6.12: Resultados obtidos pelo Algoritmo 1 para o Objeto 6. (a) Saída do módulo detector de movimento. (b) Objeto 5 (identificador 698) em estado *Rastreando* (vermelho).

Tabela 6.8: *Resultados obtidos pelo algoritmo 2.*

Objeto	Quadros	Núm. de rótulos	Oclusões/Junções	Oclusões detectadas	Rec. de Rótulo
Objeto 7	275-395	4	7	7	4
Objeto 8	985-1100	3	1	1	1
Objeto 9	1835-1955	2	2	2	1
Objeto 10	2230-2330	1	1	1	1
Objeto 11	2730-2825	2	2	2	0

objetos que se deslocam em sentidos opostos, pois apresentam pequeno período de oclusão.

Outro problema que impede a recuperação de objetos em oclusão é o fato de o algoritmo não fazer predição (ou atualização) do vetor de características durante o período em que um objeto se encontra no estado de oclusão, pois o algoritmo realiza predição apenas do retângulo mínimo de cada objeto. Consequentemente, as propriedades de forma e tamanho do objeto após o estado de oclusão podem ser muito diferentes daquelas extraídas antes da oclusão. Algumas tentativas de atualização do vetor de características através da predição do retângulo mínimo dos objetos não obtiveram êxito. As Figuras 6.13, 6.14, 6.15 e 6.16 apresentam alguns dos quadros resultantes do rastreamento dos objetos da Tabela 6.8.

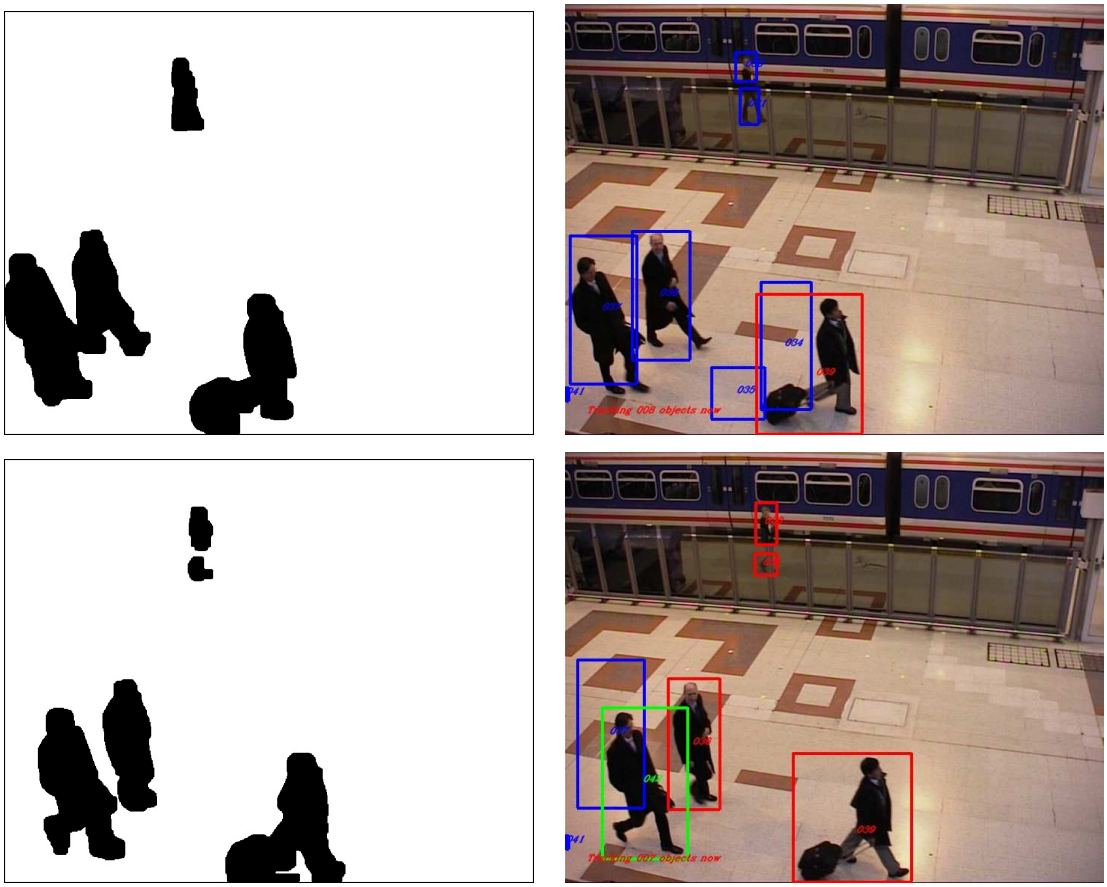
6.4 Detecção de Eventos

Como explicado na Seção 5.2, o módulo de detecção de eventos itera sobre a lista de eventos existentes aplicando um conjunto simples de restrições para identificar se o evento corresponde à remoção ou abandono de objetos. Baseado no trabalho de Dedeoglu (2004), e nos trabalhos apresentados no encontro *PETS* 2006 definiu-se que, um objeto que representa uma pessoa, abandona (ou remove) um objeto na cena quando:

1. Um objeto sofre divisão, resultando em um objeto maior (pessoa) e outro(s) menores (removidos ou abandonados);

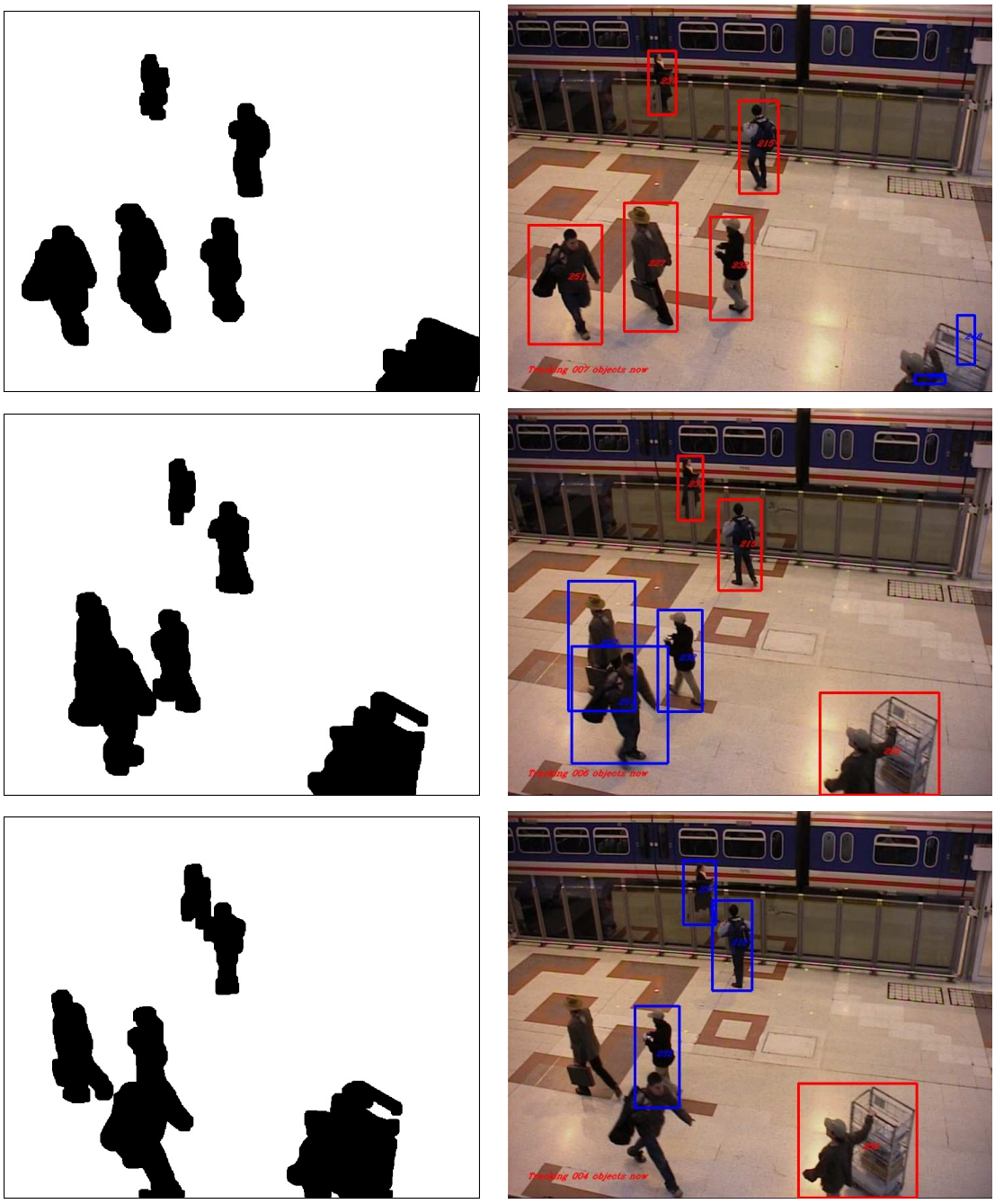
2. Um dos objetos resultantes da divisão permanece imóvel;

3. Um dos objetos resultantes da divisão permanece a uma distância maior que uma distância mínima previamente definida;



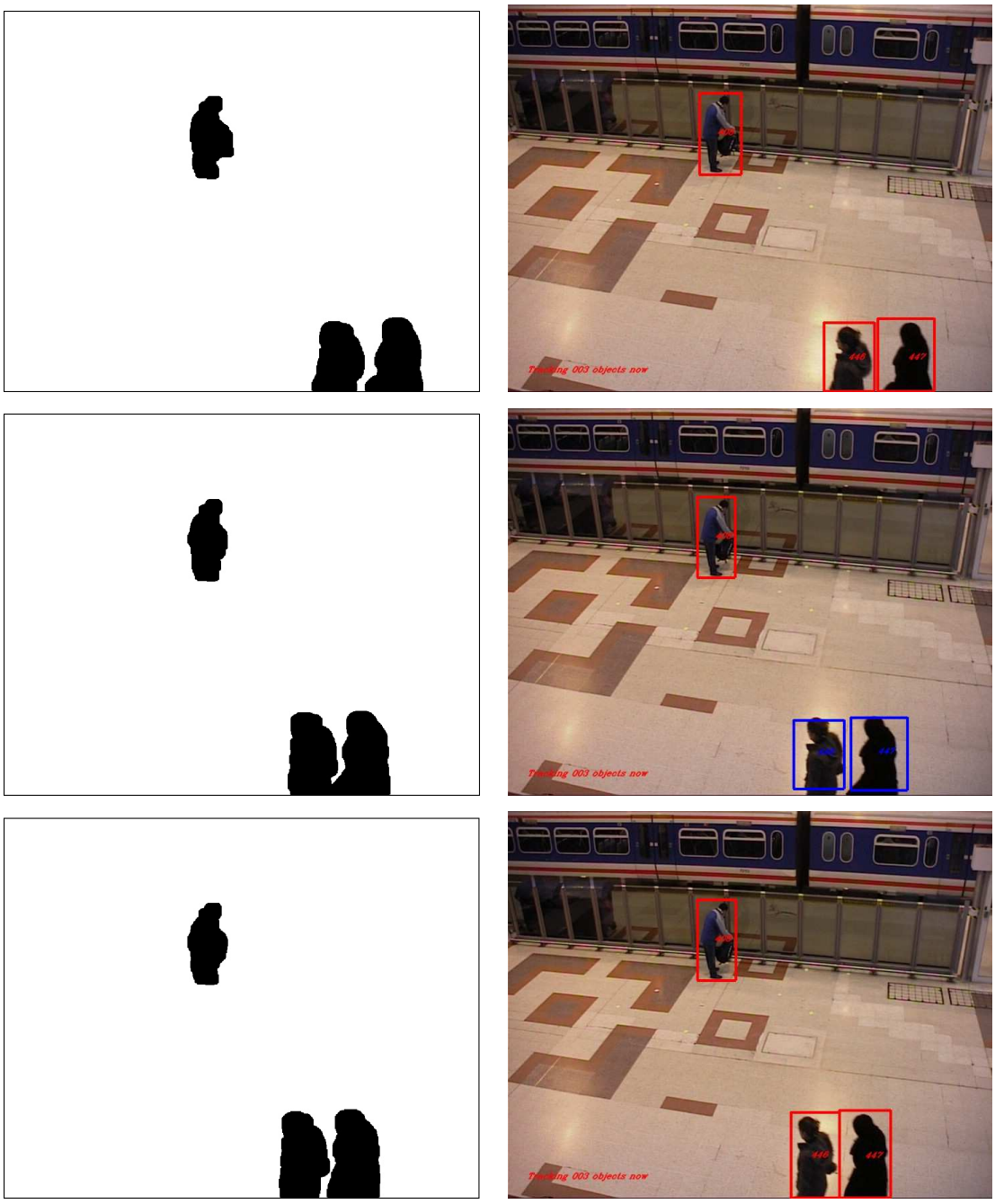
(a) (b)

Figura 6.13: Resultados obtidos pelo Algoritmo 2 para o Objeto 7. (a) Saída do módulo detector de movimento. (b) Objeto 7 (identificador 36) em estado *Oclusão* (em Azul) e posteriormente recuperado em estado *Rastreando* (em vermelho).



