UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS GRADUAÇÃO EM INFORMÁTICA

SEGMENTAÇÃO DE VÍDEOS PARA STORYTELLING INTERATIVO BASEADO EM VÍDEO

DISSERTAÇÃO DE MESTRADO

Victor Chitolina Schetinger

Santa Maria, RS, Brasil

2013

SEGMENTAÇÃO DE VÍDEOS PARA STORYTELLING INTERATIVO BASEADO EM VÍDEO

por

Victor Chitolina Schetinger

Dissertação apresentada ao Programa de Pós Graduação em Informática da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para a obtenção do grau de

Mestre em Ciência da Computação

Orientador: Prof. Dr. Cesar Tadeu Pozzer

Dissertação de Mestrado N. 0085/2013

Santa Maria, RS, Brasil

2013

Universidade Federal de Santa Maria
Centro de Tecnologia
Programa de Pós Graduação em Informática

A Comissão Examinadora, abaixo assinada,
aprova a Dissertação de Mestrado

SEGMENTAÇÃO DE VÍDEOS PARA STORYTELLING INTERATIVO BASEADO EM VÍDEO

elaborada por
Victor Chitolina Schetinger

como requisito parcial para obtenção do grau de
Mestre em Ciência da Computação

COMISSÃO EXAMINADORA:

Prof. Dr. Cesar Tadeu Pozzer
(Presidente/Orientador)

Prof. Dr. José Antônio Trindade Borges da Costa (UFSM)

Prof. Dr. Bruno Feijó (PUC-RJ)

Santa Maria, 28 de Fevereiro de 2013.

“Our imagination is stretched to the utmost, not, as in fiction, to imagine things which are not really there, but just to comprehend those things which are there. ”

RICHARD FEYNMAN, "THE CHARACTER OF PHYSICAL LAW" (1965)

AGRADECIMENTOS

A minha mãe, **Maria Rosa**, e minhas irmãs **Christina**, **Luísa** e **Isabela** por sempre me apoiarem e me aguentarem por tanto tempo.

Ao meu orientador, **Prof. Cesar Tadeu Pozzer** pela confiança, amizade e pelo auxílio ao longo dos anos.

A minha namorada, **Juliana**, pela força e pelo carinho.

Aos meus grandes amigos **Cristiano**, **João Heitor**, **Lucas**, **Matheus** e **Vinícius**, que apesar da distância sempre dão um jeito de fazer um churrasco na casa do Dizzy.

Aos professores e a todos os colegas que fizeram parte do **Laboratório de Computação Aplicada**, pela colaboração, companheirismo e madrugadas de trabalho e jogo no CT.

A **CAPES** e professores envolvidos no projeto **RH TV Digital**, pela oportunidade de desenvolver esse trabalho de mestrado.

E, finalmente, agradeço ao restante de minha família e a todos aqueles que de uma forma ou de outra me auxiliaram ou contribuíram pra a realização desse trabalho.

RESUMO

Dissertação de Mestrado
Programa de Pós Graduação em Informática
Universidade Federal de Santa Maria

SEGMENTAÇÃO DE VÍDEOS PARA STORYTELLING INTERATIVO BASEADO EM VÍDEO

Autor: Victor Chitolina Schetinger
Orientador: Prof. Dr. Cesar Tadeu Pozzer
Local e data da defesa: Santa Maria, 28 de Fevereiro de 2013.

O storytelling interativo baseado em vídeo tem como objetivo a geração de narrativas, permitindo o controle de um usuário sobre o rumo da estória e utilizando composições cinematográficas para dramatizá-la. Para que este processo seja possível, existe a necessidade de uma grande quantidade de conteúdo cinematográfico na forma de filmagens. Este conteúdo, por sua vez, precisa ser adequadamente pré-processado para permitir sua utilização adequada. Nesse trabalho, uma abordagem para segmentação de vídeos para storytelling interativo baseado em vídeo é proposta, utilizando alpha matting para extrair informações de cor e transparência de elementos de vídeos para serem reutilizados em processos de dramatização. A solução desenvolvida utiliza técnicas de subtração de fundo e minimização de energia para gerar trimaps de forma automática.

Palavras-chave: Segmentação de vídeos; Visão computacional; Alpha matting; Processamento de vídeos; Storytelling; Logtell.

ABSTRACT

Dissertação de Mestrado
Programa de Pós-Graduação em Informática
Universidade Federal de Santa Maria

VIDEO SEGMENTATION FOR VIDEO-BASED INTERACTIVE STORYTELLING
Author: Victor Chitolina Schetinger
Advisor: Prof. Dr. Cesar Tadeu Pozzer

Video-based interactive storytelling has as its main goal the generation of interactive narratives, allowing users to control the course of the story and using cinematographic compositions to dramatize it. For this process to be possible, there is a need for large amounts of cinematographic content in the form of filmed scenes. This content, on the other hand, has to be properly pre-processed in order to be usable. This work proposes an approach for video segmentation aimed for video-based interactive storytelling, using alpha matting to extract color and transparency of video elements to be later recomposed for dramatization purposes. The developed solution uses background subtraction and energy minimization techniques to automatically generate trimaps.

Keywords: Video Segmentation, Computer Vision, Video-based Interactive Storytelling, Alpha Matting.

LISTA DE FIGURAS

1.1 Exemplo de arquitetura de storytelling interativo baseado em vídeo. 16

3.1 Elementos envolvidos no processo de alpha matting. a) imagem original. b) trimap da imagem. c) alpha matte. d) recomposição do objeto extraído em outro fundo. 26
3.2 Exemplos de trimaps diferentes e suas imagens de origem. (ALPHA MATTING EVALUATION WEBSITE, 2012) 28
3.3 Diferença na estimativa de transparência para a mesma imagem ao utilizar trimaps diferentes. (ALPHA MATTING EVALUATION WEBSITE, 2012) . 29

4.1 Exemplo de segmentação por movimento baseada em diferença. (BOBICK; DAVIS, 1996) 33
4.2 Comparação de técnicas de segmentação frente/fundo baseadas em MOG e codebooks. (KIM et al., 2005) 34

5.1 Representação diagramática do processo geral de segmentação. 36
5.2 Diferença entre gamuts de espaços de cores CMYK e RGB em relação ao espectro visível. 41
5.3 Exemplo de amostras de cor diferentes sob uma perspectiva espacial. a) Paleta de cores observada. b) distribuição de pontos de cor em um espaço RGB pertencentes às classes de cor da paleta. (KIM et al., 2005) 42
5.4 Representação gráfica do espaço CIELAB. (SALM et al., 2004) 44
5.5 Diagrama de MacAdam demonstrando cores perceptualmente semelhantes em um espaço CIEXYZ. (WYSZECKI; STILES, 2000) 45
5.6 Exemplos de figuras contendo uma composição de objetos com e sem transparência sobre fundos vermelhos. 49
5.7 Exemplos de fundos vermelhos diferentes. 49
5.8 Comparação de dois mapas de distância com ordem de treinamento dos frames de fundo diferente. a) treinamento realizado com as imagens na ordem 5.7a, 5.7b e 5.7c. b) treinamento realizado na ordem 5.7b, 5.7a e 5.7c 53
5.9 Representação do grafo entre dois píxeis p e q usado no algoritmo de expansão alfa. (LOMBAERT, 2006) 57
5.10 Representação do grafo entre dois píxeis p e q usado no algoritmo de trocas alfa. (LOMBAERT, 2006) 58

7.1 Comparação da imagem original com os trimaps gerados no caso de teste ideal. a) fundo original. b) trimap gerado com distâncias no espaço de cor CIELAB. c) trimap gerado com distâncias no espaço de cor RGB. 68

7.2 Comparação de imagens de teste de alta resolução com parâmetro flexível e seus trimaps gerados. a) Imagem simples de alta resolução. b) imagem de alta resolução com adição de efeitos de composição, ruído e iluminação. c) trimap gerado com distâncias no espaço de cor CIELAB para a imagem (a). d) trimap gerado com distâncias no espaço de cor CIELAB para a imagem (b). e) fundo usado para o treinamento de ambos os testes, corresponde ao fundo da imagem (a). 70
7.3 Demonstração do efeito de variações no fundo em imagens fotográficas. a) imagem de entrada. b) Fundo usado. c) mapa de distâncias calculado usando o espaço de cor LAB. d) Trimap resultante. 72
7.4 Demonstração do efeito de variações na iluminação em imagens fotográficas. a) imagem de entrada. b) Fundo usado. c) mapa de distâncias calculado usando o espaço de cor LAB. d) Trimap resultante. 73
7.5 Trimaps gerados para ambos os cenários de teste usando os parâmetros da tabela 7.6, ajustada especificamente para o conjunto de imagens de entrada e fundo da figura 7.3. 74
7.6 Comparação de mapas de distâncias entre o quadro da imagem (d) e um modelo do fundo visto na imagem (e), sob diferentes parâmetros KLCH. a) KLCH = {0.8,1,1}. b) KLCH = {1,1,1}. c) KLCH = {0.8,1,1}. 75
7.7 Exemplo de teste de vídeo realizado em resolução 240p. a-c) Quadros de entrada. d-f) Mapas de distância. g-i) Trimaps gerados. 76
7.8 Exemplo de teste de vídeo com fatores adversos. a-c) Quadros de entrada. d-f) Mapas de distância. g-i) Trimaps gerados. 77
7.9 Teste de vídeo sobre fundo simplificado. a) Quadro de entrada. b) Mapa de distâncias. c) Trimap gerado. 79
7.10 Teste de vídeo sob condições ideais. a) Quadro de entrada. b) Trimap gerado. 80
7.11 Aplicação de alpha matting nos testes de imagens de baixa resolução. a) Imagem original. b) Trimap gerado. c) Alpha matte estimado. d) Imagem extraída. 81
7.12 Aplicação de alpha matting nos testes de imagens de alta resolução. a) Imagem original. b) Trimap gerado. c) Alpha matte estimado. d) Imagem extraída. 82
7.13 Aplicação de alpha matting no teste de vídeo realizado em resolução 240p. a-c) Quadros de entrada. d-f) Trimaps gerados. g-i) Alpha mattes estimados. j-l) Imagens extraídas. 83
7.14 Aplicação de alpha matting no caso de teste sob condições adversas. a) Imagem original. b) Trimap gerado. c) Alpha matte estimado. d) Imagem extraída. 84
7.15 Aplicação de alpha matting no teste da imagem 7.9. a) Imagem original. b) Trimap gerado. c) Alpha matte estimado. d) Imagem extraída. 85
7.16 Aplicação de alpha matting no teste da imagem 7.9. a) Imagem original. b) Trimap gerado. c) Alpha matte estimado. d) Imagem extraída. 86

LISTA DE TABELAS

5.1 Mapas de distâncias gerados para as imagens da figura 5.6 sob configurações diferentes das imagens da figura 5.7, no espaço de cores RGB..... 51
5.2 Mapas de distâncias gerados para as imagens da figura 5.6 sob configurações diferentes das imagens da figura 5.7, no espaço de cores LAB..... 52
5.3 Custos de energia relativos entre rótulos vizinhos..... 56

7.1 Exemplo de tabela de parâmetros..... 64
7.2 Tabela de parâmetros para o teste das imagens da subseção 5.2..... 65
7.3 Trimaps gerados para os diferentes mapas de distância das imagens da tabela 5.1, no espaço de cores RGB. 66
7.4 Trimaps gerados para os diferentes mapas de distância das imagens da tabela 5.2, no espaço de cores LAB. 67
7.5 Tabela de parâmetros para testes com imagens sintéticas de alta resolução. .. 69
7.6 Tabela de parâmetros ajustada para melhor se adequar ao caso de teste da imagem 7.3. 74
7.7 Parâmetros usados para os testes da imagem 7.7..... 76
7.8 Parâmetros usados para os testes da imagem 7.8..... 78
7.9 Parâmetros usados para os testes das imagens 7.9 e 7.10. 79
7.10 Tabela de tempos de execução de testes em imagens de resolução 100x100. . 87
7.11 Tabela de tempos de execução de testes em imagens de resolução 1280x720. 87
7.12 Tabela de tempos de execução de testes em vídeos de resolução 320x240.... 88
7.13 Tabela de tempos de execução de testes em vídeos de resolução 640x480.... 88

SUMÁRIO

1 INTRODUÇÃO 13
1.1 Problema 14
1.2 Objetivos 16
1.3 Estrutura do Trabalho 16

2 SEGMENTAÇÃO DE IMAGENS E VíDEOS 18
2.1 Segmentação de Vídeos 19
2.1.1 Segmentação de Vídeos para Storytelling Interativo Baseado em Vídeo 20

3 ALPHA MATTING 23
3.1 Trimaps 27

4 REVISÃO BIBLIOGRÁFICA 30
4.1 Alpha matting para vídeos 30
4.2 Segmentação baseada em detecção de movimento 32
4.3 Segmentação de Frente/Fundo 33

5 PROPOSIÇÃO DO TRABALHO 36
5.1 Treinamento do Fundo 37
5.1.1 Espaços de Cor 40
5.2 Cálculo de Distâncias de Cor 47
5.2.1 Exemplos de mapas de distância 48
5.3 Minimização de Energia 53
5.3.1 Algoritmos de minimização de energia 56
5.4 Aplicação de Alpha Matting 59

6 IMPLEMENTAÇÃO 60

7 RESULTADOS 64
7.1 Testes com imagens 65
7.1.1 Imagens sintéticas 65
7.2 Imagens fotográficas 71
7.3 Testes com vídeos 73
7.4 Aplicação de Alpha Matting 80
7.5 Tempos de Execução 82
7.5.1 Testes com Imagens 86
7.5.2 Testes com Vídeos 87

8 CONCLUSÃO 89
8.1 Contribuições 90
8.2 Limitações e Trabalhos Futuros 91
8.2.1 Implementação com paralelismo 91
8.2.2 Classes de minimização de energia adaptáveis 92
8.2.3 Múltiplas etapas de minimização de energia 92
8.2.4 Ajuste automático de parâmetros 93
8.2.5 Mapa de custos de mudança de rótulos adequado 93
8.2.6 Modelos diferentes de cor e fundo 93
8.3 Considerações Finais 94

REFERÊNCIAS 95

1 INTRODUÇÃO

O campo de estudos em storytelling interativo compreende uma grande gama de assuntos como lógica formal, computação gráfica e cinematografia com o objetivo final de criar narrativas dinâmicas e interativas. Desta forma, pode-se dizer que um hipotético estado da arte seria a geração automática e completa de filmes, permitindo usuários controlar o rumo da estória, aspectos da filmagem e até características dos personagens.

É possível generalizar storytelling interativo em três partes fundamentais: geração de estórias, interação e dramatização. As duas primeiras partes estão relacionadas com o enredo, a estutura lógica e temporal da narrativa, tanto criando conteúdo como permitindo a um usuário influenciar a estória. De forma análoga ao cinema, seria um processo semelhante à criação de um script: um escritor cria um manuscrito inicial e alguma entidade envolvida na direção, como o diretor ou o produtor podem exigir mudanças na estória original. O escritor então deve re-escrever o script mantendo a coerência entre os aspectos mais importantes ao mesmo tempo que incorpora as mudanças exigidas até chegar a um script final. Em storytelling interativo, os papéis do escritor e do diretor podem ser comparados ao sistema e um usuário nesta analogia.

Depois de pronto, um script descreve a estória textualmente. A projeção de um script textual para algum tipo de mídia visual - um filme, por exemplo - é geralmente visto como o processo de dramatização. Nesta projeção, o produto final contém muito mais informação que o original, como informação visual, espacial e temporal, o que implica em duas coisas: primeiramente, estas informações vão ser adicionadas em algum ponto durante a projeção e segundo, a partir do mesmo script original diferentes dramatizações podem emergir.

Em diversos sistemas storytelling interativo ((CAVAZZA; CHARLES; MEAD, 2002), (MATEAS; STERN, 2005) e (POZZER, 2005)) a dramatização é feita através de computação gráfica 3D. Na realidade, a distinção entre geração de script e dramatização é uma abstração que nem sempre reflete a realidade de implementação. Pozzer (POZZER, 2005) claramente separa sua

arquitetura entre um gerador interativo de enredo e um dramatizador que renderiza as cenas em 3D.

A dramatização de uma cena utilizando computação gráfica 3D geralmente tenta simular a filmagem de um filme, com uma câmera sintética e modelos 3D no lugar de atores reais. Este processo não é simples, pois além dos atores virtuais terem que representar corretamente a cena de acordo com o script, movimentando-se, falando e expressando emoções, a câmera precisa ser capaz de filmar tudo isto adequadamente aplicando conceitos de cinematografia (LIMA et al., 2009).

Abordagens para a dramatização de storytelling baseadas em vídeo têm sido pesquisadas recentemente ((PORTEOUS et al., 2010), (URSU et al., 2008)) como alternativa à computação gráfica 3D, pois algumas de suas dificuldades podem ser contornadas utilizando filmagens com atores e cenários reais. O detalhe e realismo das cenas, por exemplo, pode ser tão bom quanto a qualidade das filmagens permitir, juntamente com a atuação física e vocal dos atores. Neste sentido, o storytelling baseado em vídeo pode ser facilmente comparado a uma forma diferente de cinema.

O problema principal de utilizar cenas pré filmadas para dramatizar estórias dinâmicas está na quantidade de conteúdo requerida para satisfazer uma grande gama de eventos. Espera-se de um dramatizador que ele possa dramatizar todo o domínio de cenas que um planejador de enredo possa gerar, o que implicaria na existência prévia de filmagens para todas as combinações possíveis. Por exemplo, se em uma versão da estória o herói poderia discutir sua missão com seu mentor e em outra com o rei, ambas cenas deveriam ser filmadas.

Para contornar este problema, Edirlei (LIMA et al., 2012) propõe um sistema que utiliza segmentos de filmagens de atores e cenários diferentes, os combina em cenas de vídeo únicas e os usa para dramatização. Esta abordagem tem o potencial de diminuir drasticamente a quantidade de conteúdo necessária: atores poderiam ser filmados individualmente no mesmo local e servir como bases para composições. Adicionalmente, seria necessário apenas uma filmagem para cada cenário diferente e o trabalho de juntar dinamicamente todos estes elementos seria realizado sob demanda pelo sistema.

1.1 Problema

A utilização do sistema para compor cenas proposto por Edirlei ainda exige, no entanto, uma grande quantidade de conteúdo filmado. Este conteúdo, ademais, precisa ser adequadamente

processado para isolar os elementos desejados em cada filmagem, sejam atores, objetos ou cenários. Este tipo de segmentação é comumente resolvida com a utilização de um fundo azul em um estúdio e ferramentas de edição de vídeo. No entanto, na proposta apresentada diversos motivos são incompatíveis com essa solução:

- Idealmente as cenas serão filmadas por câmeras de vários ângulos diferentes, tornando mais difícil a aplicação de um fundo azul;
- A criação de conteúdo deve ser uma tarefa fácil e que requeira o mínimo de material possível, de forma que não seja necessário um estúdio profissional;
- A segmentação das imagens deve ter tanta qualidade quanto possível, incluindo informações de transparência;
- Devido à quantidade esperada de conteúdo a ser processada, o trabalho do usuário no processo de segmentação deve ser o menor possível.

Além dessas exigências, algumas especificações limitam o escopo esperado para segmentação:

- Todas as filmagens são realizadas com câmera estática;
- Espera-se trabalhar com resoluções altas, 720p ou superiores;
- O processo de segmentação não precisa ser realizado em tempo real e não existe uma preocupação grande com desempenho;
- Geralmente um ator deve ser segmentado por cena, compreendendo sua indumentária e objetos envolvidos na atuação.

Na figura 1.1 está representada uma possível arquitetura dos processos envolvidos em storytelling interativo baseado em vídeo. Inicialmente, filmagens são realizadas e conteúdo é produzido. Este conteúdo, após ser devidamente segmentado serve para alimentar um banco de conteúdo dramatizável. Essa parte do processo, vista na parte de cima da figura, é realizado em offline. Na parte inferior da figura, independentemente, ocorre o processo da geração das estórias e interação com o usuário. Um gerador de enredo interativo cria scripts descrevendo a estória e suas cenas que servem como entrada para o dramatizador. Por sua vez, o dramatizador utiliza o conteúdo disponível para gerar um segmento de vídeo representando a cena, que é então exibida para o usuário.

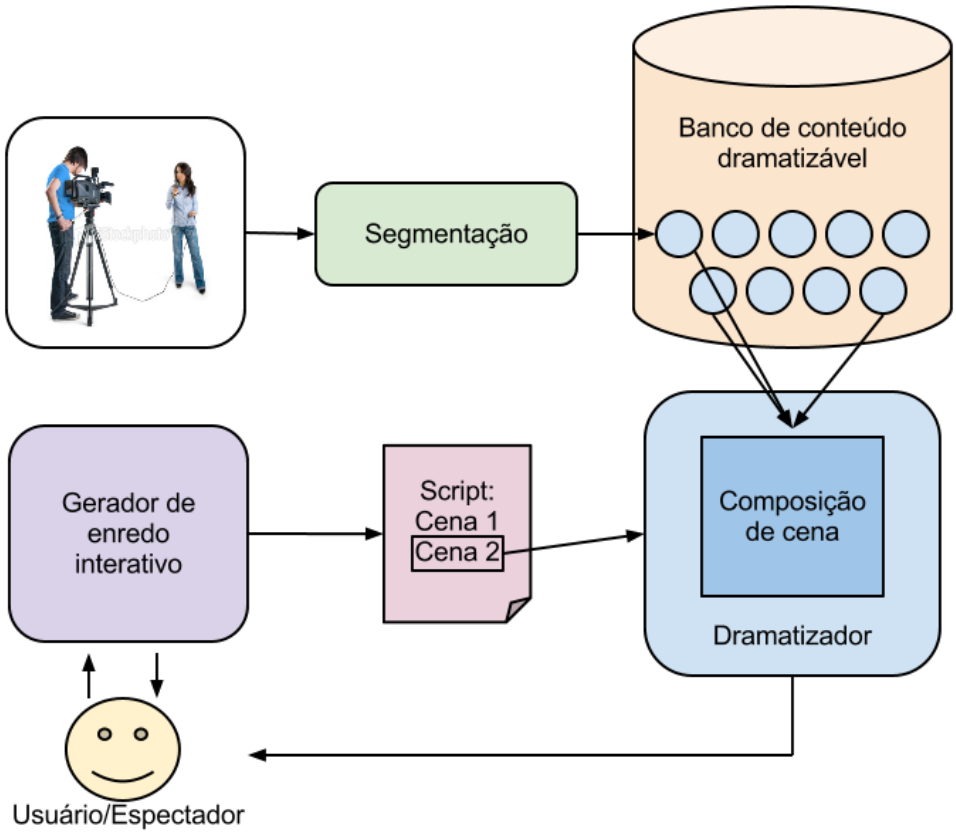


Figura 1.1: Exemplo de arquitetura de storytelling interativo baseado em vídeo.

1.2 Objetivos

Esse trabalho tem como objetivo principal desenvolver uma técnica de segmentação de vídeo para ser utilizada em storytelling interativo baseado em vídeo. A solução desenvolvida precisa ser capaz de abordar os problemas discutidos na seção anterior, adequando-se ao escopo das possíveis aplicações. Mais especificamente, são objetivos o desenvolvimento de métodos que permitam:

- A extração de atores de um vídeo com sua informação de transparência e cor absoluta;
- A utilização de filmagens realizadas de forma semi-profissional ou amadora;
- A facilidade de utilização por usuários, com o mínimo de intervenção; e
- O detalhamento de resultados, preferencialmente com resoluções altas.

1.3 Estrutura do Trabalho

O ponto de partida desse trabalho é o problema descrito na seção 1.1, que está inserido dentro da vasta área de problemas de segmentação de imagens e vídeos. Por este motivo, no

capítulo 2 são introduzidos os conceitos fundamentais de segmentação de imagens e vídeos, havendo um foco nas particularidades da segmentação de vídeos e na contextualização do problema abordado por esse trabalho.

Ao final do capítulo 2 é apresentado o problema de extração de informação de transparência em segmentação de vídeos e imagens. Para abordar este problema, esse trabalho utiliza o processo de alpha matting, de modo que este é tratado no capítulo 3. Neste capítulo são descritas as definições fundamentais de alpha matting e suas principais técnicas. A seção 3.1 explica o que são trimaps, sua importância em relação ao processo de alpha matting e define-se o problema a ser tratado como um problema de extração de trimaps.

A revisão bibliográfica é realizada no capítulo 4. Inicialmente são discutidos trabalhos que aplicam alpha matting diretamente à vídeos, bem como sua aplicabilidade à proposta desse trabalho. Em seguida, são avaliados trabalhos de duas sub-áreas diferentes de segmentação de vídeos: segmentação baseada em detecção de movimento e segmentação de frente/fundo.

A proposição do trabalho é apresentada no capítulo 5, juntamente com a arquitetura geral da solução desenvolvida. As seções desse capítulo descrevem os diferentes passos do processo de segmentação desenvolvido, com um aprofundamento maior em suas subseções. A implementação destes conceitos é explicada no capítulo 6.

O capítulo 7 apresenta os resultados obtidos utilizando a solução proposta. Devido ao grande número de parâmetros envolvidos e a natureza abrangente da arquitetura adotada, são mostrados resultados utilizando diversas configurações - inicialmente com imagens e em seguida com vídeos.

Finalmente, as conclusões obtidas com este trabalho estão presentes no capítulo 8. Neste capítulo também são listadas as contribuições, as limitações da solução atual e sugestões para trabalhos futuros.

2 SEGMENTAÇÃO DE IMAGENS E VÍDEOS

A segmentação é uma das tarefas fundamentais envolvidas no processamento de imagens e visão computacional. Seu objetivo principal é realizar a extração de informação de uma imagem na forma dos seus objetos ou elementos que a compõem. A segmentação de vídeos pode ser considerada um caso especial da segmentação de imagens, com peculiaridades específicas mas fundamentalmente equivalente, pois um vídeo é composto de diversos quadros ou imagens.