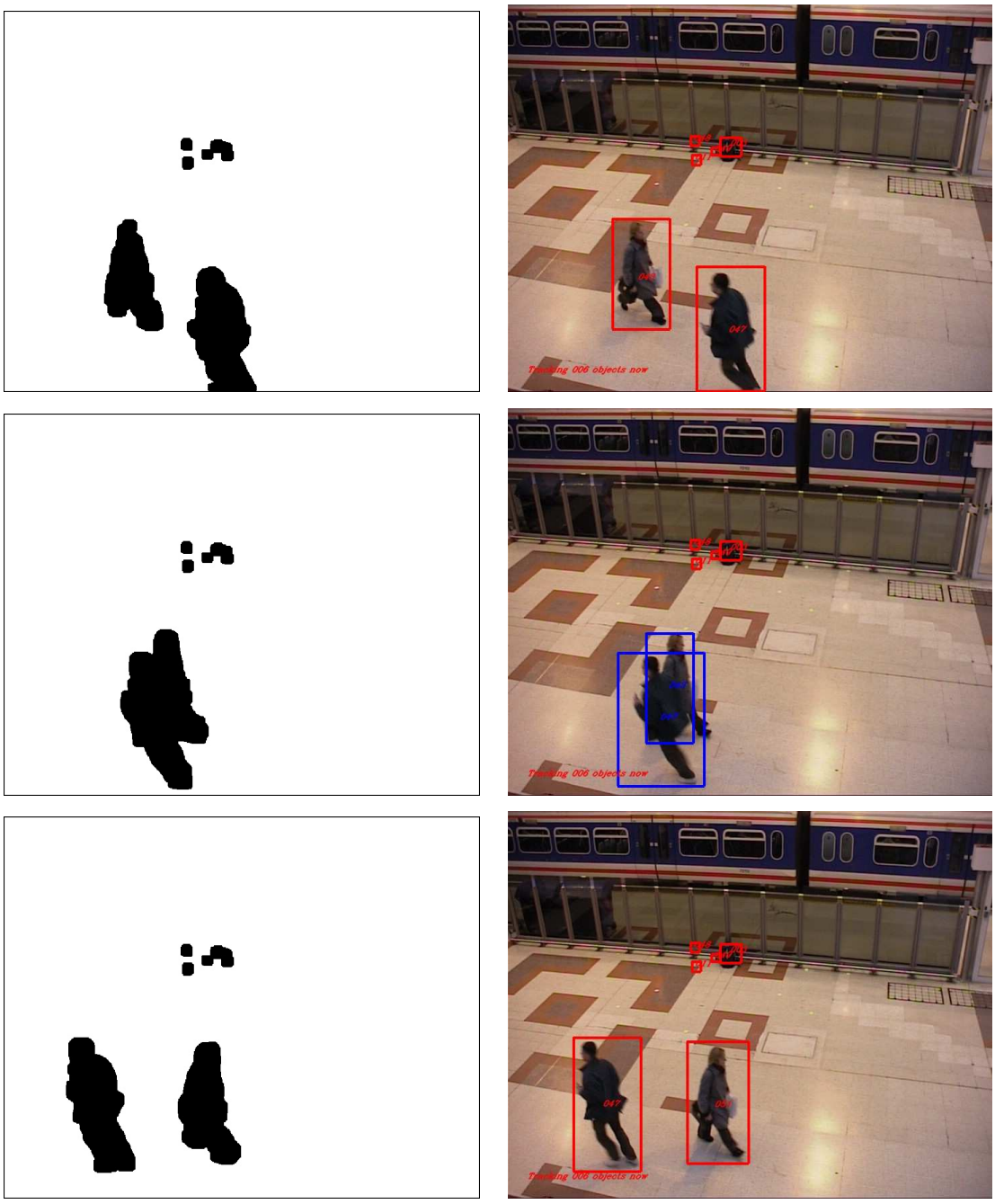
(a) (b)

Figura 6.14: Resultados obtidos pelo Algoritmo 2 para o objeto 8. (a) Saída do módulo detector de movimento. (b) Objeto 8 (identificador 227) antes, durante e após recuperação do estado de *Oclusão*.



(a) (b)

Figura 6.15: Resultados obtidos pelo Algoritmo 2 para o objeto 9. (a) Saída do módulo detector de movimento. (b) Objeto 9 (identificador 447) antes, durante e após recuperação do estado de *Oclusão*.



(a) (b)

Figura 6.16: Resultados obtidos pelo Algoritmo 2 para o objeto 10. (a) Saída do módulo detector de movimento. (b) Objeto 10 (identificador 47) antes, durante e após recuperação do estado de *Oclusão*.

4. O evento permanece em estado de alarme por um tempo (medido em quadros) maior que um valor previamente definido.

O item 1 da lista de restrições acima se refere à detecção de divisão de objetos, explicada na Subseção 6.3. Desta forma, o algoritmo de detecção de eventos procura, em sua entrada, por objetos marcados com estado *divisão* pelo módulo de rastreamento. A separação entre objeto que pratica (dono) a ação e objetos que sofrem a ação (abandonados ou removidos) foi feita pelo tamanho, aproveitando a estratégia do primeiro algoritmo de rastreamento (por semelhança de características). Ao maior objeto da divisão foi atribuído o papel de objeto que pratica a ação, enquanto que aos demais foi atribuído o papel de objetos que sofrem a ação. Os itens 2, 3 e 4 definem que os objetos que sofrem a ação devem permanecer imóveis e a uma distância mínima, por um tempo mínimo, durante a existência do evento.

Além das restrições acima citadas, cabe ao algoritmo de detecção de eventos classificar o tipo de alarme em *abandono* ou *remoção* de objetos. Baseado no trabalho de Dedeoglu (2004), a classificação dos objetos que sofrem a ação em abandono ou remoção foi feita através da comparação de histogramas. Para cada objeto que sofre a ação, são calculados dois histogramas: um correspondendo à região do objeto e outro à uma região vizinha ao objeto. Caso a diferença entre esses dois seja maior que um limiar previamente definido, o evento será classificado como abandono de objeto (ver Figuras 6.17 e 6.18), pois essa diferença indica que a área objeto e o fundo da cena diferem consideravelmente. Caso contrário será classificado como remoção de objeto, visto que os histogramas do fundo da cena e o do objeto segmentado são semelhantes (ver Figuras 6.19 e 6.20). O Apêndice F contém um fluxograma completo do algoritmo de detecção de eventos.

Para os testes do módulo detector de eventos foram utilizados três vídeos de diferentes complexidades da base de dados pública do *PETS* 2006, visto que o encontro teve como tema central a avaliação de sistemas para detecção de abandono de bagagens em locais públicos. O Vídeo 1 contém o abandono de uma mochila por uma única pessoa que desperdiça algum tempo antes de deixar a bagagem. No Vídeo 2, há uma pessoa que coloca uma mala no chão seguida de uma segunda pessoa que se aproxima e conversa com a primeira. A primeira pessoa então deixa a cena sem sua mala sem que a segunda pessoa note o abandono. Por último, no Vídeo 3, uma única pessoa com uma mala, desperdiça algum tempo antes de deixá-la abandonada no ambiente. Durante este evento, cinco outras pessoas deslocam-se nas proximidades da mala abandonada.

Para avaliar o módulo quanto a remoção de objetos do ambiente foi utilizado um vídeo (Vídeo 4) produzido nas dependências do Departamento de Sistemas e Computação (DSC) da Universidade Federal de Campina Grande (UFCG). Este último vídeo é composto por um abandono seguido de

Figura 6.17: Exemplo de quadro com objeto abandonado. Estratégia de classificação utiliza histogramas de duas áreas: a do objeto (identificado pelo retângulo menor) e a de sua vizinhança (identificado pela área compreendida entre os retângulos menor e maior).

Figura 6.18: Exemplo de histogramas. (a) Histograma da região do objeto da Figura 6.17. (b) histograma da vizinhança do objeto da Figura 6.17.

uma remoção de objetos. Neste vídeo, há um evento de abandono e outro de remoção.

Na Tabela 6.9, apresentam-se os resultados obtidos com os vídeos acima citados. Nesta tabela, na coluna **Vídeo** é identificado o vídeo de teste. Na coluna **Num. de quadros**, é indicada a quantidade de quadros do vídeo de teste. Na coluna **Núm. real de abandonos**, indica-se a quantidade de vezes um objeto é abandonado no vídeo de teste. Na coluna **Abandonos**, é indicada a quantidade de vezes em que um alarme de abandono de objeto foi gerado. Na coluna **Núm. real de remoções**, indica-se a quantidade de vezes um objeto é removido no vídeo de teste. Na coluna **Remoções**, indicam-se quantas vezes o algoritmo detectou alarmes de remoção de objetos. Por último, na coluna **Falsos positivos** são indicadas quantas vezes o algoritmo gerou alarmes incorretamente.

Pela Tabela 6.9, os vídeos 1 e 4 foram os que apresentaram melhores resultados. No Vídeo 1, o alarme é gerado imediatamente após a saída da pessoa que o abandona e assim continua após sua saída da cena como apresentado na sequência de imagens da Figura 6.21. Durante os testes com esses vídeos, o alarme deixa de existir em alguns quadros devido à imprecisão do método de segmentação, causando algum deslocamento ao objeto abandonado e, consequentemente, fazendo com que este

Figura 6.19: Exemplo de quadro com objeto abandonado. Estratégia de classificação utiliza histogramas de duas áreas: a do objeto (identificado pelo retângulo menor) e a de sua vizinhança (identificado pela área compreendida entre os retângulos menor e maior).

Figura 6.20: Exemplo de histogramas. (a) Histograma da região do objeto da Figura 6.19. (b) histograma da vizinhança do objeto da Figura 6.19.

objeto não satisfaça o item 2 das restrições comentadas no início da seção.

Os Vídeos 2 e 3 foram os que apresentaram considerável número de falsos positivos. O Vídeo 2 não possui um abandono propriamente dito, visto que há sempre uma pessoa próxima ao objeto. Mesmo assim, o algoritmo gera um alarme quando uma das pessoas se afasta um pouco mais do objeto abandonado (ver Figura 6.22). No Vídeo 3, o alarme é gerado por poucos quadros devido à problemas de incorporação do objeto. Vídeos com pessoas que desperdiçam algum tempo antes de abandonar ou remover um objeto também estão sujeitos à incorporação desses objetos pelo *background*.

Os falsos positivos encontrados nos vídeos 2 e 3 foram ocasionados principalmente pela estratégia de rastreamento do Algoritmo 1 (ver Subseção 6.3). Quando dois objetos entram em junção e, posteriormente, se dividem, o algoritmo de detecção de eventos poderá classificar a divisão desses objetos como um abandono, caso um dos objetos resultantes permaneça imóvel. Na Figura 6.23, ilustra-se um exemplo de um desses falsos positivos. Acredita-se que este tipo de problema pode ser resolvido pela adição de um classificador de objetos que diferencie pessoas de outros objetos, mas por questões de restrições de tempo e escopo, esse melhoramento foi deixado como trabalho futuro.

Figura 6.21: Alarme gerado para abandono de objeto no Vídeo 1. Os objetos envolvidos no alarme são destacados por um retângulo amarelo. A letra *O* indica o dono do objeto. A letra *A* indica o tipo de ação primitiva: abandono (A) ou remoção (R).