Utilizando a definição clássica de segmentação descrita por Haralick e Shapiro (HARALICK; SHAPIRO, 1985) como um processo de partição da imagem em um conjunto de regiões não sobrepostas na qual a sua união é a imagem total, estas devem seguir as seguintes regras:

1. As regiões devem ser uniformes e homogêneas em relação a certas características;
2. Seus interiores devem ser simples e sem muitos buracos pequenos;
3. Regiões adjacentes devem apresentar diferenças nas características que estão sendo consideradas; e
4. Os limites das regiões devem ser simples e espacialmente precisos.

As regras 1 e 3 são as mais expressivas em relação ao significado da segmentação, especialmente ao serem consideradas as diferentes aplicações possíveis de uma segmentação, como imageamento médico, metalografia ou produção gráfica, que fundamentalmente trabalham com características de imagem diferentes.

A pesquisa em segmentação de vídeos e imagens nos últimos 40 anos tem sido massiva e pouco organizada (ZHANG, 2006a), pois diversas áreas realizam pesquisas em paralelo orientadas às suas aplicações. Classificar a grande quantidade de trabalhos existentes é muito difícil, especialmente porque muitos deles possuem um certo grau de semelhança ou intersecção. Fu (FU; MUI, 1981) propõe uma possível classificação de métodos de segmentação, baseada na sua abordagem:

1. Segmentação por thresholding;
2. Segmentação por classificação de píxeis;
3. Segmentação alcance ou variedade;
4. Segmentação por cor;
5. Segmentação por bordas; e
6. Métodos baseados em teoria dos conjuntos fuzzy.

Esta classificação remete indiretamente à elementos de uma imagem que podem ser analisados ou formas de analisar estes elementos, com o objetivo final de realizar uma segmentação que represente as características desejadas. Para determinar a melhor forma de segmentação para uma aplicação é preciso identificar seus objetivos e problemas envolvidos. Para segmentação de vídeos, por exemplo, um dos objetivos mais comuns é indentificar o mesmo objeto contiguamente através de vários quadros consecutivos, o que pode ser difícil caso a câmera e/ou o objeto se movimentem.

2.1 Segmentação de Vídeos

Como foi comentado na seção anterior, a segmentação de vídeos pode ser considerada uma classe especial de problema de segmentação de imagens. Cada quadro de um vídeo é, essencialmente, uma imagem a ser segmentada individualmente. No entanto, os objetos a serem segmentados em um vídeo possuem uma dimensão temporal, e a segmentação precisa representar consistentemente as mesmas regiões com características semelhantes ao longo de quadros consecutivos (KOPRINSKA; CARRATO, 2001). Dentre os problemas consistentemente encontrados com aplicações de segmentações de vídeo podem ser citados ((ZHANG, 2006b), (ZAPPELLA; LLADó; SALVI, 2008)):

1. A segmentação simultânea de múltiplos objetos é bastante complexa de ser lidada;
2. A movimentação da câmera e dos objetos envolvidos pode alterar drasticamente o posicionamento e a aparência de elementos do vídeo ao longo de quadros consecutivos. Em casos onde o movimento dos objetos não é uniforme pode ser especialmente difícil criar relações temporais;

3. Objetos podem ser ocludidos e revelados posteriormente, de modo que certas regiões podem desaparecer completamente em partes da segmentação;

4. Parâmetros da câmera como foco e exposição ou a iluminação podem variar ao longo do vídeo, afetando toda cena ou apenas algumas partes dela;

5. Vídeos geralmente possuem grande quantidade de ruído, em grande parte devido à natureza dos dispositivos digitais de captura;

6. Vídeos contém em média 25 quadros por segundo, tornando a quantidade de processamento necessária para realizar a sua segmentação muito maior do que de uma imagem;

7. Arquivos de vídeo possuem um tamanho elevado, principalmente em resoluções altas. Em contrapartida, ao utilizar formatos com compactação têm-se perda de informação e adição de ruído e artefatos;

8. A interação com o usuário ou entrada de dados pode ser problemática. Ao requerir entrada em todos os quadros do vídeo a segmentação torna-se uma tarefa árdua e de precisão, pois uma entrada diferente em quadros consecutivos pode afetar o resultado. Em casos onde este tipo de informação é estritamente necessária, solucionar este problema agrega signitificativa complexidade ao processo de segmentação.

Estas dificuldades são incorporadas às já existentes ao processo de segmentação de imagens, tornando a segmentação de vídeos uma tarefa particularmente desafiadora. Por este motivo, é extremamente difícil desenvolver uma solução genérica para segmentação de vídeos que resolva todos estes problemas. Todas as soluções existentes definem aplicações e limitam seu escopo para poder obter resultados objetivos. Na próxima subseção é tratado o escopo específico de segmentação de vídeos desse trabalho, sendo abordados também trabalhos relacionados.

2.1.1 Segmentação de Vídeos para Storytelling Interativo Baseado em Vídeo

Recapitulando as definições desse trabalho descritas na seção 1.1, deve-se fazer a pergunta: "Quais dos problemas comuns de segmentação de vídeos se aplicam nessa situação e de que forma eles se manifestam?". Esta pergunta é importante pois situa o problema a ser tratado dentro do escopo da área, permitindo determinar quais trabalhos relacionados são relevantes e qual a abordagem inicial a ser tomada. Relacionando às questões apresentadas na seção anterior ao contexto desse trabalho, faz-se as seguintes considerações:

1. Na aplicação definida somente um ator será segmentado por cena. Este ator pode estar segurando objetos adicionais ou possuir uma indumentária complexa que também serão segmentados, mas todos estes elementos formarão uma única região conexa;

2. A câmera é estática, mas o ator se movimenta de forma não-regular;

3. Devido à natureza das filmagens, o ator nunca poderá sofrer oclusão direta. Partes do seu corpo e dos objetos utilizados na sua interpretação podem ser parcialmente auto-ocludidos, e precisa ser tratado;

4. Como os vídeos serão filmados especificamente para este fim, é possível estipular parâmetros de câmera invariáveis para uma tomada ou conjunto de tomadas. A iluminação das cenas pode ser facilmente controlada caso as filmagens sejam feitas em interiores, mas em tomadas realizadas no exterior é preciso levar em conta o efeito da iluminação natural;

5. O ruído é um problema grande a ser resolvido, pois espera-se qualidade visual no resultado final da segmentação. Ademais, deve ser possível a obtenção de bons resultados sem a necessidade de se utilizar equipamentos profissionais ou sofisticados para realizar as filmagens;

6. A segmentação será realizada em offline, de modo que não há uma grande preocupação com desempenho;

7. Para a aplicação final serão utilizados vídeos de resolução 720p ou superiores, que apresentam um tamanho considerável. O armazenamento destes vídeos em si não é problemático, mas o uso de memória precisa ser considerado. A utilização de vídeos com compressão não é ideal, pois agrega mais ruído e prejudica a qualidade do resultado final. A maioria das câmeras comerciais atuais, no entanto, não fazem gravações em formato bruto;

8. No modelo de storytelling interativo beaseado em vídeo proposto existe uma necessidade por grandes quantidades de conteúdo na forma de filmagens. Como este conteúdo precisa ser segmentado antes de poder ser utilizado, a praticidade de segmentação é fundamental.

Pode-se condensar estes pontos em três características importantes que definem a solução de segmentação esperada: inicialmente, como não é necessária a execução em tempo-real, pode-se

optar por abordagens com alto custo computacional; em segundo lugar, o uso de uma câmera estática significa que o fundo das filmagens pode ser modelado e estimado; finalmente, a necessidade de praticidade exige que o usuário não precise realizar intervenções no processo.

Para esse trabalho, no entanto, existe também a necessidade de extrair a informação de cor e transparência dos objetos para poder reincorporá-los adequadamente em outras cenas. Esse tipo de problema geralmente sai do escopo de segmentação clássica, e pode ser bastante difícil de ser resolvido. Uma das formas mais estabelecidas na literatura para abordar este problema é o processo de alpha matting, que será discutido em detalhes no capítulo seguinte.

3 ALPHA MATTING

As definições fundamentais do problema de alpha matting foram inicialmente estabelecidas por Porter e Duff em "Compositing Digital Images" (PORTER; DUFF, 1984). Neste trabalho, foi proposta a utilização de informações de opacidade para elementos gráficos, através de um canal adicional alpha para computar a sua influência no resultado final de uma composição. Adicionalmente, esta abordagem foi relevante também para a decomposição, ou segmentação de imagens, que até então era limitada à segmentação binária. Isto permitiu que elementos com transparência ou translúcidos, como vidro, penugem e fumaça pudessem ser corretamente segmentados.

Matematicamente, considera-se que uma determinada imagem possua um plano de frente e um plano de fundo, juntamente com um coeficiente de transparência, chamado de alpha matte. A cor final observada é uma combinação convexa das cores de ambos os planos utilizando o alpha matte como coeficiente de interpolação, seguindo a seguinte equação:

$$C_f = C_1\alpha + C_2(1 - \alpha) \tag{3.1}$$

Os coeficientes C_f, C_1 e C_2 representam as cores final, do plano de frente e do plano de fundo, respectivamente. Esta forma da equação de composição alpha considera apenas dois planos diferentes da imagem, de modo que o único coeficiente é referente à transparência do plano de frente, e assume-se que o plano de fundo é opaco. No caso de múltiplos objetos se sobrepondo, a equação aplica-se recursivamente a partir do fundo e a cada iteração C_f torna-se C_2 para resolver a próxima camada de frente.

Considerando um espaço de cor tridimensional baseado em um modelo de cores aditivo, como RGB, a equação 3.1 corresponde à equação paramétrica de um segmento de reta do ponto C_2 ao ponto C_1, tendo o coeficiente alpha como parâmetro de controle. Os pontos ao longo deste segmento de reta representam as cores que um pixel poderia assumir naquele ponto da

imagem, dependendo da transparência do objeto.

Nesta forma a composição alpha é uma interpolação linear entre dois pontos em um espaço de cor, delimitada entre 0 e 1. Em $\alpha = 1$, o plano de frente é completamente opaco, e a cor final é igual a cor do plano de frente. Ao diminuir o alpha, aumentando a transparência do plano de frente, a cor final torna-se uma combinação linear do plano de frente e de fundo. Nesta representação, $\alpha = 0$ representa transparência total, ou inexistência de um plano de frente, e sua informação de cor torna-se irrelevante.

O cálculo da composição alpha é geralmente realizado pixel a pixel, de modo que apesar de termos ambos os planos, as cores são relativas a cada pixel localmente. Após realizar a combinação dos planos perdemos as informações de cor de frente, de fundo e de alpha, de forma que reverter o processo é um problema muito complexo.

Em fotografias e filmes, uma câmera capta luz do ambiente vindo de diversas fontes diferentes, sofrendo reflexões, refrações e difusões. A cor final de um ponto da imagem é determinada pela combinação de raios de luz que incidem sobre sua posição relativa no dispostivo de captura, como um filme fotográfico ou CCD. Este processo de captura de luz pode ser considerado uma projeção de um espaço tridimensional para um espaço bidimensional, de forma que informações de profundidade e configuração espacial são perdidas.

Ademais, quando capturamos uma fotografia de uma cena contendo objetos transparentes, como vidros, tecido ou cabelo, sua cor resultante é naturalmente composta na imagem final e em momento algum temos a informação das cores de frente, fundo e alpha. Para podermos segmentar e extrair objetos corretamente de imagens, no entanto, é preciso determinar sua transparência real, estimando sua cor e alpha.

O maior problema envolvido no processo de estimar o alpha matte é a grande quantidade de variáveis desconhecidas. Como entrada para o problema geralmente temos uma imagem, e necessita-se encontrar duas imagens (plano de frente e fundo) e finalmente o alpha matte. Se considerarmos um espaço de cor tridimensional como RGB, percebe-se que são três variáveis conhecidas (os três componentes da imagem original) contra sete variáveis desconhecidas (duas triplas de cor RGB juntamente com o alpha matte).

Sem restrições adicionais, o problema de calcular o alpha matte de uma imagem com três variáveis livres e sete desconhecidas possui infinitas soluções. Isso é facilmente notado se considerarmos as diferentes formas que podemos classificar os planos de uma imagem. Sendo a cor final combinação linear de duas outras cores quaisquer, existem infinitas formas de escrever

a equação 3.1:

$$(20, 40, 10) = 0.5*(20, 20, 20)+0.5*(20, 60, 0) = 0.1*(200, 0, 100)+0.9*(0, 44.44, 0) \quad (3.2)$$

Por este motivo, todas as formas de cálculo de alpha mating requerem algum tipo de informação adicional além da imagem de entrada (WANG; COHEN, 2007). Uma das formas mais usadas é a classificação por parte de um usuário em áreas de absoluto plano de frente (alpha igual a 1) e absoluto plano de fundo (alpha igual a 0). Isso pode ser feito através de rabiscos sobre a imagem para indicar as áreas ou através de trimaps, onde uma segmentação total da imagem é feita e áreas desconhecidas são marcadas como cinza. Desta forma, o problema é reduzido à estimar as cores de frente, fundo e matte na área desconhecida utilizando as informações das áreas conhecidas.