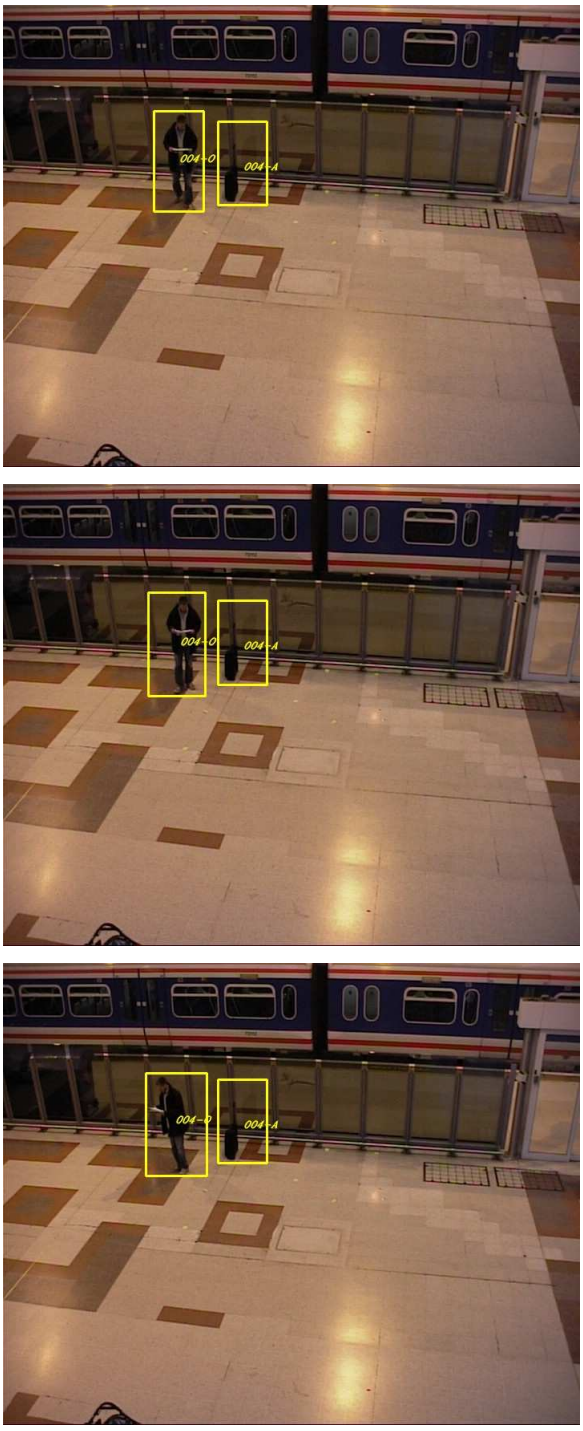


Figura 6.22: Alarme gerado para abandono de objeto no Vídeo 2. Os objetos envolvidos no alarme são destacados por um retângulo amarelo. A letra *O* indica o dono do objeto. A letra *A* indica o tipo de ação primitiva: abandono (A) ou remoção (R).

Figura 6.23: Exemplos de falsos alarmes gerados para o Vídeo 2. Os objetos envolvidos no alarme são destacados por um retângulo amarelo. A letra *O* indica o dono do objeto. A letra *A* indica o tipo de ação primitiva: abandono (A) ou remoção (R).

Tabela 6.9: *Resultados obtidos pelo módulo de detecção de eventos.*

Vídeo	**Núm. de quadros**	**Núm. real de abandonos**	**Abandonos detectados**	**Núm. real de remoções**	**Remoções detectadas**	**Falsos positivos**
Vídeo 1	3021	1	1	0	0	0
Vídeo 2	3000	0	6	0	0	5
Vídeo 3	3400	1	5	0	0	3
Vídeo 4	840	0	1	0	1	0

Como consideração final, no que diz respeito à estratégia de classificação de eventos através de histograma do objeto e de suas vizinhanças, verificou-se que pode haver erro na classificação de objetos em ambientes que não possuem *background* homogêneo.

Capítulo 7

Conclusões e Trabalhos Futuros

Neste capítulo, são apresentadas conclusões obtidas a partir dos experimentos realizados ao longo desta dissertação, no tocante às técnicas comumente empregadas em sistemas de vigilância automática a partir de vídeo. Ao final do capítulo, são apresentados possíveis novos experimentos e linhas de investigação que surgiram como decorrência deste trabalho. As discussões que se seguem estão divididas em seções associadas a cada módulo desenvolvido e experimentado.

7.1 Detecção de Movimento

Neste trabalho, foram avaliados dois modelos estatísticos adaptativos de segmentação: o modelo bimodal *W4*, proposto por Haritaoglu et al. (2000), e o modelo de mistura de Gaussianas (GMM), inicialmente proposto por Stauffer and Grimson (1999), com implementação disponível na biblioteca *OpenCV*.

Comparados aos resultados obtidos por Lara (2007) (que obteve 83% de detecção total para o modelo *W4*) e Nascimento and Marques (2006) (que obteve 85% de detecção correta para o modelo *GMM*), ambos os modelos de segmentação apresentaram resultados satisfatórios mas que merecem algumas observações relevantes:

- A principal questão a ser observada no uso de modelos de segmentação adaptativos diz respeito ao momento em que as adaptações são feitas. Adaptações muito rápidas podem prejudicar etapas posteriores, como rastreamento e detecção de eventos, enquanto que adaptações tardias podem gerar um número excessivo de objetos (não incorporados) em cena.

- Por ter um modelo baseado no valor mediano assumido por cada *pixel*, o modelo *W4* necessita armazenar todos os valores assumidos pelos *pixels* da imagem até que o *n*-ésimo quadro seja

alcançado. No momento do processo de atualização, esses valores são ordenados para que seja encontrada a nova mediana de cada *pixel*. Todo esse processo tem como consequência, maior consumo de memória e tempo de processamento, se comparado com modelos como o *GMM*, os quais atualizam o modelo de *background* a cada quadro.

- O algoritmo para detecção de sombras implementado apresentou melhores resultados quando experimentado em conjunto com o modelo *W4*, por ter sido desenvolvido para modelos de segmentação que representam o *background* em tons de cinza (ao contrário do modelo *GMM*). Entretanto, o mesmo algoritmo de detecção de sombras ocasionou um aumento nas taxas de fragmentação dos objetos, devido a falsos positivos (*pixels* de *foreground* classificados como sombra).

- O algoritmo *GMM* foi inferior nos testes com bases de dados contendo sequências de imagens em ambiente externo. Além disso, o modelo não apresentou alteração nas taxas detecção quando variado o número de Gaussianas. Como mencionado na Subseção 6.2, acredita-se que esse problema seja específico à implementação utilizada, causado pela maneira como os parâmetros da cada Gaussiana são iniciados.

Uma alternativa aos métodos que atualizam o modelo a cada n quadros é considerar a taxa de quadros por segundo na qual o vídeo foi ou está sendo gerado, pois para altas taxas de amostragem de vídeo, deve-se ter maiores valores de *n* a fim de que haja mudança considerável nos valores dos pixels até o momento da atualização do *background*. Baixas taxas de quadros por segundo sugerem atualizações de *background* mais frequentes, pois as diferenças de posicionamento dos objetos entre dois quadros consecutivos é maior.

Outra maneira de obter melhoramento nos resultados obtidos através de métodos de segmentação baseados em estatísticas extraídas de valores de *pixels* é através do refinamento desses resultados por meio do uso de técnicas de remoção de ruído (ver Subseção 2.2.3), processamento espacial de imagem como detecção das bordas dos objetos ou técnicas de morfologia matemática como as utilizadas por Lara (2007).

7.2 Rastreamento

No início do Capítulo 3, foi mencionado que o maior desafio na etapa de rastreamento seria manter um mesmo identificador para o mesmo objeto, ao longo de uma sequência de quadros de vídeo, isto é, estabelecer relações temporais entre os objetos segmentados em cada quadro de entrada. Durante

os experimentos realizados, verificou-se que são dois os principais fatores inerentes a esse desafio: possíveis falhas de segmentação e a oclusão/junção entre dois ou mais objetos.

Ao realizar a pesquisa bibliográfica e executar os primeiros experimentos, foi sendo delineado o foco das implementações e dos experimentos finais desse trabalho. Dois algoritmos de rastreamento foram implementados. Ambos tiveram por base a correspondência de características de baixo nível e estratégias para detecção de junções e divisões.

Para o Algoritmo 1, ao ser detectada uma junção, o rótulo do maior objeto (o que contém o maior número de *pixels*) era atribuído ao objeto resultante. Analogamente, ao ser detectada uma divisão, atribui-se o rótulo do objeto dividido ao maior objeto resultante. Essa estratégia obteve bons resultados com relação ao número de identificadores atribuídos a um mesmo objeto (ver Tabela 6.7) que, ao longo dos quadros, apresentou falhas de segmentação. Entretanto, em situações de oclusão real (não por falha de segmentação) o algoritmo pode trocar os identificadores dos objetos.

Para o Algoritmo 2, ao ser detectada uma junção, utiliza-se o filtro de Kalman para projetar as posições dos objetos em oclusão e tentar rastreá-los até que sejam recuperados pela saída do estado de oclusão ou a tolerância do objeto em estado de oclusão seja alcançada.

Apesar de conseguir recuperar alguns objetos corretamente (ver Tabela 6.8), a estratégia do Algoritmo 2 falhou na maioria dos casos devido ao fato de o algoritmo não fazer predição (ou atualização) do vetor de características durante o período em que um objeto se encontra no estado de oclusão, pois o algoritmo realiza predição apenas do retângulo mínimo de cada objeto. Consequentemente, propriedades de forma e tamanho do objeto após o estado de oclusão podem ser bastante diferentes daquelas extraídas antes da oclusão.

Abordagens alternativas podem ser utilizadas para melhorar os resultados de algoritmos para tratamento de oclusão como o a busca por vetores de características invariantes ou mecanismos de atualização dos vetores de características em estado de oclusão. No entanto, a tendência para tratamento de oclusão em sistemas de vigilância automática tem passado, na maior parte dos casos, pelo uso de mais de uma câmera e uso de funções de transformação que permitam ao módulo rastreamento manipular os objetos segmentados em um mesmo referencial espacial.

7.3 Detecção de Eventos

No que diz respeito à detecção de eventos, as implementações e experimentos realizados neste trabalho basearam-se em heurísticas simples para detecção de abandono e subtração de objetos, através da combinação de técnicas para detecção de divisão de objetos do módulo de rastreamento e a aplica-

ção de algumas restrições simples aos objetos dessa divisão. Os resultados encontrados mostraram a viabilidade do uso dessas heurísticas na detecção de eventos primitivos em vídeos de baixa complexidade, ou seja, vídeos com um número reduzido de objetos em oclusão.

Entretanto, a detecção de eventos e ações humanas complexas (andar, correr, lutar, por exemplo), exige o uso dos mecanismos mais tradicionais de aprendizagem de máquina, por possuírem maior poder de generalização e, consequententemente, alcançarem baixos índices de falsos positivos.

No que diz respeito ao processo de vigilância automática, percebeu-se que os pontos críticos para o êxito de tais sistemas estão na segmentação e na manutenção dos rótulos da cada objeto em cena. Esses dois itens se traduzem em uma segmentação confiável e tratamento de oclusão robusto. Caso contrário, um módulo receberá entradas deficientes do outro.

Contudo, a adição, cada vez maior, de novas camadas de software para abordar, mais e melhor, os desafios propostos por sistemas de vigilância automática, esbarra em outro requisito essencial inerente ao funcionamento desse tipo de sistema: a necessidade de tempos de resposta cada vez menores aos objetivos da aplicação.

7.4 Trabalhos Futuros

Diante dos resultados apresentados neste trabalho e de acordo com a arquitetura proposta para um sistema de vigilância automática apresentada na Seção 5.1 do Capítulo 5 surgem, naturalmente, novas idéias para melhoramento dos experimentos, desenvolvimentos de novos modelos e linhas de pesquisa. Neste sentido, estão listados a seguir alguns dos trabalhos futuros sugeridos para a continuidade da pesquisa em sistemas de vigilância automática:

- Experimentar novos modelos de segmentação de movimento, bem como a combinação desses modelos;
- Pesquisar e avaliar a representatividade de características que compõem os vetores de características utilizados no rastreamento de objetos;
- Implementar algoritmos de rastreamento capazes de utilizar informações de múltiplas câmeras;
- Incorporar o uso de classificadores de diversos tipos de objetos (pessoas, carros, etc) com objetivo de minimizar falsos positivos;
- Investigar a viabilidade de paralelização das tarefas realizadas pelos módulos do sistema, com objetivo de atender aos requisitos de tempo de resposta das aplicações;

- Elaborar técnicas para otimização dos diversos parâmetros de cada módulo.

A realização da lista de possíveis experimentos e linhas de investigação acima citados seria facilitada pela elaboração de um *framework* que permitisse utilizar as técnicas tradicionais já existentes, bem como adição e teste de novos modelos, não apenas em sistemas de vigilância automática, mas em sistemas de análise de movimento como um todo.

Apêndice A

Modelo Bimodal Adaptativo (*W4*)

Como informado no Capítulo 2, o modelo *W4* proposto por Haritaoglu et al. (2000) é denominado modelo bimodal, pois é baseado nos valores máximo e mínimo capturados em cada *pixel* ao longo do tempo. O modelo de *background* é representado por uma estrutura contendo três valores para cada posição (x, y) da imagem: $m(x, y)$, $n(x, y)$ e $d(x, y)$ que correspondem, respectivamente, aos valores mínimo, máximo e a diferença máxima entre dois quadros consecutivos.

O processo completo de detecção de movimento no modelo *W4* consiste de quatro etapas: iniciação, classificação dos *pixels*, atualização dos mapas de mudança e atualização do modelo (parte adaptativa do processo). Abaixo segue um breve resumo de cada uma dessas etapas:

Iniciação: Seja V um conjunto de n_0 quadros consecutivos. A iniciação do modelo de *background* é feita calculando-se os valores $m(x, y)$, $n(x, y)$ e $d(x, y)$ dos *pixels* de V que obedecem à condição de *pixels* estacionários. A fim de encontrar os *pixels* estacionários, aplica-se um filtro da mediana temporal no conjunto V, isto é, no conjunto de quadros capturados durante alguns segundos de monitoramento. A etapa de iniciação é sintetizada na Equação A.1.

$$\begin{bmatrix} m(x,y) \\ n(x,y) \\ d(x,y) \end{bmatrix} = \begin{bmatrix} min_z\{V^z(x,y)\} \\ max_z\{V^z(x,y)\} \\ max_z\{|V^z(x,y) - V^{z-1}(x,y)|\} \end{bmatrix} \tag{A.1}$$

em que

- $V^z(x, y)$ é um *pixel* classificado como estacionário;
- $|V(x, y) - \lambda(x, y)| < 2\sigma(x, y)$ é a restrição para que um *pixel* seja considerado estacionário;
- $\lambda(x, y)$ é a mediana dos valores de um *pixel* na posição (x, y);

- $\sigma(x, y)$ é o desvio padrão dos valores de um *pixel* na posição (x, y);

Classificação dos *pixels*: Uma vez que o modelo de *background* seja iniciado, o modelo de segmentação passa a classificar os *pixels* dos quadros de entrada em *foreground* e *background* de acordo com uma equação de restrição definida na Equação A.2. Vale lembrar que a Equação A.2 é um aperfeiçoamento proposto por Jacques et al. (2005) em relação à equação original proposta no sistema por Haritaoglu et al. (2000).

$$M^t(x,y) = \begin{cases} 1 & \text{se } I^t(x,y) \leq |m(x,y) - kd_\mu| \text{ ou} \\ & \quad I^t(x,y) \geq |n(x,y) - kd_\mu| \\ 0 & \text{caso contrário} \end{cases} \tag{A.2}$$

Atualização dos mapas de mudança: A atualização do modelo de *background W4* é feita com o auxílio de três mapas denominados mapas de mudança. Assim como no modelo de *background*, em cada mapa existe uma valor para cada *pixel* da imagem. Os valores de cada mapa são atualizados toda vez que um quadro é processado durante a fase de classificação dos *pixels*. A cada n quadros classificados os mapas são utilizados para atualização. Os mapas de atualização são definidos como nas Equações A.3, A.4 e A.5.

- **Mapa de detecção** (Dm): representa o número de vezes que um *pixel* foi classificado como *background* nos últimos n quadros.

$$Dm(x,y,t) = \begin{cases} Dm(x,y,t-1) + 1 & \text{, se } (x,y) \text{ é um } \textit{pixel} \text{ do } \textit{background} \\ Dm(x,y,t-1) & \text{, se } (x,y) \text{ é um } \textit{pixel} \text{ do } \textit{foreground} \end{cases} \tag{A.3}$$

- **Mapa de movimento** (Mm): representa o número de vezes que um *pixel* foi classificado como um *pixel* em movimento nos últimos n quadros. Um *pixel* é classificado como em movimento pela subtração do mesmo em três quadros consecutivos.

$$Mm(x,y,t) = \begin{cases} Mm(x,y,t-1) + 1 & \text{se } mv(x,y) = 1 \\ Mm(x,y,t-1) & \text{se } m(x,y) = 0 \end{cases} \tag{A.4}$$

em que

$$mv(x,y,t) = \begin{cases} 1 & \text{se } (|I^t(x,y,t) - I^t(x,y,t+1)| > 2\sigma) \text{ e} \\ & \quad (|I^t(x,y,t-1) - I^t(x,y,t)| > 2\sigma) \\ 0 & \text{caso contrário} \end{cases}$$

- **Mapa de histórico de mudança** (Hm): representa o tempo decorrido desde a última vez que um *pixel* foi classificado como *foreground*.

$$Hm(x,y,t) = \begin{cases} 255 & \text{se } (x,y) \text{ é } \textit{foreground} \\ Hm(x,y,t-1) - \frac{255}{n} & \text{caso contrário} \end{cases}$$

Atualização do modelo: A atualização do modelo é feita a cada n quadros substituindo o modelo de *background* corrente, quando necessário, pelo modelo de *foreground* ou *background* dos últimos n quadros. Estes dois últimos modelos são computados em paralelo ao modelo corrente. A substituição do modelo na fase de atualização corresponde à adaptação do modelo às mudanças do ambiente. O modelo proposto por *W4* possui duas formas de atualização: baseada em *pixel* (*pixel based*) e baseada em objeto (*object based*). A primeira atualiza o modelo de *background* com relação a mudanças de iluminação no ambiente. A segunda atualiza o modelo pela incorporação de objetos que permaneçam algum tempo parados no ambiente. A atualização do modelo originalmente proposta é realizada de acordo com a Equação A.5.

$$[m(x,y), n(x,y), d(x,y)] = \begin{cases} [m(x,y)^b, n(x,y)^b, d(x,y)^b] & \text{se } Dm(x,y) > kn \\ [m(x,y)^f, n(x,y)^f, d(x.y)^f] & \text{se } Dm(x,y) < k*n \text{ e} \\ & Hm(x,y) < r*n \\ [m(x,y)^c, n(x,y)^c, d(x,y)^c] & \text{caso contrário} \end{cases}$$

em que

- r e k são constantes, tipicamente 0.8 e 0.01;
- $[m(x,y)^b, n(x,y)^b, d(x,y)^b]$ representa o modelo de *background* gerado para os pixels classificados como *background*;
- $[m(x,y)^f, n(x,y)^f, d(x,y)^f]$ representa o modelo de *background* gerado para os pixels classificados como *foreground*;
- $[m(x,y)^c, n(x,y)^c, d(x,y)^c]$ representa o modelo de *background* atual;

As restrições apresentadas na Equação A.5 correspondem, respectivamente, às duas formas de atualização anteriormente mencionadas. Entretanto, testes realizados após a implementação do modelo utilizando a Equação A.5 apresentaram grande quantidade de falsos positivos a cada n quadros.

Diante dos resultados apresentados, foi proposta neste trabalho, uma modificação na atualização baseada em objeto. Considerando que um objeto deve ser incorporado ao *background* quando o mesmo estiver parado e classificado como *foreground*, a restrição para que o modelo de *foreground* substitua o modelo corrente nos *pixels* pertencentes ao objeto passou a ser como descrito pela Equação A.5.

$$[m(x,y), n(x,y), d(x,y)] = \begin{cases} [m(x,y)^b, n(x,y)^b, d(x,y)^b] & \text{se } Dm(x,y) > kn \\ [m(x,y)^f, n(x,y)^f, d(x.y)^f] & \text{se } Mm(x,y) < r * n \text{ e} \\ & Hm(x,y) < k * n \\ [m(x,y)^c, n(x,y)^c, d(x,y)^c] & \text{caso contrário} \end{cases}$$

A Tabela A.1 mostra um mapeamento dos parâmetros do modelo nas respectivas variáveis de configuração existentes na implementação.

Tabela A.1: *Parâmetros do modelo W4.*

Parâmetro	**Descrição**
n_0	Número de quadros utilizados no processo de iniciação do modelo.
n	Número de quadros utilizados para atualização do modelo.
k	Constante utilizada na atualização do modelo.
r	Constante utilizada na atualização do modelo.
t	Constante de limiarização do modelo na fase de classificação.

Apêndice B

Modelo de Mistura de Gaussianas (*GMM*)

Modelo de mistura é um tipo de função de densidade que engloba um número de funções componente, usualmente funções Gaussianas. Estas funções componentes são combinadas para prover uma nova função densidade. Em termos matemáticos, define-se um modelo de mistura Gaussiana como:

$$P(x) = \sum_{i=1}^{k} \omega_i \eta(x, \mu_i, \Sigma_i) \tag{B.1}$$

em que,

- $P(x)$ representa a probabilidade posterior de x;
- k é o número de Gaussianas utilizadas;
- ω é uma ponderação atribuída à i-ésima Gaussiana;
- η representa a função componente (densidade de probabilidade) parametrizada por μ_i e Σ_i, que são, respectivamente, o valor médio e a matrix de covariância da i-ésima Gaussiana da mistura.

No modelo proposto por Stauffer and Grimson (1999) existe uma mistura de k curvas Gaussianas para cada *pixel* da imagem. Analogamente ao modelo *W4*, o processo de segmentação no modelo *GMM* possui três etapas:

Iniciação: consiste da iniciação dos valores do modelo, como μ_i e Σ_i da cada Gaussiana. Um algoritmo muito utilizado para estimativa destes parâmetros é o de Maximização de Expectativas (*Expectation Maximization*). Na implementação do modelo disponível na biblioteca *OpenCV*, o valor

de μ_i é iniciado com o valor do *pixel* da imagem de entrada. A matriz Σ_i é representada pelo desvio padrão da Gaussiana σ_i. O valor inicial de σ_i é atribuído através de parâmetro de entrada.

Classificação: as k distribuições que representam um determinado *pixel* são ordenadas de acordo com a relação $\frac{\omega_i}{\sigma_i}$. As B primeiras distribuições são utilizadas como modelo de *background* do *pixel* para a cena. O valor de B é dado pela Equação B.2.

$$B = argmin_b \left(\sum_{j=1}^{b} w_j > T \right) \tag{B.2}$$

Um *pixel* da imagem de entrada é então classificado como *foreground* se seu valor for maior que $\sigma_t h$ vezes o desvio padrão de qualquer uma das B Gaussianas escolhidas.

Atualização: A atualização do modelo é feita imediatamente após a etapa de classificação através da avaliação da Gaussiana mais representativa entre as k Gaussianas que representam o *pixel*, isto é, atualizando os parâmetros da Gaussiana mais representativa para o valor do pixel. Caso nenhuma das Gaussianas se aproxime do valor do pixel, uma nova Gaussiana será adicionada à mistura em substituição a menos representativa.

A Tabela B.1 mostra um mapeamento dos parâmetros do modelo nas respectivas variáveis de configuração existentes na implementação.

Tabela B.1: *Parâmetros do modelo GMM.*

Parâmetro	**Descrição**
T	Porcentagem mínima de background presente na cena. Utilizado na fase de classificação.
σ_{th}	Número de vezes que o valor do *pixel* deve ultrapassar o desvio padrão de uma das B Gaussianas para ser considerado como *foreground* .
w	Constante utilizada atualização do modelo dos pesos das Gaussianas.
ng	Número de Gaussianas por pixel.
σ_0	Desvio padrão inicial de cada Gaussiana.
ω_0	Peso inicial de cada Gaussiana.

Apêndice C

Análise *ROC* dos Modelos *GMM* e *W4*

As figuras que seguem apresentam os resultados obtidos pela análise *ROC* dos parâmetros dos modelos GMM e W4. Os comentários a estes resultados encontram-se na Seção 6.2.

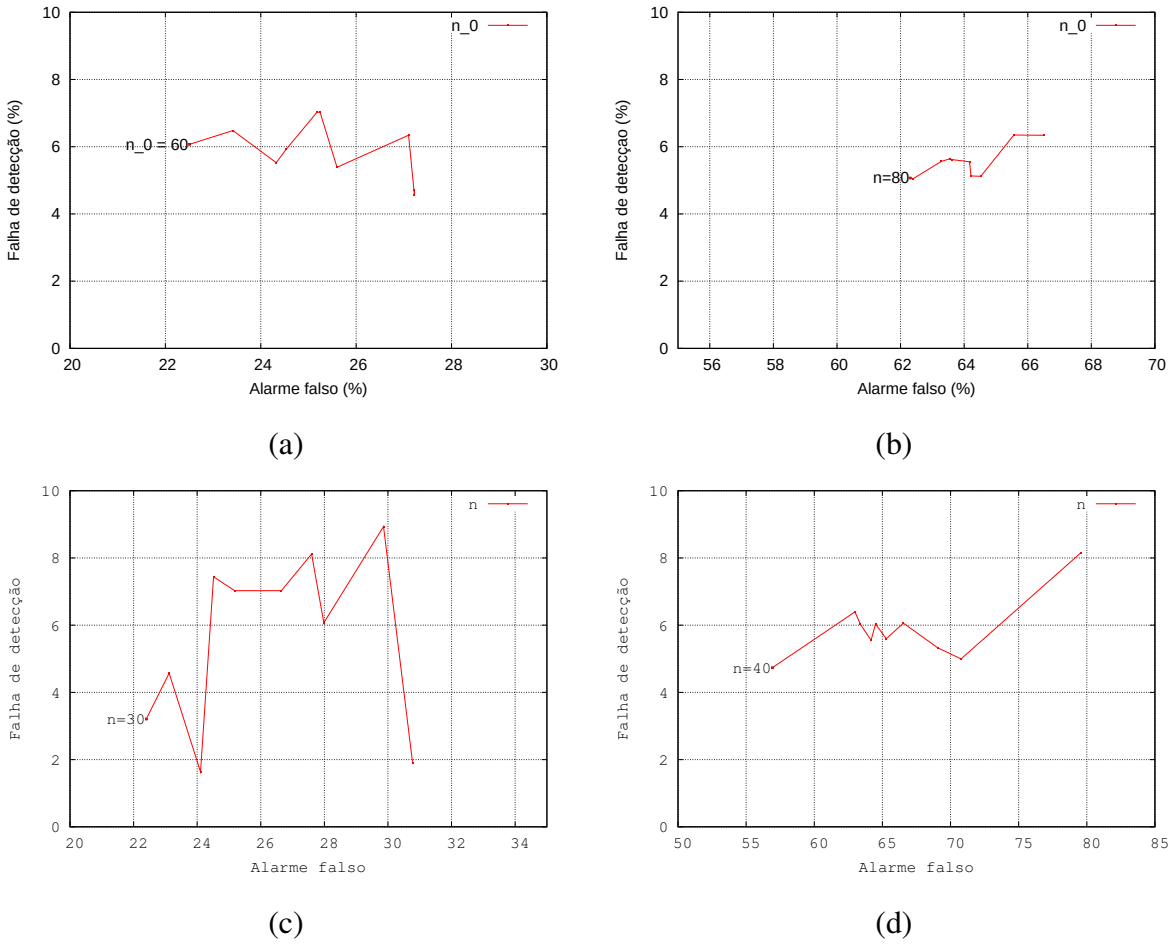


Figura C.1: Análise *ROC* do modelo *W4*. À esquerda ((a) e (c)), ambiente interno. À direita ((b) e (d)) ambiente externo.