A figura 3.1 exemplifica alguns dos elementos envolvidos no processo de alpha matting: a figura 3.1a corresponde à imagem original a ser segmentada; um possível trimap usado para segmentação pode ser visto na figura 3.1b; a figura 3.1c representa seu alpha matte e sua recomposição em um fundo diferente, isolando os objetos desejados com informação de transparência é apresentada na figura 3.1d. Neste exemplo, foi utilizada a técnica de "Shared Matting" (GASTAL; OLIVEIRA, 2010).

Um ponto importante a ser considerado é que a partir do momento em que informação adicional, neste caso na forma de trimaps, está sendo provida, a qualidade e precisão dessa informação afeta o resultado final da estimativa do alpha matte. Idealmente, as regiões desconhecidas do trimap devem estar limitadas somente à áreas onde ocorre sobreposição de planos. No entanto, na prática, esta tarefa iria requerer muito tempo e precisão do usuário, de modo que comumente são usados trimaps menos exatos, como pode ser visto na figura 3.1b.

O problema de alpha matting com informação provida pelo usuário é geralmente abordado de duas formas fundamentais diferentes: por amostragem e/ou por afinidade. Técnicas baseadas em amostragem procuram buscar amostrar de píxeis de frente e fundo conhecidos que possam modelar as características de um píxel desconhecido. A partir de suposições estatísticas, por exemplo, estas técnicas tentam determinar conjuntos de pixels vizinhos que se adequem à equação 3.1. Trabalhos que usam essa abordagem incluem "Alpha Estimation in Natural Images" (RUZON; TOMASI, 2000), "A Bayesian Approach to Digital Matting" (RUZON; TOMASI, 2000) e, mais recentemente "Shared Sampling for Real-Time Alpha Matting" (GASTAL; OLIVEIRA, 2010).

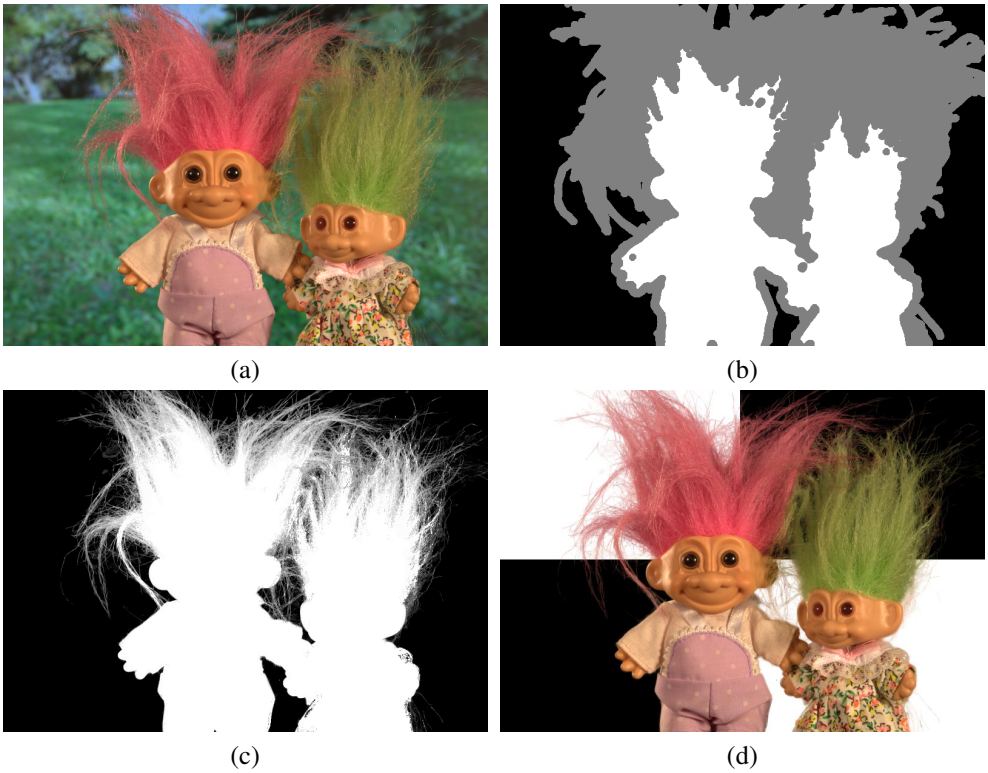


Figura 3.1: Elementos envolvidos no processo de alpha matting. a) imagem original. b) trimap da imagem. c) alpha matte. d) recomposição do objeto extraído em outro fundo.

Abordagens que se baseiam em afinidade consideram que exista uma relação forte entre vizinhanças de pixels e seus parâmetros de matting (cores de frente, fundo e transparência), e que estes são localmente suaves. Desta forma, técnicas de alpha matting por afinidade procuram estimar de algum modo os gradientes dos planos da imagem para obter o resultado final. Técnicas como "Poisson Matting" (SUN et al., 2004), "Spectral Matting" (LEVIN; RAV ACHA; LISCHINSKI, 2008) e "Geodesic Matting" (BAI; SAPIRO, 2009) se enquadram nesta categoria.

Alpha Matting é um tema bastante pesquisado nos últimos anos (WANG; COHEN, 2007), de modo que para melhor poder avaliar as qualidades de cada trabalho Rhemman e colegas (CHRISTOPH RHEMANN CARSTEN ROTHER, 2009) estabeleceram um padrão de utilização e testes. Na página "Alpha Matting Evaluation Website" (ALPHA MATTING EVALUATION WEBSITE, 2012) este teste está disponível e autores podem submeter seus métodos para atestar a qualidade do seu trabalho. Este teste utiliza conjuntos de imagens de testes e trimaps de entrada, juntamente com alpha mattes absolutamente precisos para comparação de resultados.

Esta conjuntura atual da pesquisa em alpha matting permite a sua utilização para segmentação de imagens com abstração em relação à técnica utilizada, através do uso de trimaps. Isto é ideal para a proposta desse trabalho, pois permite uma redução significativa de escopo além

da possibilidade de modularização. Ao separar o processo de segmentação em duas etapas, inicialmente uma segmentação ternária na forma de trimaps e em seguida a aplicação de uma técnica de trimap, estabelece-se uma arquitetura flexível.

3.1 Trimaps

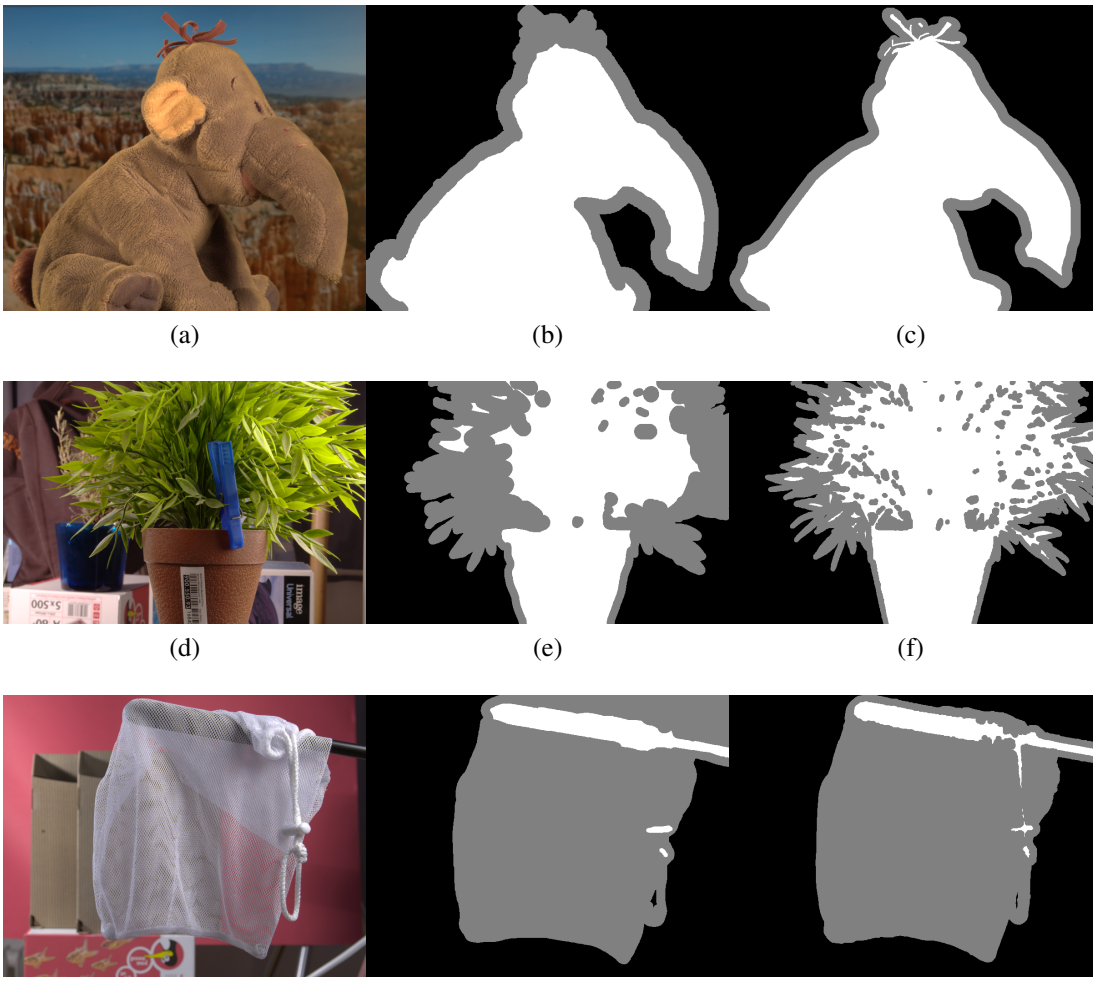
Ao utilizar alpha matting para a extração de cor e informações de transparência de vídeos, faz-se necessária a geração de trimaps para servirem como entrada ao processo, como visto na figura 3.1. Um trimap é uma imagem em tons de cinza, que ao ser sobreposta sobre sua imagem original representa regiões onde se tem conhecimento sobre o plano de frente e fundo.

A figura 3.2 apresenta conjuntos de trimaps com suas imagens de origem. As regiões marcadas em branco e preto são regiões consideradas totalmente pertencentes ao plano de frente, ou fundo, respectivamente. Considera-se que estas regiões sejam absolutamente opacas, ou seja, $\alpha = 1$ ou $\alpha = 0$. A região cinza é chamada região desconhecida, e indica onde o processo de estimativa de transparência deve realmente ocorrer.

Cada imagem de origem pode ser representada por trimaps diferentes, mais ou menos precisos. A precisão de um trimap pode afetar profundamente seu resultado, como mostra a figura 3.3. As figura 3.3b e 3.3a representam as estimativas de transparência obtidas utilizando Shared Matting (GASTAL; OLIVEIRA, 2010) na imagem 3.2a com os trimaps 3.2b e 3.2c, respectivamente. O trimap da imagem 3.2c é mais detalhado, indicando corretamente os fios de cabelo sobre a cabeça do elefante que pertencem à frente. Como resultado, esta região é corretamente identificada como opaca, e suas bordas com certa transparência. Na imagem 3.3a à mesma região foi atribuída transparência de forma errônea e diversos artefatos podem ser vistos no contorno geral do objeto.

As imagens 3.2d e 3.2g apresentam objetos muito mais difíceis de serem delimitados por um trimap. O primeiro objeto por possuir muitos recortes e regiões irregulares, enquanto o segundo por ser em grande parte semi-transparente, com poucas regiões claras de frente ou fundo. Da mesma forma que na imagem do elefante, a precisão do trimap vai incidir no resultado final, o que pode ser problemático nestes casos.

A maioria das técnicas de alpha matting não leva em consideração o processo de obtenção dos trimaps, assumindo que sejam parte da entrada a ser provida pelo usuário. Para a utilização de alpha matting nesse trabalho, no entanto, são necessários trimaps para todos os frames de vídeos. Deste modo, a geração de trimaps para vídeos é um dos objetivos centrais.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figura 3.2: Exemplos de trimaps diferentes e suas imagens de origem. (ALPHA MATTING EVALUATION WEBSITE, 2012)

(a)

(b)

Figura 3.3: Diferença na estimativa de transparência para a mesma imagem ao utilizar trimaps diferentes. (ALPHA MATTING EVALUATION WEBSITE, 2012)

4 REVISÃO BIBLIOGRÁFICA

Como apresentado nos capítulos 2 e 3, existe uma quantidade vasta de pesquisa em segmentação de vídeos e imagens e alpha matting. Nesta seção serão abordados os trabalhos que possuem uma estrita relação à esse, em duas formas principais: trabalhos que tratam diretamente com aplicação de alpha matting para vídeos e trabalhos de segmentação de vídeos que podem ser usados para a obtenção de trimaps.

Na literatura, problemas de segmentação de vídeos onde a câmera é estática geralmente são tratados de duas formas: através de detecção de movimento ou por segmentação de frente/fundo (também chamada de subtração de fundo). Cada uma destas abordagens é tratada separadamente a seguir, bem como sua aplicabilidade para resolver o problema central desse trabalho.