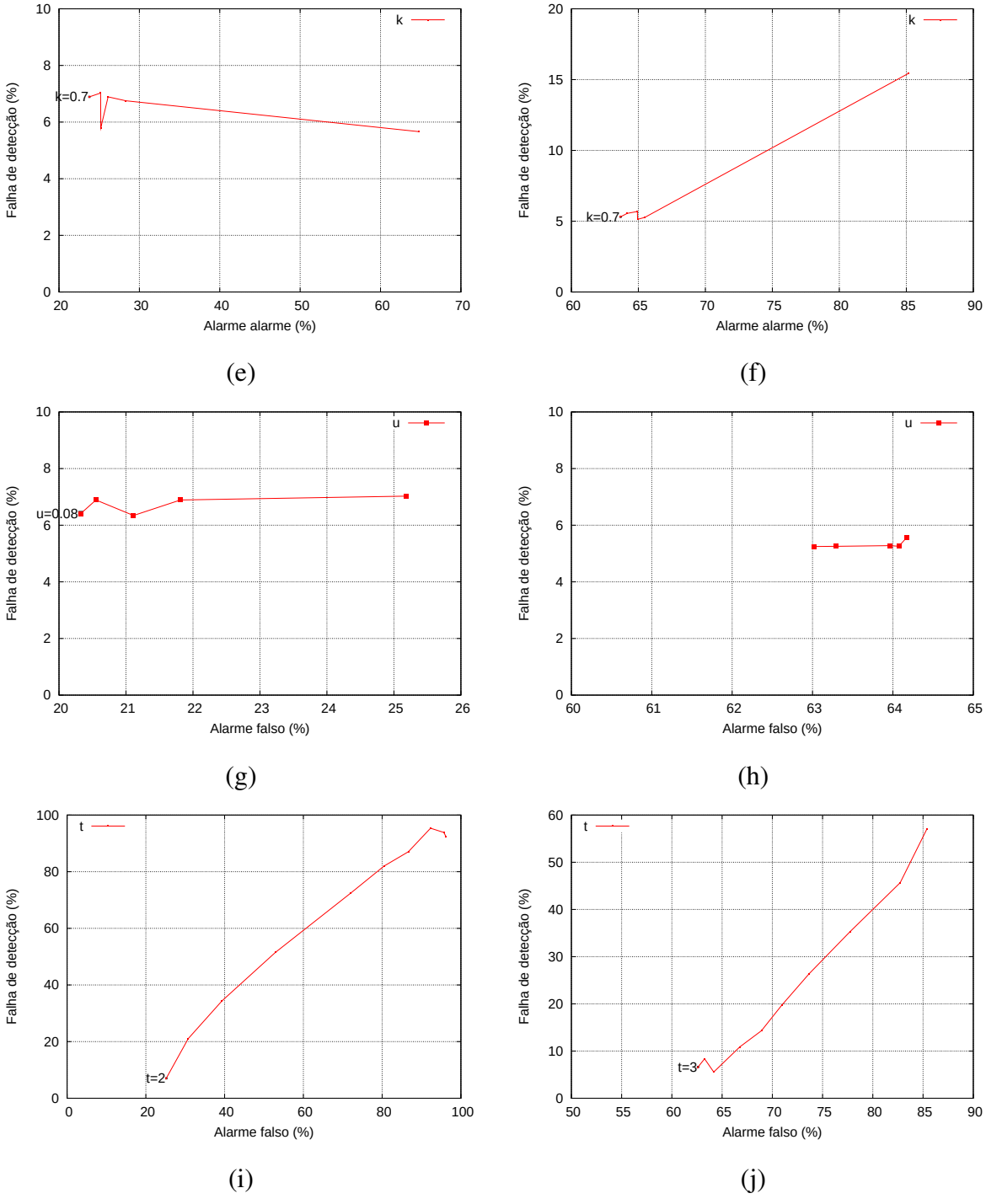


Figura C.2: Análise *ROC* do modelo *W4*. Continuação. À esquerda ((e), (g) e (i)), ambiente interno. À direita ambiente externo ((f), (h) e (j)) .

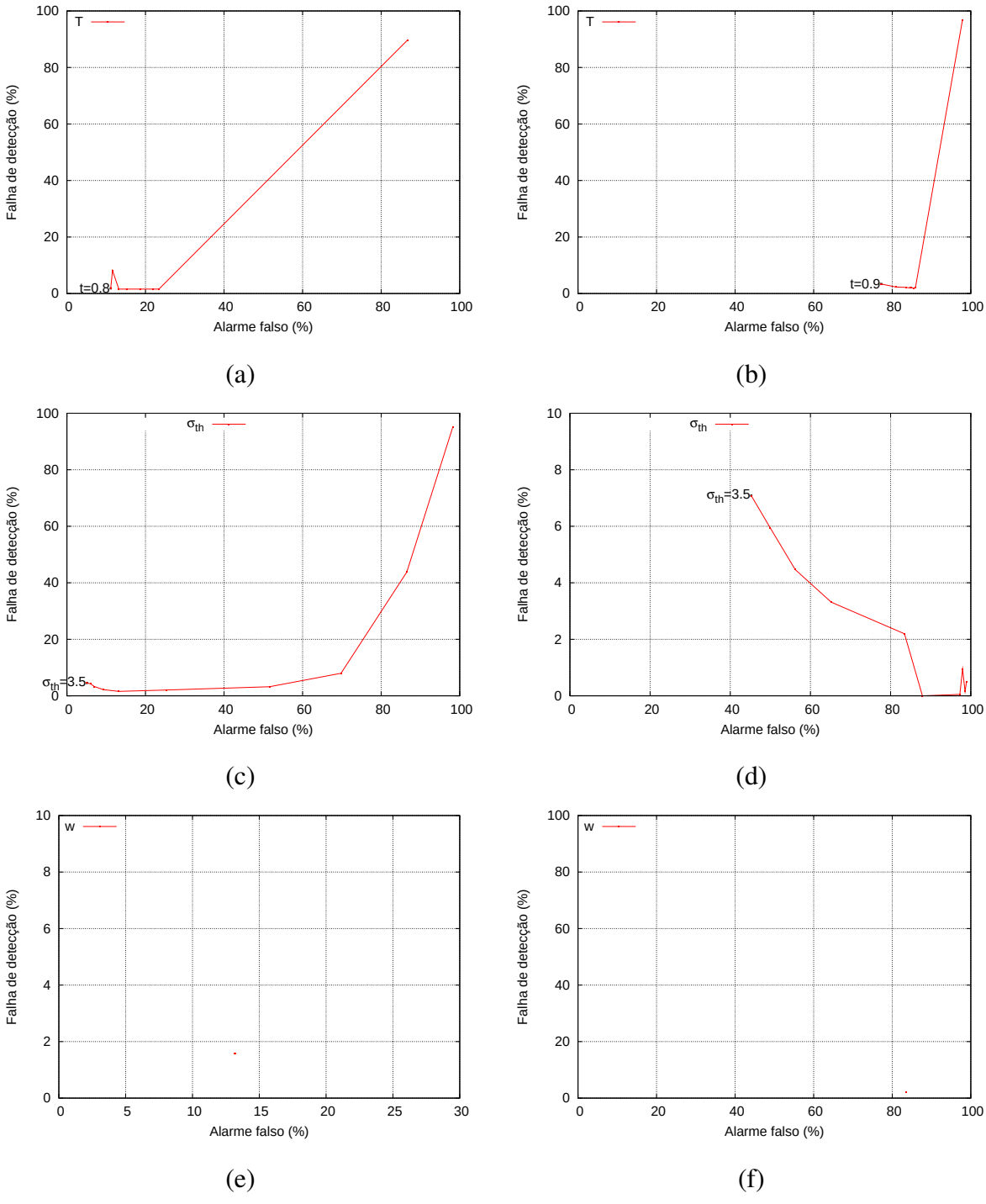


Figura C.3: Análise ROC do modelo *GMM*. À esquerda ((a), (c) e (e)), ambiente interno. À direita ambiente externo ((b), (d) e (f)).

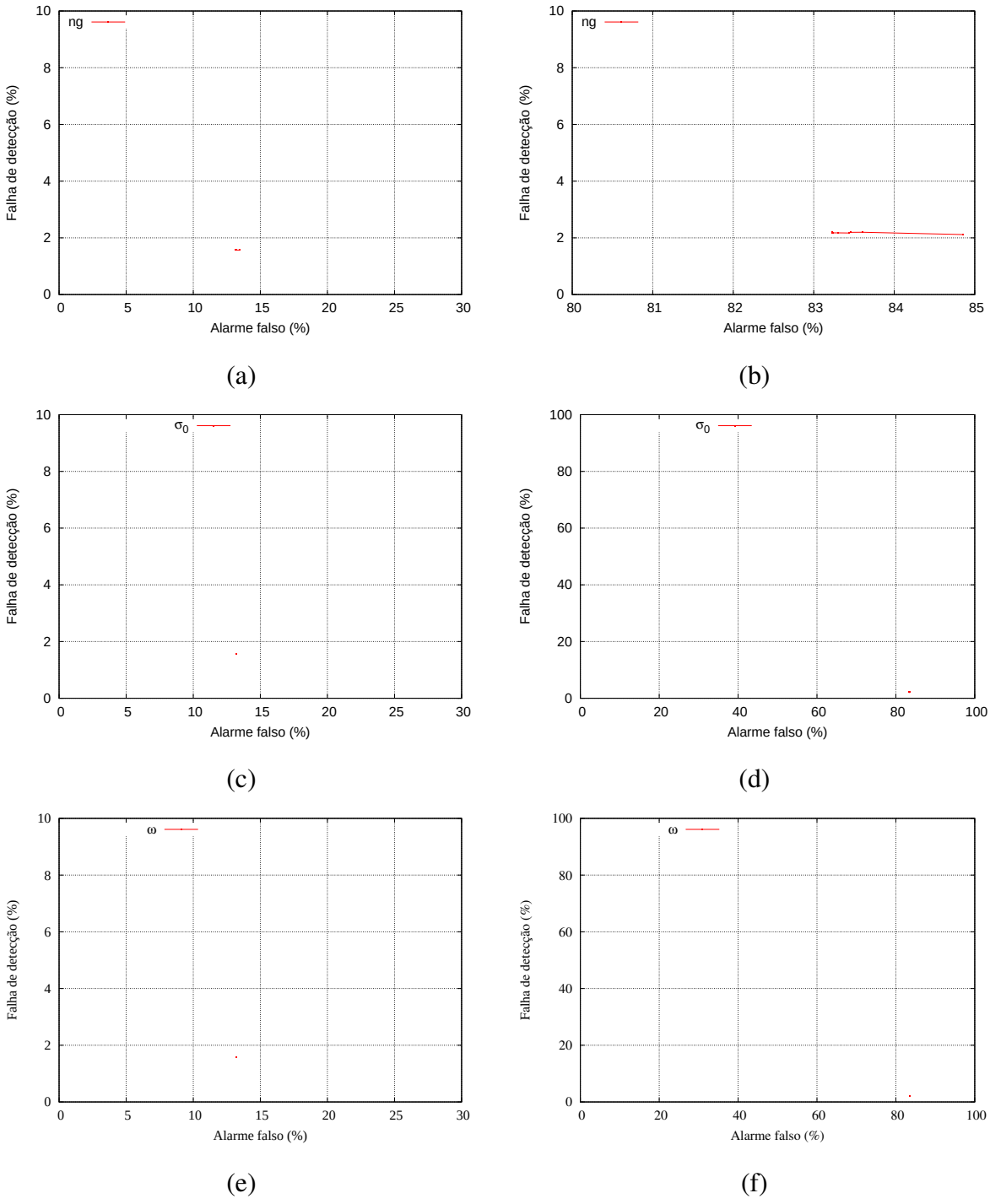


Figura C.4: Análise ROC do modelo *GMM*. Continuação. À esquerda ((a), (c) e (e)), ambiente interno. À direita ambiente externo ((b), (d) e (f)).