4.1 Alpha matting para vídeos

A aplicação de alpha matting para vídeos também é uma área de pesquisa emergente, especialmente pelas suas aplicações em mídia. Este processo sofre das mesmas limitações e dificuldades da segmentação de vídeos, sendo que seu resultado final apresenta muito mais detalhes do que uma simples segmentação, de modo que há maior margem para erros. A forma mais comum de se abordar este problema é a separação do processo em duas partes, inicialmente uma segmentação da entrada em trimaps e em seguida a aplicação de alpha matting (WANG; COHEN, 2007).

No caso deste tipo de arquitetura, uma das maiores dificuldades é garantir a coesão temporal dos trimaps ao longo de quadros. Interpolação entre quadros utilizando fluxo óptico é uma das formas utilizadas para solucionar esse problema, como nos trabalhos de Black (BLACK; ANANDAN, 1996), Szeliski (SZELISKI; SHUM, 1997) e Chuang (CHUANG, 2002). Este tipo de abordagem, no entanto, exige que um usuário entre com trimaps completos para alguns quadros chave do vídeo, além de apresentar uma precisão baixa quando existe grande variação

entre quadros.

Rotoscoping, que em produção cinematográfica clássica originalmente remete ao processo de se traçar desenhos sobre filme, foi recentemente adaptado para aplicações interpolação de contornos em vídeos. No trabalho "Keyframe-based tracking for rotoscoping and animation" (AGARWALA, 2004) uma técnica que permite um usuário traçar contornos de objetos ao longo quadros subsequentes é proposta. Inicialmente, modela-se o contorno do objeto desejado em um quadro inicial e um final utilizando curvas paramétricas. Em seguida, a interpolação das curvas nos quadros intermediários é realizado como um problema de otimização através de minimização de energia, resultando em uma sequência de contornos. As duas maiores limitações desta técnica são a necessidade de um usuário modelar curvas e sua incapacidade de lidar com mudanças na topologia do objeto, como pode ocorrer em casos de oclusão ou auto-oclusão.

A técnica de Graph-cut (BOYKOV; FUNKA-LEA, 2006) utilizada em segmentação de imagens foi adaptada para geração de trimaps para vídeos nos trabalhos de Yi (LI; SHUM, 2005) e Wang (J. WANG P. BHAT; COHEN, 2005). Estes trabalhos apresentam avanços significativos em termos de usabilidade, permitido a um usuário realizar segmentações interativas. A arquitetura das soluções propostas por ambos é bastante extensa e complexa, buscando robusteza e refinamento de resultados para vídeos de entrada genéricos.

Bai e Sapiro propõem um framework capaz de realizar o processo de alpha matting tanto em imagens quanto em vídeos (BAI; SAPIRO, 2009). Neste trabalho distâncias geodésicas são usadas para modelar proximidades de píxeis espacialmente e em relação à similaridade de cores. No caso de vídeos, as distâncias geodésicas são extendidas para um modelo volumétrico ao longo do tempo. Os resultados apresentados são muito promissores, para ambos os casos.

Wang (WANG et al., 2012) apresenta um algoritmo para obtenção automática de alpha matting utilizando uma câmera TOF (Time-of-Flight). Esta câmera é um dispositivo especial capaz de obter um mapa das distâncias físicas juntamente com a imagem, de modo que estas informações são utilizadas para a geração de trimaps. Sua técnica é bastante promissora, além de ser em tempo real e automática. A necessidade de um equipamento especial, no entanto, limita a sua aplicabilidade.

De uma forma geral, métodos atuais de alpha matting para vídeos demonstram resultados muito bons, normalmente utilizando um processo dividido em extração de trimaps seguido por aplicação de uma técnica de alpha matting. Uma das limitações visíveis nos trabalhos que usam essa abordagem é a impossibilidade de extrair corretamente objetos com uma grande quantidade

de píxeis transparentes, pois a geração do trimap geralmente é feita através da propagação da região desconhecida nas bordas externas dos objetos. Ademais, como muitas das aplicações dos trabalhos são para propósito geral, há uma certa preocupação com soluções assistidas e interativas.

4.2 Segmentação baseada em detecção de movimento

Em técnicas de segmentação por movimento múltiplos quadros de um vídeo de entrada são analisados e procura-se estimar objetos que tenham apresentado alguma forma de movimento entre quadros. A detecção de movimento entre quadros pode ser feita de diversas formas, como por diferença de píxeis ou detecção de bolha (LINDEBERG; EKLUNDH, 1990) (blob detection). Um exemplo de segmentação realizada por diferença pode ser vista na figura 4.1, onde um simples cálculo de threshold é realizado entre a diferença de intensidade de dois quadros para realizar a detecção de bolha.

Métodos mais sofisticados geralmente empregam modelos estatísticos para estimar o comportamento dos objetos no vídeo, como os trabalhos dos autores Rasmussen (RASMUSSEN; HAGER, 2001), Shen (H. SHEN L. ZHANG; LI, 2007) e Stolkin (R. STOLKIN A. GREIG; GILBY, 2008). Ademais, outras abordagens trabalhadas na literatura incluem: segmentação por fluxo óptico (J. ZHANG F. SHI; LIU, 2007), wavelets (M. KONG J.-P. LEDUC; WICKERHAUSER, 1998) e fatorização (GOH; VIDAL, 2007) (C. JULIà A. SAPPA; LOPEZ, 2007).

A segmentação baseada em detecção de movimento, no entanto, possue diversas limitações. Primeiramente, devido à sua finalidade principal de identificar objetos que se moveram para fins de vigilância e segurança, não existe uma preocupação com a extração de forma, e suas segmentações são imprecisas. Em segundo lugar, câmeras de vigilâncias geralmente trabalham com resoluções baixas, de modo que poucas técnicas são preparadas para lidar com resoluções altas. Finalmente, como espera-se que operar em tempo real, limita-se a quantidade de processamento realizado nas imagens, de modo que o ruído torna-se um problema sério (ZAPPELLA; LLADó; SALVI, 2008). Devido à esses motivos, técnicas de segmentação baseada em detecção de movimento foram consideradas inadequadas para esse trabalho.

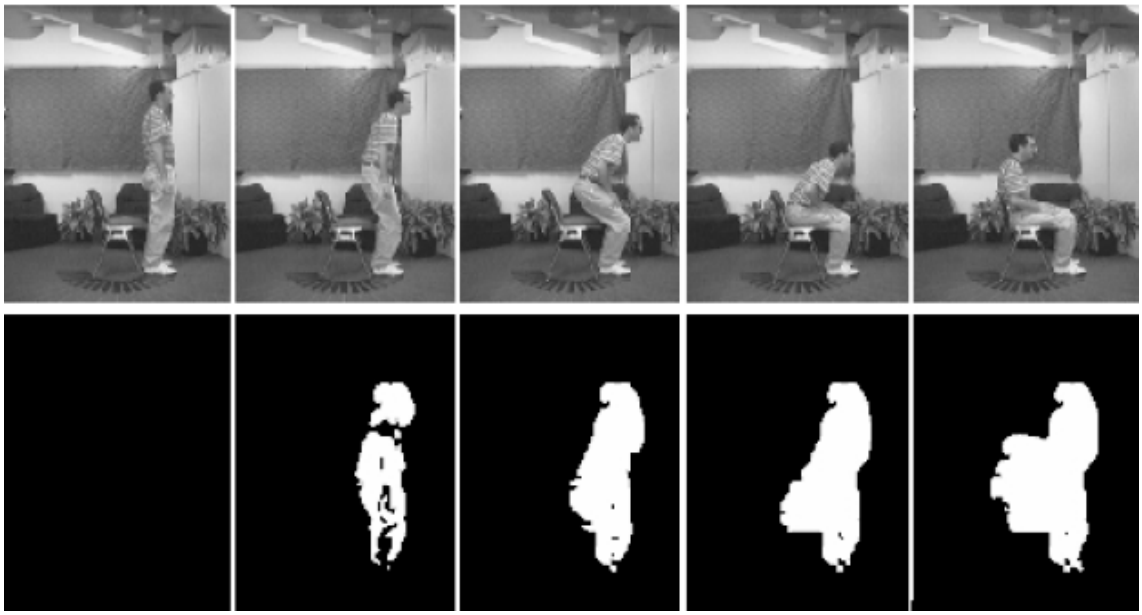


Figura 4.1: Exemplo de segmentação por movimento baseada em diferença. (BOBICK; DAVIS, 1996)

4.3 Segmentação de Frente/Fundo

O objetivo principal de técnicas de segmentação frente/fundo ou subtração de fundo é criar um modelo do fundo esperado com uma câmera estática para que este possa ser posteriormente removido de cenas. Não existe uma separação explícita entre segmentação por detecção de movimento e segmentação de frente/fundo, sendo que muitas possuem abordagens diferentes e são usadas com as mesmas finalidades. No entanto, o foco na modelagem do fundo é o que fundamentalmente as diferencia.

Formas simples de segmentação frente/fundo consideram que a variação de intensidades de um píxel pode ser modelada por apenas uma distribuição unimodal, como os trabalhos propostos por Wren (WREN CR AZARBAYEJANI A, 1997) e Horpraset (HORPRASERT, 1999). Esta forma de modelagem é bastante limitada e com apenas uma distribuição unimodal não é possível de se descrever fundos dinâmicos, como movimentos periódicos de plantas, veículos ou maquinário.

Modelos de misturas generalizadas de Gaussianas (comumente chamados de MOG), que são modelos estatísticos probabilísticos apresentam uma forma mais robusta para modelar fundos complexos. A utilização de MOGs sozinha pode ser usada para a extração de fundos dinâmicos, como demonstrado por Stauffer (STAUFFER C, 1999). Adicionalmente, o uso de MOGs acoplado à outras abordagens também é comum na literatura, como estatística Bayesiana (LEE DS HULL JJ, 2003), informações de cor e gradiente (JAVED O SHAFIQUE K, 2002)

(a) (b) (c)

Figura 4.2: Comparação de técnicas de segmentação frente/fundo baseadas em MOG e codebooks. (KIM et al., 2005)

e análise de mean-shift (PORIKLI F, 2003).

Técnicas baseadas em MOG apresentam algumas limitações recorrentes, no entanto. Primeiramente, fundos que possuem variações rápidas não são modelados facilmente através da utilização de algumas distribuições gaussianas, o que causa erros na subtração do fundo. Este efeito pode ser causado por fatores como microvibrações da câmera ou pela frequência de iluminação. Adicionalmente, são comuns erros de falso positivo em casos onde os objetos de frente se movimentam lentamente, devido a questões de treinamento. Ambos problemas são abordados nos trabalhos de Elgamall (ELGAMMAL A HARWOOD D, 2000) e Toyama (TOYAMA K KRUMM J, 1999), respectivamente, e em parte solucionados por Mittal (MITTAL A, 2004).

Kim (KIM et al., 2005) propõe uma modelagem de fundo baseada em píxel, onde cada ponto dos quadros é modelado temporalmente independente de seus vizinhos espaciais. Para tal, é utilizada uma estrutura de agregagem de informação chamada de codebook. Inicialmente é feito um processo de treinamento sobre alguns quadros, onde para cada ponto na resolução usada é feito uma análise de sua variação de cor ao longo do tempo, criando unidades de informação chamadas codewords que representam pequenos sub-espaços de cor. Para subtrair o fundo, então, pontos de entrada são comparados com os codebooks existentes em cada codeword e caso se encaixem no sub-espaço descrito por estes, são considerados pertencentes ao fundo. Este método é bastante robusto, modelando movimentos periódicos como variações regulares em píxeis.

A figura 4.2 mostra a comparação da segmentação frente/fundo baseada em MOG descrito por Stauffer (STAUFFER C, 1999) com o método de codebooks proposto por Kim (KIM et al., 2005). Na figura 4.2a pode ser visto o quadro original, sendo que a pessoa representa a frente e o ambiente contendo a parede de tijolos e a grama representa o fundo. As figuras 4.2b e

4.2c representam os resultados da segmentação usando MOG e codebooks, respectivamente. É possível observar na figura 4.2b os resultados falso-positivos já comentados na identificação de partes do suéter vermelho da pessoa como pertencentes à parede de tijolos.

Uma arquitetura para a segmentação de vídeos baseada em subtração de fundos é proposta por Ong em "Fast Automatic Video Object Segmentation for Content-Based Applications" (ONG E. P., 2006). Sua solução é composta de várias técnicas diferentes, como determinação de threshold adaptativa, geração robusta de referência para quadros estacionários, seleção automática de frames e refinamento espacial baseado em contornos. Neste trabalho, o autor demonstra como a combinação de diversas abordagens pode ser usada para aumentar a qualidade da segmentação.

De uma forma geral, a solução apresentada por Ong apresenta os melhores resultados dentre os trabalhos de segmentação frente/fundo estudados. Isto é atribuído em grande parte à sua arquitetura, que foi estruturada de forma ad hoc para resolver uma classe de problemas de segmentação. Esta arquitetura, no entanto, é demasiadamente complexa e pouco compatível com o processo de extração de trimaps.

Durante a pesquisa foram realizados testes com implementações de técnicas de MOG e codebooks, de modo que as segundas se apresentaram muito mais aplicáveis à proposta desse trabalho. A característica de técncias baseadas em codebooks de tratar cada ponto da imagem separadamente as torna suscetíveis a ruídos não periódicos, mas também permite o uso de resoluções altas com um aumento linear de complexidade computacional, bem como aumenta a capacidade de precisão dos resultados. Ademais, a simplicidade de implementação confere uma adaptação fácil para classes diferentes de problemas, como no caso desse trabalho a extração de trimaps. Nas sessões 5.1 e 5.2 é descrita a solução baseada em codebooks desenvolvida e como esta difere da abordagem original.

5 PROPOSIÇÃO DO TRABALHO

Para solucionar os problemas descritos na sessão 1.1, esse trabalho propõe uma arquitetura para a segmentação de vídeos dividida em duas etapas: extração de trimaps e aplicação de alpha matting. A extração de trimaps utiliza treinamento de fundo e minimização de energia, sendo que o processo geral pode ser visto na figura 5.1.

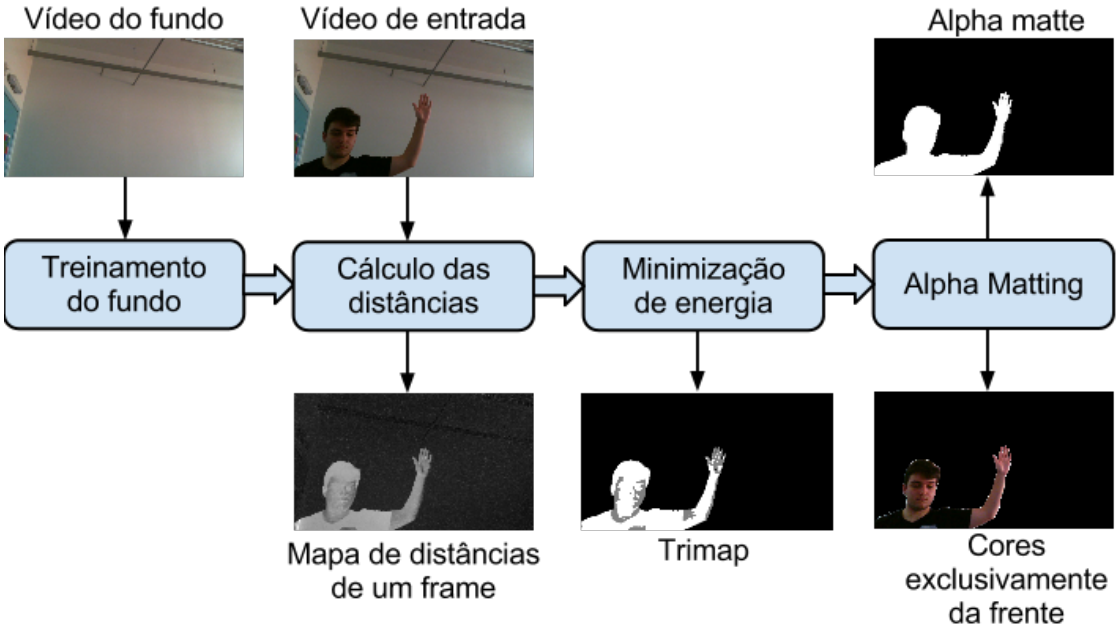


Figura 5.1: Representação diagramática do processo geral de segmentação.

O ponto central do processo de extração de trimaps é a aplicação da minimização de energia, de modo que as etapas de treinamento e cálculo de distâncias foram projetadas para a suportar. Esta abordagem foi escolhida pois técnicas de minimização de energia podem ser facilmente modeladas para solucionar problemas de segmentação ternária através de rotulagem. No caso

da extração de trimaps, a rotulagem de píxeis deve ser criteriosa de modo a poder determinar quando a cor de um píxel é suficientemente diferente para ser considerada frente, quando é suficientemente igual ao fundo e quando pode ser uma combinação de ambos.

A decisão da classificação dos píxeis é feita baseada em custos de energia, levando em consideração a semelhança de cor entre píxeis e suas regiões vizinhas. A entrada principal para este processo é um mapa de distâncias de cores, que descreve em escala de cinza a diferença entre as cores dos píxeis esperados do fundo e um quadro de entrada do vídeo. Este mapa de distâncias é calculado em relação à um modelo do fundo obtido através de uma fase de treinamento.

A ordem de etapas do processo pode ser então descrita como:

1. Treinamento do fundo: um vídeo é utilizado para criar um modelo de cores do fundo;
2. Cálculo de distâncias: um quadro do vídeo de entrada é comparado com o modelo conhecido do fundo para determinar as distâncias de cores;
3. Minimização de energia: resolve-se um problema de rotulagem por minimização de energia para extrair o trimap; e finalmente,
4. Aplicação de alpha matting: uma técnica de alpha matting é aplicada tendo como entrada o trimap gerado e o quadro atual. A saída é o mapa de transparências e a cor do objeto extraído.

As seções a seguir descrevem em detalhes os processos envolvidos em cada uma das etapas e suas dificuldades.

5.1 Treinamento do Fundo

Nesta etapa um vídeo de poucos segundos (geralmente três a cinco) representando o espaço de fundo da filmagem é usado para criar um modelo espacial de cores, utilizando uma abordagem baseada em codebooks. Como foi visto na seção 4.3, um codebook é uma estrutura de dados que armazena informação na forma de codewords. Cada codeword agrega características de uma classe de amostras de píxeis observados na fase de treinamento em uma determinada posição.

Seja $\Omega_{x,y} = \{c_1, c_2, ..., c_n\}$ o codebook que modela o píxel da posição (x, y) da imagem. Este codebook vai conter tantos codewords c quanto forem necessários para representar as variações que ocorrem em (x, y) ao longo do vídeo de treinamento. Uma codeword contém um

vetor de cor v e uma n-tupla de valores, dependendo do espaço de cor utilizado. Para esse trabalho, foram desenvolvidas duas implementações do algoritmo baseadas em dois espaços de cores diferentes: RGB e CIELAB. Questões relacionadas à estes modelos de cor são tratadas na subseção 5.1.1.

As equações 5.1 e 5.2 representam a configuração das tuplas em um codeword RGB e CIELAB, respectivamente. Na forma RGB, B^{max} e B^{min} são os valores de brilho mínimo e máximo associados à um codeword, enquanto na forma CIELAB estes não são necessários. Juntamente com o vetor de cor, estes valores modelam um sub-espaço de cor que busca expressar as variações encontradas nas amostras de entrada.

$$c_{RGB}(i) = \langle B_i^{max}, B_i^{min}, f, \lambda_i, a_i^{prim}, a_i^{fin} \rangle \tag{5.1}$$

$$c_{CIELAB}(i) = \langle f, \lambda_i, a_i^{prim}, a_i^{fin} \rangle \tag{5.2}$$

Os valores f, λ, a_i^{prim} e a_i^{fin} são utilizados apenas durante o treinamento e servem para modelar a periodicidade de ocorrência de codewords. A frequência de ocorrência de uma codeword é armazenada em f, sendo um valor incremental. λ representa o tamanho do maior intervalo na qual este codeword não ocorreu durante o treinamento, enquanto a_i^{prim} e a_i^{fin} representam o primeiro e último quadro na qual ele ocorreu, respectivamente. Com estas informações, é possível determinar se um determinado codeword pertence à píxeis de um objeto estático, de um objeto realizando movimentos periódicos ou de um objeto passageiro, que não deve ser incluído no modelo do fundo.

Durante o processo de treinamento cada ponto da imagem é avaliado separadamente ao longo do tempo. Seja $\chi_{x,y} = \{p_1, p_2, ..., p_t\}$ o conjunto de píxeis dos quadros do vídeo de treinamento em uma determinada posição (x, y) ao longo do tempo t. Para cada elemento em χ aplica-se o seguinte algoritmo:

- Se existe em $\Omega_{x,y}$ um codeword c_i que pode conter p_t em seu subespaço de cor:
  - Incrementa-se o valor de f em um;
  - É realizada uma média entre o vetor de cor v_i e p_t, da forma $v_i = \frac{fv_i + p_t}{f+1}$;
  - Os limites do subespaço definidos por B_i^{max} e B_i^{min} ou R_i são ajustados para melhor corresponder à p_t, caso necessário;

  - Os valores de λ_i e a_i^{fin} são atualizados, caso necessário.

- Caso contrário:

  - É criada uma nova codeword c_{n+1}
  - É definido $f = 1$;
  - É definido v_{n+1} como p_t;
  - Os limites do subespaço definidos por B_i^{max} e B_i^{min} ou R_i são inicializados com parâmetros estáticos;
  - Os valores de a_{n+1}^{ini}, λ_{n+1} e a_{n+1}^{fin} são inicializados como t.

Desta forma, em $t = 0$ têm-se $\Omega_{x,y} = \{\}$, de modo que cria-se a primeira codeword c_0 com $v_0 = p_0$ e a n-tupla como $\langle \epsilon^{max}, \epsilon^{min}, 1, 0, 0, 0 \rangle$ ou $\langle 1, 0, 0, 0 \rangle$, onde ϵ^{max} e ϵ^{min} e ρ representam valores estáticos iniciais para o brilho máximo e mínimo, respectivamente. Para cada píxel p_t subsequente que pertencer à uma codeword c_i, esta vai sofrendo uma expansão do seu subespaço para melhor agregas as amostras encontradas.

Ao final do processo, os valores de f, a^{ini}, λ e a^{fin} são analisados para indicar quais codewords devem ser removidas do modelo do fundo. Valores pequenos de f em relação à t indicam que este codeword ocorreu de forma escassa e deve ser eliminado. Um valor de a^{ini} próximo à t representa uma codeword que só foi registrada ao final do processo de treinamento, e provavelmente não deve ser considerada como pertencente ao fundo. λ indica o tempo máximo consecutivo na qual a codeword não ocorreu e, caso seja maior que $t/2$ esta deve ser invalidada. Eventos que podem justificar a presença de codewords inválidos incluem a passagem de pessoas pela cena filmada, a queda de um objeto ou uma trepidação da câmera.

A ordem dos píxeis em $\chi_{x,y}$ pode afetar o resultado final, pois uma codeword pode sofrer diversas expansões e ir lentamente incorporando píxeis com diferenças incrementais de cor, que caso estivessem em ordem diferente seriam considerados para exclusão. Isto não pode ser considerado um problema, pois sempre são utilizados vídeos com quadros ordenados temporalmente. Ademais, este efeito é adequado pois espera-se que mudanças de cor em um determinado píxel sejam temporalmente suaves e representem fenômenos que estejam ocorrendo na cena.

O modelo final do fundo é uma estrutura que contém um codebook $\Omega_{x,y}$ para cada píxel da imagem (307200 em uma imagem de 640x480 de resolução) e cada codebook contém tantos codewords quanto forem necessários para modelar as variações de cor naquela posição. Este

modelo é muito mais robusto em representar o fundo do que uma simples imagem, pois é capaz de prever variações de iluminação, ruído e até certos movimentos periódicos que poderiam acontecer no fundo, como a movimentação de folhas em uma árvore.

Na subseção a seguir serão discutidas as particularidades das implementações dos modelos de cores utilizados.

5.1.1 Espaços de Cor

Espaços de cor são representações matemáticas que modelam a disposição de cores de acordo com vários critérios. Um dos exemplos mais comuns são espaços baseados no modelo RGB (vermelho, verde, azul), como sRGB e Adobe RGB, usados principalmente em dispositivos de vídeo e fotografia. Estes espaços de cor se baseiam na combinação de intensidade dos espectros de luz vermelha, verde e azul, criando um espaço tridimensional de três eixos onde cada cor corresponde à um ponto diferente.

Espaços de cor diferente, mesmo utilizando o mesmo modelo de cor, apresentam limitações e usos diferentes. A propriedade que mais define um espaço de cor é o seu gamut, ou o espaço total de cores mapeadas do modelo para espaço de cor. Como pode ser visto na figura 5.2, a área exterior na forma de uma ferradura representa o limite das cores visíveis para uma determinada luminosidade, e as impressões sobre ela são os gamuts dos espaços de cor RGB e CMYK. Ademais, geralmente os gamuts tem uma forma triangular ou aproximada, relativa a os três espectros R,G e B. No entanto, os triângulos não são equiláteros, de forma que cada um dos componentes tem uma importância diferente no processo de mapeamento.

Uma das maiores preocupações ao se definir um espaço de cor é a sua relação com a percepção humana de cores, pois a capacidade de perceber cada espectro de cor não é uniforme e está relacionada à formação biológica do olho. CIE XYZ (SMITH; GUILD, 1931) foi o primeiro espaço de cores projetado para representar a gama de cores visíveis pelo olho humano. Diferente de um modelo RGB, os componentes do espaço XYZ representam fatores de percepção dos cones e bastonetes do olho. Os espaços de cor LAB (SMITH; GUILD, 1931) e CIELAB, proposto pela Commision Internationale L'Eclairage, são baseados no espaço CIEXYZ e utilizam um componente para a luminosidade (L ou L*) e dois para as informações de croma (a e b ou a* e b*).