Apêndice D

Análise *ROC* baseada em Pixel

Como mencionado na Seção 6.2, as métricas baseadas em objetos utilizadas na avaliação quantitativa realizada neste trabalho podem gerar curvas atípicas quando comparadas a curvas ROC clássicas. Isso pode ser constatado nas curvas apresentadas para a propriedade t do modelo *W4* e na propriedade σ_{th} do modelo GMM (ver Apêndice C).

Esse comportamento se deve ao fato de que a análise quantitativa como proposta por Nascimento and Marques (2006) compara características de *Falha de Detecção* (objetos do conjunto verdade que não foram detectatos) e *Alarme Falso* (detecção de objetos que não existem no conjunto verdade).

No caso do parâmetro t do modelo *W4*, a variação de 2 até 10 deste parâmetro fez, como comentado na Seção 6.2, com que o modelo ficasse mais seletivo, isto é, menos *pixels* foram classificados como *foreground*. Consequentemente, os objetos detectados que não atingem a sobreposição mínima exigida pela métrica utilizada (50%, por exemplo) serão contabilizados como ruído e aumentarão o valor da métrica *Alarme falso*. De maneira análoga, um objeto que não seja detectado pelo aumento da restrição do algoritmo também ocasionará aumento na métrica *Falha de Detecção*. Dessa maneira, a análise quantitativa realizada para o parâmetro t do modelo *W4* apresentou uma curva quase linear (ver Figura C.2).

Com objetivo de investigar o comportamento das curvas *ROC* como em sua definição original, decidiu-se coletar métricas de positivos verdadeiros e falsos positivos para a propriedade t do modelos *W4*. Para estes novos gráficos foram utilizadas as definições apresentadas na Tabela D.1.

As métricas definidas na Tabela D.1 foram calculadas para cada quadro do mesmo vídeo utilizado para análise quantitativa do parâmetro t do modelo *W4* em ambiente interno, apresentada no Apêndice C. Por fim, foram contruídos gráficos com as taxas médias de *TP FP* nos encontradas para os diferentes valores de t. Os resultados obtidos estão ilustrados na Figura D.1.

Tabela D.1: Definições das métricas utilizadas para análise *ROC*.

Métrica	**Descrição**
TP	Positivos verdadeiros. Correspondem ao número de *pixels* da interseção entre objetos detectados pelo algortimo e objetos pertencente ao conjunto verdade.
FP	Falsos positivos. Correspondem ao número de *pixels* que foram detectados pelo algoritmo como *foreground*, mas que no conjunto verdade possuem valor de background.
P	Total de *pixels* de objetos *foreground* do conjunto verdade.
N	Total de *pixels* de *background* do conjunto verdade
Taxa de TP	Razão entre TP e P
Taxa de FP	Razão entre FP e N

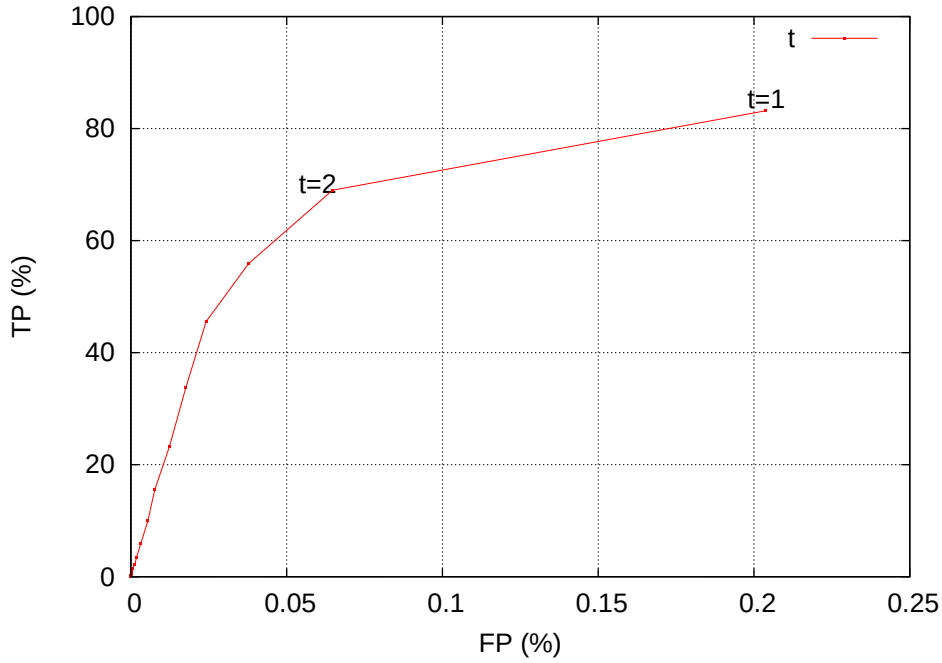


Figura D.1: Análise *ROC* para o parâmetro *t* do modelo *W4*.

De acordo com a Figura D.1, percebe-se que a análise baseada em *pixel* apresentou uma curva semelhante àquelas apresentadas por curvas *ROC* características. No entanto, percebe-se que uma pequena variação eixo que corresponde às taxas de falsos positivos. Isso se deve ao fato de que há muito mais *pixels* em N (total de *pixels* do *background*) do que em P (total de *pixels* do *foreground*).

Com objetivo de ampliar a variação nas taxas de falsos positivos, foi incluído o valor 0 nos testes com o parâmetro t do modelo *W4*. O comportamento obtido está ilustrado na Figura D.2. Nota-se pela Figura D.1 que a inclusão do desse valor elevou a taxa de *Falso Positivos* para o valor máximo, alterando o comportamento da curva. Esta modificação se deve ao fato de que valores muito abaixo do valor padrão (2) diminuem a restrição do algoritmo, isto é, o algoritmo passa a classificar quase toda a imagem como *foreground*.

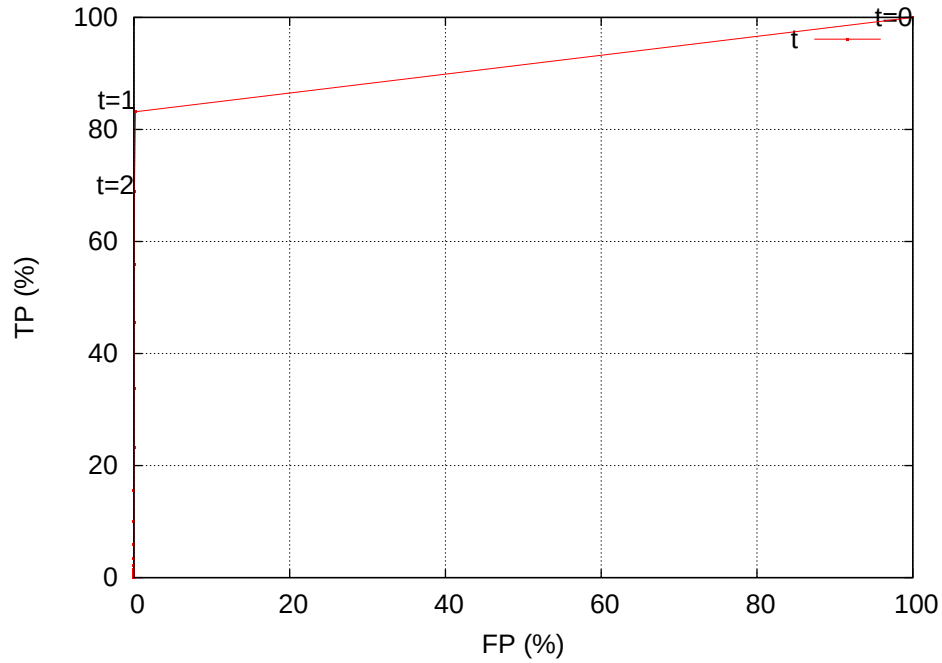


Figura D.2: Análise *ROC* para o parâmetro *t* do modelo *W4* com valor 0.

Apêndice E

Módulo de Rastreamento

E.1 Algoritmo de Rastreamento por Similaridade de Características

Na Figura E.1 está ilustrado o fluxograma que representa o Algoritmo 1 de rastreamento, baseado na correspondência das características listadas na Tabela 3.1.

E.2 Algoritmo de Rastreamento com Filtro de Kalman

Na Figura E.2 o fluxograma que representa o Algoritmo 2 de rastreamento, baseado na correspondência das características listadas na Tabela 3.1 e no uso do Filtro de Kalman.

E.3 Possíveis Estados Assumidos por um Objeto Rastreado

A Tabela E.1 lista os possíveis estados assumidos por um objeto rastreado pelo módulo de rastreamento.

E.4 Estratégia para Detecção de Oclusão

Quando as regiões de dois ou mais objetos entram em contado, ocorre *oclusão* ou *junção* e, enquanto houver oclusão, tais regiões serão representadas por uma única região nos quadros subsequentes. Segundo a estratégia apresentada por Amer (2005), quando ocorre oclusão, o algoritmo de corres-

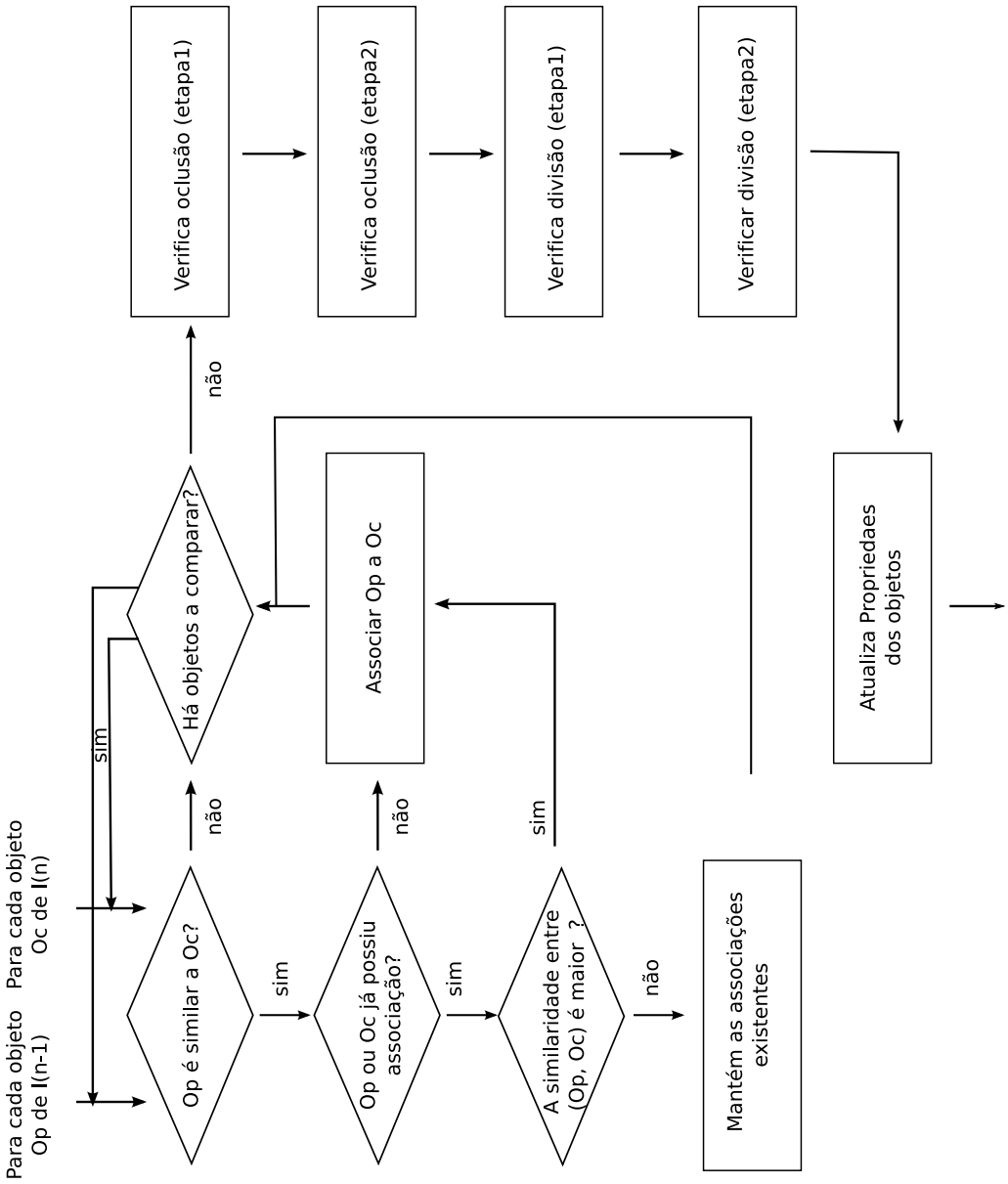


Figura E.1: Fluxograma do algoritmo de rastreamento por similaridade de características.