Nesse trabalho foram desenvolvidas duas implementações diferentes para o processo de treinamento de fundo, uma utilizando o espaço de cor RGB como na implementação original

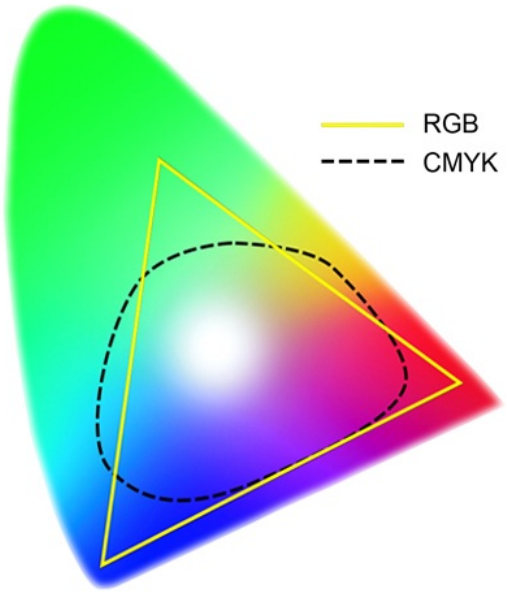


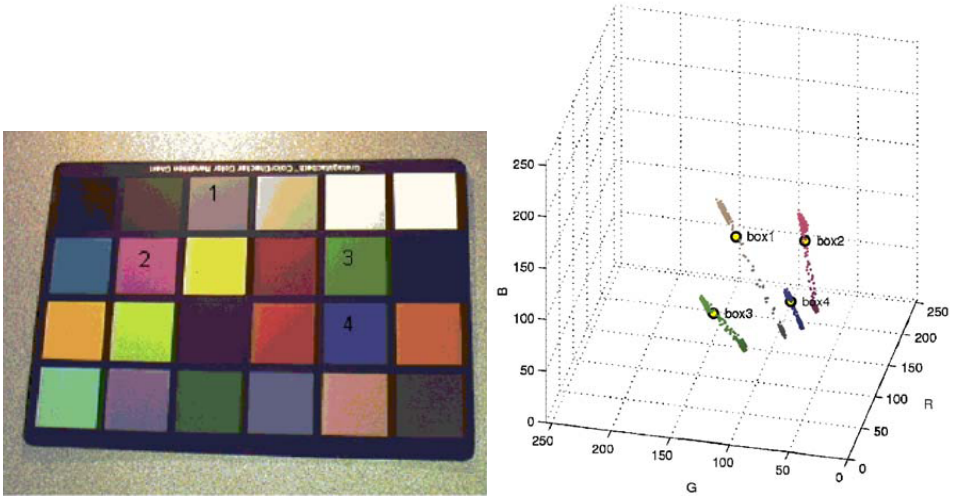
Figura 5.2: Diferença entre gamuts de espaços de cores CMYK e RGB em relação ao espectro visível.

de codebooks, e a outra o espaço CIELAB para tentar contornar algumas de suas limitações. Ambas as implementações são baseadas na abordagem proposta por Kim (KIM et al., 2005), que em seu trabalho usa apenas o espaço RGB.

5.1.1.1 Espaço RGB

O espaço RGB é um espaço euclidiano onde existe uma distância de 90^o entre os eixos de cor vermelho, verde e azul e os pontos $(0, 0, 0)$ e $(255, 255, 255)$ representam preto e branco, respectivamente. Neste espaço, uma cor com a sua variação de brilho pode ser representada pela reta paramétrica formada por um vetor direção (r, g, b) e um parâmetro escalar de brilho β. Desta forma, duas cores diferentes $\beta_1(r_1, g_1, b_1)$ e $\beta_2(r_2, g_2, b_2)$ vão apresentar uma distância euclidiana maior entre seus pontos quanto mais claras forem, ao mesmo tempo que para valores mínimos de β sua diferença será muito pequena.

Na figura 5.3 pode ser visto como uma distribuição de pontos de cores em um espaço RGB se comporta com variações de iluminação. Uma planilha de cores contendo quadrados com cores diferentes numeradas de 1 a 4 é filmado sobre fontes de luz diferentes (figura 5.3a) e as informações de cores de seus píxeis são armazenadas. Na figura 5.3b estas informações de cor são convertidas em pontos em um espaço tridimensional RGB, de modo que quatro agrupamentos de pontos distintos que podem ser relacionados às cores de 1 a 4 são apresentados. Como observado anteriormente, cada cor descreve uma espécie de linha a partir da origem que



(a) (b)

Figura 5.3: Exemplo de amostras de cor diferentes sob uma perspectiva espacial. a) Paleta de cores observada. b) distribuição de pontos de cor em um espaço RGB pertencentes às classes de cor da paleta. (KIM et al., 2005)

se distancia das demais quanto maior o brilho.

Esta distorção de distâncias é um problema bastante grande para uma modelagem de cor em vídeo, pois torna bastante difícil distinguir entre cores diferentes ou variações de iluminação. Para contornar este problema, no trabalho de Kim codewords procuram modelar um sub-espaço de cor cilíndrico para melhor representar distribuições de cores como as vistas na figura 5.3b, diferenciando entre variações de brilho (altura do cilindro) e diferenças de cor (raio do cilindro).

O espaço cilíndrico de um codeword é centrado em seu ponto de cor v, seu raio é definido de acordo com um critério de tolerância de variação de cor e sua base e topo correspondem à B_i^{max} e B_i^{min}, respectivamente. Nesta forma de representação, um píxel p_i pode ser considerado como pertencente à um codeword c_n se estiver contido dentro de seu cilindro.

Para um píxel $p = \{R_p, G_p, B_p\}$ estar contido no cilindro de um codeword c_n de centro $v_n = \{R_n, G_n, B_n\}$ e tupla $\langle B_n^{max}, B_n^{min}, f_n, \lambda_n, a_n^{prim}, a_n^{fin} \rangle$, duas condições precisam ser verdadeiras:

1. O valor de distorção de cor $\delta(p, v_n)$ precisa ser menor que um valor Δ estipulado; e

2. O valor de brilho B_p estar entre αB_n^{min} e βB_n^{max}.

O brilho do píxel é definido como $B_p = \sqrt{R_p^2 + G_p^2 + B_p^2}$ e a distorção de cor δ pode ser calculada por

$$\kappa^2 = \|p\|^2 cos^2\theta = \frac{\langle p, v_n \rangle^2}{\|v_n\|^2},$$
$$\delta_{RGB}(p, v_n) = \sqrt{\|p\|^2 - \kappa^2} \tag{5.3}$$

onde α e β são parâmetros que permitem a regulagem de altura dos cilindros e Δ_{RGB} determina seu raio. Esta fórmula representa uma distância ponderada entre dois pontos de cor que leva em consideração o brilho relativo do píxel avaliado. Os valores de αB_n^{min} e βB_n^{max}, definidos na tupla do codeword c_n armazenam, respectivamente, o menor e o maior valor de B_p encontrado nos píxeis que foram considerados pertencentes à esse codeword.

As vantagens principais de se utilizar esta implementação de codebooks sobre um espaço RGB são a simplicidade dos cálculos e eficiência computacional. Para a aplicação desse trabalho, onde após o treinamento dos codebooks se requer uma medida de distância e não uma classificação binária de espaço, a modelagem cilíndrica de espaço é pouco adequada, pois cilindros não possuem simetria tridimensional. Por este motivo, além de adaptar a implementação original de codebooks em RGB para esse trabalho, foi desenvolvida uma implementação diferente sobre o espaço CIELAB que será descrita na seção a seguir.

5.1.1.2 Espaço CIELAB

Espaços de cor LAB são espaços de cor oponentes que utilizam um modelo de três componentes: L para luminância e a e b para crominância, baseado em coordenadas do espaço CIEXYZ (SMITH; GUILD, 1931) não linearmente comprimidas.

Diferente do espaço RGB que é um espaço euclidiano com bases ortonormais representando os componentes de vermelho, verde e azul, o espaço CIELAB modela o espaço esfericamente utilizando crominância e luminância, como demonstra a figura 5.4. Na imagem podem ser vistos dois componentes laterais de crominância com cores que se opões: o componente a com verde/vermelho e o componente b com azul/amarelo. O componente l, ou luminância é representado pela altura da esfera, de modo que valores altos (tons claros) e baixos (tons escuros) possuem uma circunferência menor, ou seja, menos cores.

O objetivo principal de se utilizar o espaço de cor CIELAB é para melhor representar diferenças perceptíveis de cor. Isso significa não só uma modelagem mais adequada que compense as distorções de iluminação do espaço RGB, como visto anteriormente, mas também que leve em consideração a percepção do olho humano. Na figura 5.5 pode ser visto um diagrama de

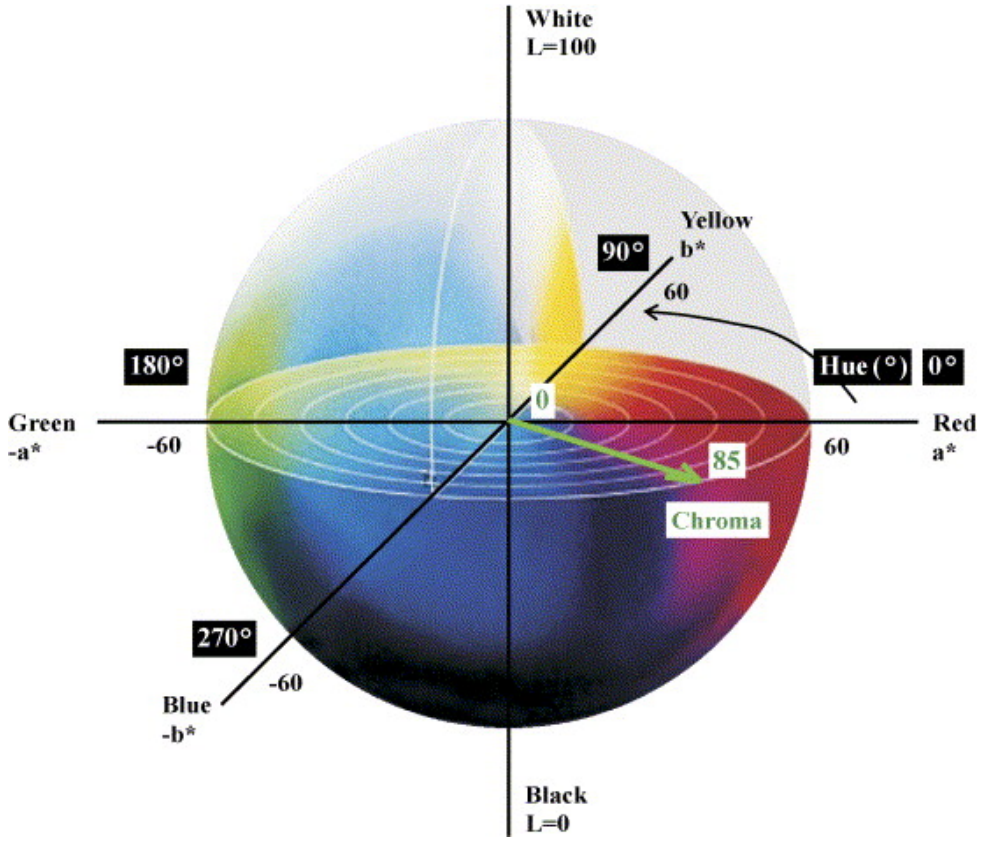


Figura 5.4: Representação gráfica do espaço CIELAB. (SALM et al., 2004)

MacAdam (MACADAM, 1942), onde cada elipse representa o que pode ser considerado uma cor perceptualmente semelhante sobre um diagrama de cromaticidade XY de um espaço XYZ. Isto demonstra que para calcular a distância entre duas cores que leve em consideração aspectos perceptuais de iluminação e cromaticidade é necessária uma modelagem matemática adequada.

A métrica de distância oficial definida pela CIE (Comissão Internacional de Iluminação) é chamada de ΔE^*_{ab}, aplicada sobre o espaço CIELAB e apresentando versões diferentes ao longo dos anos. Esta métrica utiliza um modelo de cor L*c*h sobre oe spaço CIELAB: luminância, croma e matiz. De uma forma simplificada, a luminância seria a altura de uma cor na esfera do espaço de cor, como visto anteriormente, o croma sua distância do centro e o matiz seu ângulo.