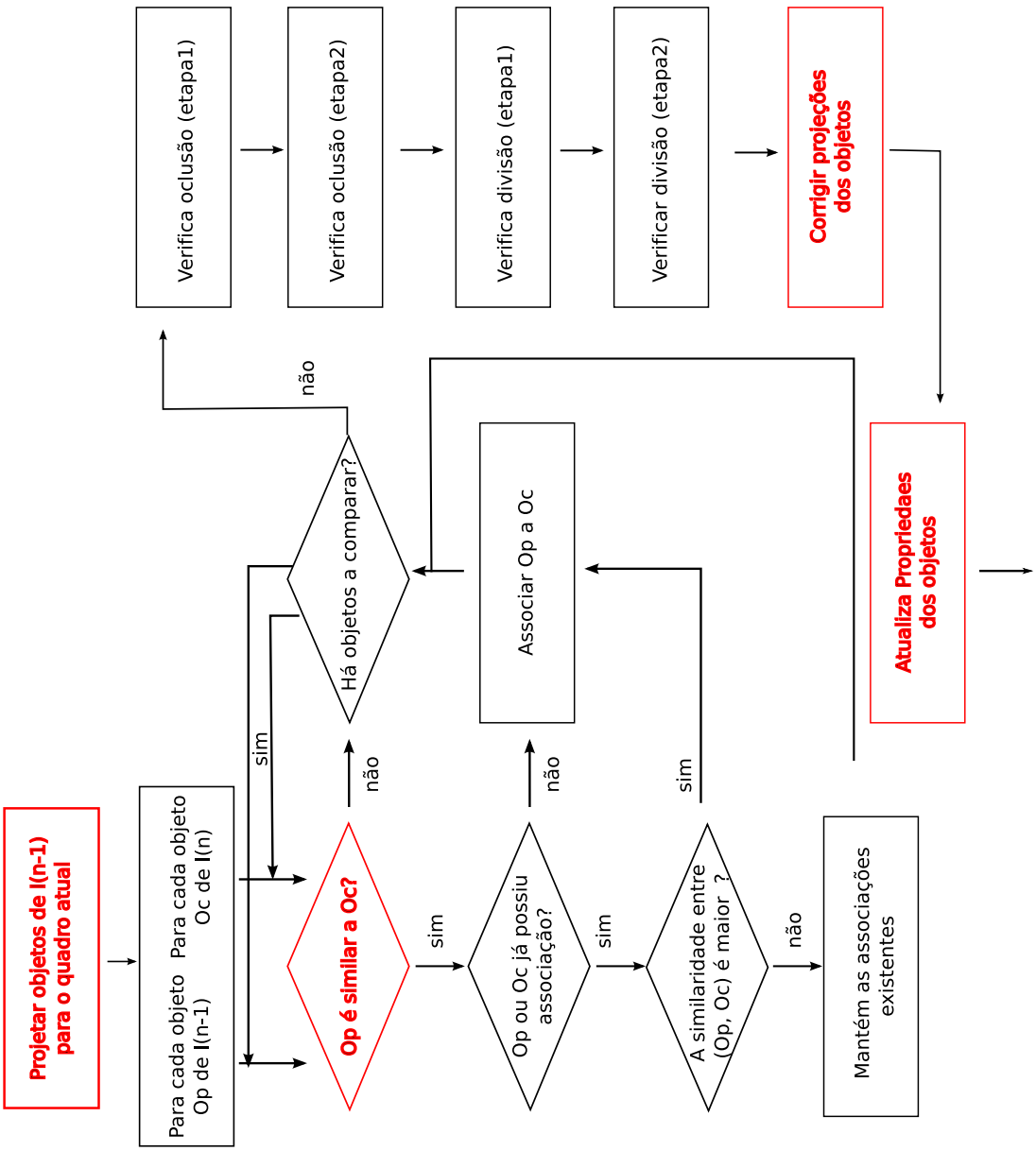


Figura E.2: Fluxograma do Algoritmo 2 de rastreamento por similaridade de características com auxílio do Filtro de Kalman. Diferenças em relação ao Algoritmo 1 destacadas em vermelho.

Tabela E.1: *Estados de um objeto rastreado.*

Estado	**Descrição**
Novo	Quando um objeto aparece pela primeira vez em cena.
Rastreando	Quando o objeto está sendo rastreado normalmente pelo algoritmo.
Perdido	Quando um objeto que estava em estado *Rastreando* deixa de rastreado por falta de correspondência.
Oclusão	Quando uma oclusão é detectada.
Divisão	Quando um objeto dá origem a dois outros.
Desconhecido	Quando um objeto não se encontra em nenhum dos estados anteriores.

pondência por similaridade de objetos associa a região que representa o grupo a pelo menos um dos objetos que existia em quadros anteriores. Assim, o objeto que no quadro corrente, representa o grupo de objetos em oclusão, experimenta um considerável aumento em pelo menos um dos lados de seu retângulo mínimo. Sejam as seguintes definições:

- O_c A lista de objetos extraídos do quadro corrente;
- O_p A lista de objetos rastreados pelo sistema;
- O_{ic} Objeto pertencente ao quadro corrente e que representa o grupo de objetos em oclusão (digamos O_{kp} e O_{lp});
- d_{kl} Distância entre dois objetos em oclusão (O_{kp} e O_{lp});
- $w_{kp} = (w_x, w_y)$ Deslocamento de O_{kp} nos dois últimos quadros;
- w_{vl} Deslocamento da borda inferior do retângulo mínimo de O_{kp} nos dois últimos quadros;
- w_{vu} Deslocamento da borda superior do retângulo mínimo de O_{kp} nos dois últimos quadros;
- w_{hr} Deslocamento da borda direita do retângulo mínimo de O_{kp} nos dois últimos quadros;
- w_{hl} Deslocamento da borda esquerda do retângulo mínimo de O_{kp} nos dois últimos quadros;
- t_w limiar de deslocamento de objetos;
- t_d limiar de distância entre objetos.

Dois objetos são declarados em estado de oclusão quando, 1) o deslocamento da associação ($O_{kp} \rightarrow O_{ic}$) e o deslocamento de uma das bordas de O_{kp} diferem significativamente e 2) quando o

deslocamento de uma das bordas é positivo. Matematicamente, essas duas restrições estão representadas na Equação E.1.

$$\begin{aligned}
&(|w_x - w_{hr}| > t_w) \wedge (w_{hr} > 0) \wedge (d_{kl} < t_d) \vee \\
&(|w_x - w_{hl}| > t_w) \wedge (w_{hl} < 0) \wedge (d_{kl} < t_d) \vee \\
&(|w_y - w_{vl}| > t_w) \wedge (w_{vl} > 0) \wedge (d_{kl} < t_d) \vee \\
&(|w_y - w_{vu}| > t_w) \wedge (w_{vu} < 0) \wedge (d_{kl} < t_d)
\end{aligned} \tag{E.1}$$

Após a realização de testes com sequências de imagens sintéticas, verificou-se que a estratégia representada na Equação E.1 não obteve sucesso em casos nos quais o objeto do quadro corrente que representa objetos em oclusão não era associado a qualquer objeto do quadro anterior. Optou-se então por adicionar mais uma etapa de verificação de objetos em oclusão.

A segunda etapa de verificação de oclusão consiste em procurar por objetos do quadro corrente que não possuem associação e que fazem interseção com pelo menos dois objetos rastreados até o quadro anterior que também não possuam associação com outros objetos.

E.5 Rastreamento de Objetos em Oclusão

Objetos que se encontram no estado *Oclusão* necessitam de alguma heurística para a atualização de suas posições enquanto não retornarem ao estado de *Rastreado* ou assim permanecerem até que sejam declarados como perdidos. Para a versão do algoritmo de rastreamento sem o auxílio do Filtro de Kalman, optou-se pela atualização através da repetição do último deslocamento (w_x, w_y) de cada objeto. Na versão do algoritmo que possui Filtro da Kalman, quando um objeto entra em estado de oclusão, a atualização da posição do objeto é feita com a posição estimada na fase de projeção do algoritmo. Na implementação feita para este trabalho, coube ao módulo de rastreamento a decisão de quando eliminar da lista de objetos rastreados, aqueles que permaneçam em estado de Oclusão ou Perdido.

E.6 Estratégia para Detecção de Divisões

Quando a região de um objeto rastreado dá origem a duas ou mais regiões, ocorre divisão (*split*). A divisão pode ser ocasionada, tipicamente por erros de segmentação, separação de objetos em estado de oclusão e eventos de abandono de objetos. No trabalho de Amer (2005), a detecção de divisões foi feita com objetivos de corrigir erros de segmentação. Neste trabalho, a detecção de divisões proposta

por Amer (2005) foi feita com intuito de auxiliar na detecção de objetos abandonados.

Analogamente ao processo de detecção de oclusão, quando ocorre uma divisão, o algoritmo de correspondência por similaridade de objetos associa apenas um dos dois (ou mais) objetos gerados ao objetos original, enquanto que os demais permanecem sem associação. Segundo Amer (2005), um objeto que no quadro corrente sofreu divisão, experimenta considerável diminuição em pelo menos um dos lados de seu retângulo mínimo. Sejam as seguintes de definições:

- O_c A lista de objetos extraídos do quadro corrente;
- O_p A lista de objetos rastreados pelo sistema;
- O_{kp} Objeto pertencente a lista de objetos rastreados que, no quadro corrente, deu o origem e outros dois objetos(digamos O_{1c} e O_{2c});
- d_{i2} Distância entre os dois novos objetos (O_{1c} e O_{2c});
- $w_{kp} = (w_x, w_y)$ Deslocamento de O_{kp} nos dois últimos quadros;
- w_{vl} Deslocamento da borda inferior do retângulo mínimo de O_{kp} nos dois últimos quadros;
- w_{vu} Deslocamento da borda superior do retângulo mínimo de O_{kp} nos dois últimos quadros;
- w_{hr} Deslocamento da borda direita do retângulo mínimo de O_{kp} nos dois últimos quadros;
- w_{hl} Deslocamento da borda esquerda do retângulo mínimo de O_{kp} nos dois últimos quadros;
- t_w limiar de deslocamento de objetos;
- t_d limiar de distância entre objetos.

Dois objetos são declarados em estado de oclusão quando, 1) o deslocamento da associação ($O_{kp} \rightarrow O_{1c}$) e o deslocamento de uma das bordas de O_{kp} diferem significativamente e 2) quando o deslocamento de uma das bordas é negativo. Matematicamente, essas duas restrições estão representadas na Equação E.2.

$$\begin{aligned}
&(|w_x - w_{hr}| > t_w) \wedge (w_{hr} < 0) \wedge (d_{12} < t_d) \vee \\
&(|w_x - w_{hl}| > t_w) \wedge (w_{hl} > 0) \wedge (d_{12} < t_d) \vee \\
&(|w_y - w_{vl}| > t_w) \wedge (w_{vl} < 0) \wedge (d_{12} < t_d) \vee \\
&(|w_y - w_{vu}| > t_w) \wedge (w_{vu} > 0) \wedge (d_{12} < t_d)
\end{aligned} \tag{E.2}$$

De maneira semelhante à segunda verificação de para detecção de oclusão, adicionou-se mais uma segunda etapa de verificação de divisão de objetos. A segunda etapa de verificação de divisão consiste em procurar por objetos rastreados que não possuem associação e que fazem interseção com pelo menos dois objetos do quadro corrente que também não possuam associação.

E.7 Matrizes utilizadas no Filtro de Kalman

Conforme apresentado no Capítulo 3, o Filtro de Kalman possui duas matrizes principais que definem a transição de estados e as transformações das medições feitas. Abaixo seguem as equações de transição e medição seguidas de suas versões instanciadas com suas respectivas matrizes.

$$x_{k+1} = Ax_k + w_k \tag{E.3}$$

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \Delta x_{k+1} \\ \Delta y_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ \Delta x_k \\ \Delta y_k \end{bmatrix} A + w_k \tag{E.4}$$

$$z_k = Hx_k + v_k \tag{E.5}$$

$$\begin{bmatrix} z_{x_k} \\ z_{y_k} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ \Delta x_k \\ \Delta y_k \end{bmatrix} + v_k \tag{E.6}$$

Apêndice F

Módulo de Detecção de Eventos

Na Figura F.1 está ilustrado o fluxograma para o algoritmo detector de abandono e remoção de objetos.

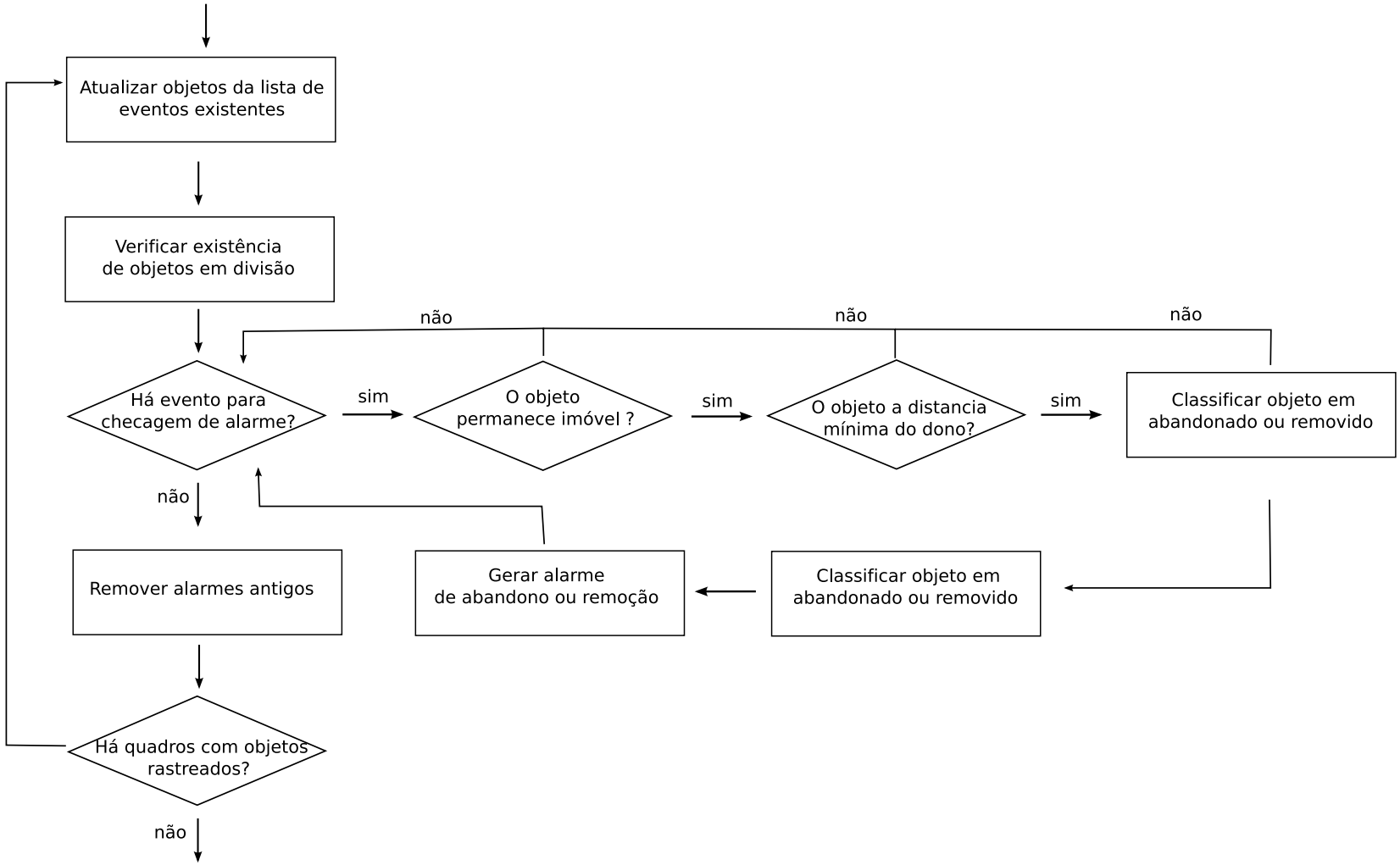


Figura F.1: Fluxograma do algoritmo de detecção de eventos utilizado neste trabalho.

Referências Bibliográficas

J. K. Aggarwal and Q. Cai. Human motion analysis: A review. *Computer Vision and Image Understanding*, 73(3):428–440, 1999.

J. K. Aggarwal and S. Park. Human motion: modeling and recognition of actions and interactions. In *Proceedings of Second International Symposium on 3D Data Processing, Visualization and Transmission*, pages 640–647, Thessaloniki, Greece, september 2004.

A. Amer. Voting-based simultaneous tracking of multiple video objects. *IEEE Transaction on Circuits and Systems for Video Technology*, 15(11):1448–1462, November 2005.

E. Auvinet, E. Grossmann, C. Rougier, M. Dahmane, and J. Meunier. Left-luggage detection using homographies and simple heuristics. In *Proceedings 9th IEEE International Workshop on Performance Evaluation in Tracking and Surveillance (PETS 06)*, pages 51–58, New York, NY, USA, June 2006.

J. L. Barron, D.J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

A. F. Bobick. action: the role of knowledge in the perception of motion. *Royal Society Workshop on Knowledge-based Vision in Man and Machine*, 352:1257–1265, 1997.

A. F. Bobick and A. D. Wilson. A state-based technique for the summarization and recognition of gesture. In *ICCV '95: Proceedings of the Fifth International Conference on Computer Vision*, page 382, Washington, DC, USA, 1995. IEEE Computer Society. ISBN 0-8186-7042-8.

T. Chang, S. Gong, and En. Ong. Tracking multiple people under occlusion using multiple cameras. In *Proceedings of Britsh Machine Vision Conference*, Bristol, UK, September 2000. British Machine Vision Association.

R. Collins, A. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, and O. Hasegawa. A system for video surveillance and monitoring: VSAM final report. Technical Report CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University, 2000.

R. Cucchiara, C. Grana, G. Neri, M. Piccardi, and A. Prati. The sakbot system for moving object detection and tracking. *Video-Based Surveillance Systems Computer Vision and Distributed Processing*, pages 145–157, 2001.

Y. Dedeoglu. Moving object detection, tracking and classification for smart video surveillance. Master's thesis, Bilkent University, Faculty of Engineering, August 2004.

S.E. Deering and D.R. Cheriton. Host groups: A multicast extension to the Internet Protocol. RFC 0966, Internet Engineering Task Force, December 1985.

A. M. Elgammal, D. Harwood, and L. S. Davis. Non-parametric model for background subtraction. In *ECCV '00: Proceedings of the 6th European Conference on Computer Vision-Part II*, pages 751–767, London, UK, 2000. Springer-Verlag. ISBN 3-540-67686-4.

D. M. Gavrila. The visual analysis of human movement: a survey. *Computer Vision and Image Understand*, 73(1):82–98, 1999.

C. R. Gonzalez and R. E. Woods. *Digital Image Processing (2nd Edition)*. Prentice Hall, 2nd edition, January 2002. ISBN 0201180758.

W. N. Gonçalves and J. O. Monteiro. Multiple mice tracking using a combination of particle filter and k-means. In *Proceedings of Simopósio Brasileiro de Computação Gráfica e Processamento de Imagem (SIBGRAPI)*, pages 173–178, Belo Horizonte, Minas Gerais, Brasil, October 2007.

I. Haritaoglu, D. Harwood, and L. S. Davis. W4: real-time surveillance of people and their activities. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):809–830, 2000.

J. Heikkilä and O. Silvén. A real-time system for monitoring of cyclists and pedestrians. In *VS '99: Proceedings of the Second IEEE Workshop on Visual Surveillance*, page 74, Washington, DC, USA, 1999. IEEE Computer Society. ISBN 0-7695-0037-4.

T. Horprasert, D. Harwood, and L. S. Davis. A statistical approach for real-time robust background subtraction and shadow detection. volume 99, pages 1–19, 1999.

J. A. Humphreys. The automated tracking of vehicle and pedestrians in cctv for use in the detection of novel behavior. Master's thesis, Durham University, Department of Cpomputer Science, August 2004.

J. C. S. Jacques, C. R. Jung, and S. R. Musse. Background subtraction and shadow detection in grayscale video sequences. In *Proceedings of Brazilian Simposium on Computer Graphics and Image Processing (SIBGRAPI)*, pages 189–196. IEEE Computer Society, 2005.

P. Kaewtrakulpong and R. Bowden. An Improved Adaptive Background Mixture Model for Real time Tracking with Shadow Detection. In *Proceedings of European Workshop on Advanced Video Based Surveillance Systems*, 2001.

P. H. Kelly, A. Katkere, D. Y. Kuramura, S. Moezzi, and S. Chatterjee. An architecture for multiple perspective interactive video. In *MULTIMEDIA '95: Proceedings of the third ACM international conference on Multimedia*, pages 201–212, New York, NY, USA, 1995. ACM. ISBN 0-89791-751-0.

S. Khan, O. Javed, Z. Rasheed, and M. Shah. Human tracking in multiple cameras. *Internacional Conference on Computer Vision*, 01:331, 2001.

A. C. Lara. Segmentação de movimento usando morfologia matemática. In *Proceedings of Simpósio Brasileiro de Computação Gráfica (SIBGRAPI)*, Belo Horizonte, Minas Gerais, Brasil, October 2007.

C. Lin, H. Nein, and W. Lin. A space-time delay neural network for motion recognition and its application to lipreading. *International Journal of Neural Systems*, 9(4):311–334, 1999.

A. J. Lipton, H. Fujiyoshi, and R. S. Patil. Moving target classification and tracking from real-time video. pages 8–14, Princeton, NJ, USA, 1998. IEEE Computer Society. ISBN 0-8186-8606-5.

P. Maybeck, S. Siouris, and George M. Stochastic models, estimation, and control, volume i. *IEEE Transactions on Systems, Man and Cybernetics*, 10(5):90–126, May 1980.

T. B. Moeslund and A. Hilton. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2):90–126, November 2006.

A. Mohan. Example-based object detection in images by components. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:349–361, 2001.

J. C. Nascimento and J. S. Marques. Performance evaluation of object detection algorithms for video surveillance. *IEEE Transactions on Multimedia*, 8(4):761–774, 2006.

R. Polana and R. Nelson. Low level recognition of human motion (or how to get your manwithout finding his body parts). In *Proceedings of the 1994 IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pages 77–82, 1994.

A. Prati, I. Mikic, M. M. Trivedi, and R. Cucchiara. Detecting moving shadows: Algorithms and evaluation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(7):918–923, 2003.

J. R. Renno, L. N. Mcmanus, D. Makris, and G. A. Jones. Evaluating motion detection algorithms: Issues and results. In *Proceedings of IEEE International Workshop on Visual Surveillance*, pages 97–104, Graz, Austria, 2006.

P. C. Ribeiro and J. Santos-Victor. Human activity recognition from video: modeling, feature selection and classification architecture. In *Proceedings of HAREM 2005 - International Workshop on Human Activity Recognition and Modelling*, 2005.

J. Rincón, J. E. Herrero-Jaraba, J. R. Gómez, and C. Orrite-Uruñuela. Automatic left luggage detection and tracking using multi-camera ukf. In *Proceedings 9th IEEE International Workshop on Performance Evaluation in Tracking and Surveillance (PETS 06)*, pages 59–66, New York, NY, USA, June 2006.

S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2nd edition edition, 2003. ISBN 0137903952.

N. T. Siebel. *Design and Implementation of People Tracking Algorithms for Visual Surveillance Applications*. PhD thesis, Department of Computer Science, The University of Reading, Reading, UK, March 2003.

M. S. Smith and J. M. Brady. Asset-2: Real-time motion segmentation and shape tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):814–820, 1995.

C. Stauffer and L. E. W. Grimson. Adaptative background mixture models for real-time tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR 99)*, volume 2, pages 246–252, Fort Collins, USA, 1999.

R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation*, 3(4):323–344, 1987.

A. Utsumi, H. Mori, J. Ohya, and M. Yachida. Multiple-view-based tracking of multiple humans. In *ICPR '98: Proceedings of the 14th International Conference on Pattern Recognition-Volume 1*, page 597, Washington, DC, USA, 1998. IEEE Computer Society. ISBN 0-8186-8512-3.

C. Wang and M. Brandstein. A hybrid real-time face tracking system. *Acoustics, Speech, and Signal Processing, 1998. ICASSP'98. Proceedings of the 1998 IEEE International Conference on*, 6, 1998.

L. Wang, W. Hu, and T. Tan. Recent developments in human motion analysis. *Pattern Recognition*, 36(3):585–601, 2003.

G. Welch and G. Bishop. An introduction to the kalman filter. Technical report, University of North Carolina, Chapel Hill, NC, USA, 1995.

J. Yamato, J. Ohya, and K. Ishii. Recognizing human action in time-sequential images using hidden markov model. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 379–385, 1992.