Uma das versões mais atuais e a utilizada neste trabalho é chamada ΔE^*_{00} ou CIEDE2000 (SHARMA; WU; DALAL, 2005), apresentando diversas correções em sua modelagem, incluindo a possibilidade de ajustar pesos para cada componente. Considerando dois píxeis $P_1 = \{L_1, a_1, b_1\}$ e $P_2 = \{L_2, a_2, b_2\}$, sua fórmula geral pode ser descrita pelo seguinte conjunto de equações:

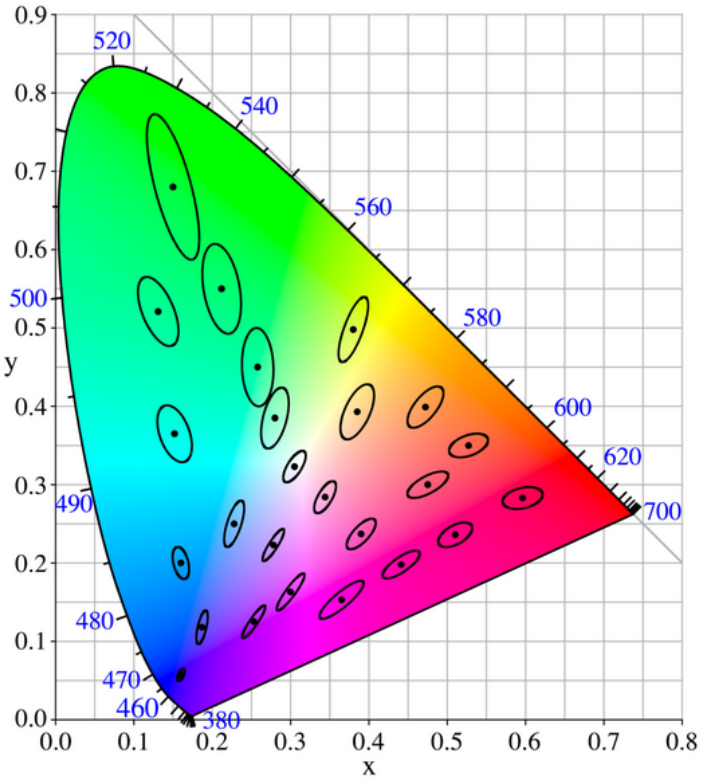


Figura 5.5: Diagrama de MacAdam demonstrando cores perceptualmente semelhantes em um espaço CIEXYZ. (WYSZECKI; STILES, 2000)

$$\Delta E^*_{00}(P_1,P_2)=\sqrt{\left(\frac{\Delta L'}{k_L S_L}\right)^2+\left(\frac{\Delta C'}{k_C S_C}\right)^2+\left(\frac{\Delta H'}{k_H S_H}\right)^2+R_T\frac{\Delta C'}{S_C}\frac{\Delta H'}{S_H}},$$

$$\Delta L'=L_2-L_1,\qquad \Delta C'=C_2-C_1,\qquad \Delta H'=2\sqrt{C'_1C'_2}sen(\Delta h'/2),$$

$$C_1=\sqrt{a_1^2+b_1^2},\qquad C_2=\sqrt{a_2^2+b_2^2},\qquad \bar{C}=\frac{C_1+C_2}{2}, \tag{5.4}$$

$$\text{onde}\quad C'_1=\sqrt{a_1'^2+b_1^2},\qquad C'_2=\sqrt{a_2'^2+b_2^2}$$

$$\text{e}\quad a'_1=a_1+\frac{a_1}{2}\left(1-\sqrt{\frac{\bar{C}^7}{\bar{C}^7+25^7}}\right),\quad a'_2=a_2+\frac{a_2}{2}\left(1-\sqrt{\frac{\bar{C}^7}{\bar{C}^7+25^7}}\right),$$

enquanto a diferença de matiz, que é uma diferença angular, apresenta a forma:

$$\Delta h' = \begin{cases} h'_2 - h'_1 & |h'_2 - h'_1| \leq 180^o \\ h'_2 - h'_1 + 360^o & |h'_2 - h'_1| > 180^o, h'_2 \leq h'_1 \\ h'_2 - h'_1 - 360^o & |h'_2 - h'_1| > 180^o, h'_2 > h'_1 \end{cases} \tag{5.5}$$

$$\text{com} \quad h'_1 = tan^{-1}(b_1/a'_1) \quad \text{e} \quad h'_2 = tan^{-1}(b_2/a'_2).$$

Nesta equação k_L, k_C e k_H são parâmetros que ajustam a influência de cada componente na distância final. Os coeficientes S_L, S_C e S_H juntamente com o termo $R_T \frac{\Delta C'}{S_C} \frac{\Delta H'}{S_H}$ representam ajustes e compensações para peculiaridades sensoriais e matemáticas desse modelo de cor. Sua descrição e cálculo são descritos em detalhe no trabalho "The CIEDE2000 Color-Difference Formula: Implementation Notes, Supplementary Test Data, and Mathematical Observations" (SHARMA; WU; DALAL, 2005).

A implementação de codebooks desenvolvida no espaço de cor CIELAB utiliza a distância CIEDE2000 como base fundamental. Ao invés de definir parâmetros que modelam um subespaço de cores cilíndrico para as amostras, o valor de cor v_n de um codebook c_n representa seu ponto de cor central e todas as cores são comparadas através da sua distância ΔE em relação a esse centro. Em um espaço euclidiano esta modelagem poderia ser considerada uma esfera centrada em v_n, mas como demonstra a figura 5.5 distâncias perceptuais de cor apresentam formas elipsóides não-regulares.

Desta forma, para o treinamento no espaço CIELAB define-se uma distância máxima de similaridade de cor Δ_{CIELAB}, análoga a Δ_{RGB} utilizada anteriormente. Um píxel de entrada p pode ser considerado contido em um codeword c_n se $E^*_{00}(p, v_n) < \Delta_{CIELAB}$. Revendo a figura 5.5, pode ser feita uma comparação onde cada uma das elipses representa um codeword com ponto central v indicado e as cores dentro da elipse como sendo as que apresentam uma distância inferior ao Δ_{CIELAB} estipulado. As peculiaridades de percepção de cor fazem com que as elipses tenham áreas e formatos diferentes, mas a utilização de uma métrica adequada garante que todas mapeiam um espaço perceptual de tamanho semelhante.

Quando um píxel p é considerado pertencente à um codebook c_n durante o processo de treinamente, aplicam-se praticamente as mesmas etapas descritas na seção 5.1.1, com excessão das considerações de brilho. É realizada a média ponderada entre p e v_n para ajustar o ponto central e atualizam-se os valores de f, λ, a^{ini} e a^{fin}. Caso não exista codebook para conter o píxel de entrada, cria-se um novo codebook com $v_{n+1} = p$ e tupla $\langle 1, 0, t, t \rangle$.

O motivo principal da utilização de um espaço de cor CIELAB com distâncias de cor CIEDE2000 é fazer uma modelagem de fundo com foco em representar diferenças de cor. Através

dos parâmetros k_L, k_C e k_H é possível ajustar a influência da luminância sobre o cálculo de distâncias, sendo mais fácil de compensar problemas de diferença de iluminação, como descritos nas seções 2.1 e 2.1.1. Ademais, como está sendo utilizada uma métrica de distâncias baseada na percepção do olho humano, espera-se reproduzir em alguns aspectos uma segmentação que um usuário faria. Os resultados obtidos e sua comparação com a implementação no espaço de cor RGB são apresentados no capítulo 7.

Existem duas limitações principais nesta abordagem, no entanto: o cálculo da distância E^*_{00} é composto de diversas operações e consome um tempo computacional considerável ao ser executado para vários píxeis, codebooks e quadros consecutivos; e ao utilizar um espaço de cores não aditivo, diferente do padrão para alpha matting (RGB) existe a possibilidade de que os trimaps gerados não apresentem uma sinergia ideal com o processo de alpha matting.

A primeira limitação não é preocupante pois a aplicação é executada em off-line, resultando apenas em um processo de testes mais lento e trabalhoso do que desejado. Caso necessário, o treinamento e o cálculo de distâncias podem ser facilmente paralelizados e portados para uma arquitetura de GPU em trabalhos futuros, potencialmente aumentando seu desempenho. A manifestação da segunda limitação não foi perceptível durante o desenvolvimento do trabalho, especialmente ao ser considerada a grande quantidade de variáveis envolvidas no processo. Ademais, este efeito, apesar de plausível, não é tratado na literatura e uma pesquisa mais detalhada seria necessária para descrever sua ocorrência.

5.2 Cálculo de Distâncias de Cor

Após a realização do processo de treinamento descrito na seção anterior, é obtido um conjunto de codebooks que modelam a distribuição de cores que ocorrem no fundo de forma dinâmica. Estes codebooks podem ser utilizados em conjunção a diversos vídeos diferentes com o mesmo fundo, desde que estes mantenham as mesmas configurações de câmera (posição, orientação, foco), espaciais (disposição do local e de objetos) e de iluminação do vídeo de treinamento.

Na aplicação de codebooks conceitualizada por Kim (KIM et al., 2005), o processo de extração de fundo de um vídeo de entrada é uma segmentação binária onde um píxel de entrada $p_{x,y}$ é comparado com cada codeword c_n presente no codebook $\Omega_{x,y}$ de sua posição. Para cada c_n os mesmos critérios de contenção descritos na seção 5.1.1.1 são avaliados:

1. O valor de distorção de cor $\delta(p, v_n)$ precisa ser menor que um valor Δ estipulado; e

2. O valor de brilho B_p estar entre αB_n^{min} e βB_n^{max}.

Caso as condições 1 e 2 sejam verdadeiras para qualquer codeword c_n o píxel de entrada é considerado como contido em seu subespaço e, portanto, pertencente ao fundo conhecido, sendo pintado de preto na segmentação resultante. Se nenhum dos codewords c_n em $\Omega_{x,y}$ puder conter o píxel p em seu subespaço, ele apresenta uma cor não modelada no treinamento do fundo e deve ser destacado na segmentação, sendo pintado de branco.

Esta segmentação resultante não é criteriosa o suficente para gerar um trimap, de modo que a comparação do fundo treinado com a entrada foi adaptada. O objetivo desta estapa nesse trabalho é a obtenção de um mapa de distâncias de cor entre um quadro do vídeo de entrada e o modelo de cores conhecidas do fundo. Para esta finalidade, também é realizada uma comparação de um píxel de entrada p_{x_y} com os codewords em $\Omega_{x,y}$, mas o resultado é um valor no intervalo $(0, 1)$ ou $[0, 255]$ em escala de cinza indicando a distância entre o píxel p_{x_y} e o codebook c_n mais próximo encontrado.

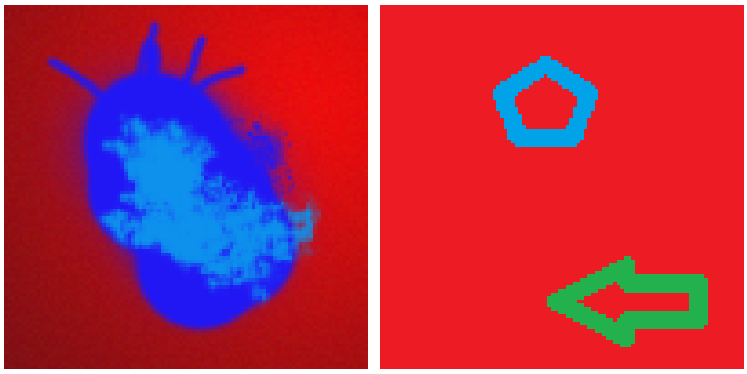
As distâncias calculadas são normalizadas entre si para corresponderem adequadamente a uma distribuição monocromática de cores. Desde modo, tons mais claros de cinza indicam distâncias mais altas de cor, mas apenas relativamente às outras cores da imagem e não absolutamente.

No caso da implementação em RGB, inicialmente é verificado se o píxel p_{x_y} está contido em algum dos cilindros das codewords em $\Omega_{x,y}$. Caso este esteja contido em mais de um cilindro, é calculada a menor distância euclidiana entre p_{x_y} e o segmento de reta central de cada cilindro, e este é o resultado final. Caso contrário, é utilizada a distância euclidiana entre o ponto p_{x_y} e a superfície do cilindro mais próximo. Esta abordagem não é ideal e não considera adequadamente a distorção de cor descrita na subseção 5.1.1.1, o que é consequência das formulações utilizadas para a aplicação original.

Na subseção a seguir dois conjuntos de imagens serão usados para demonstrar o processo de cálculo de distâncias em ambos os espaços de cores trabalhados.

5.2.1 Exemplos de mapas de distância

O objetivo deste exemplo é demonstrar como diferentes entradas e configurações afetam o modelo do fundo criado e, em consequência, o mapa de distâncias final. A imagem 5.6a apresenta um objeto de duas cores azuis diferentes sobrepondo-se a um fundo de cor vermelha dominante contendo um foco de luz e uma certa quantidade de ruído. Esta imagem representa



(a) (b)

Figura 5.6: Exemplos de figuras contendo uma composição de objetos com e sem transparência sobre fundos vermelhos.

(a) (b) (c)

Figura 5.7: Exemplos de fundos vermelhos diferentes.

em pequena escala diversos elementos que estarão presentes nas imagens reais: iluminação, ruído, transparência e fundo com uma cor não-constante. A imagem 5.6b é uma simplificação para fins demonstrativos representando dois objetos de cores diferentes, sem transparência em um fundo vermelho constante.

O fundo 5.7a corresponde exatamente ao fundo da imagem 5.6a e configura um caso ideal de comparação entre diferenças de cor de frente e fundo. A imagem de fundo 5.7b apresenta um tom constante de vermelho, levemente diferente do visto na imagem 5.6b, mas não o suficiente para apresentar distâncias perceptíveis de cor. Finalmente, o fundo visto na imagem 5.7c é uma composição análoga a da imagem 5.7a com um foco de luz diferente.

As tabelas 5.1 e 5.2 apresentam os diversos mapas de distância gerados nos espaços RGB e CIELAB, respectivamente. As colunas representam qual imagem de fundo foi usada: 5.6a ou 5.6b, enquanto as linhas representam quais imagens foram usadas no treinamento do fundo. Para o treinamento do fundo, em cada caso criou-se um codebook vazio e considerou-se como frames de entrada somente as imagens listadas, na ordem em que aparecem.

O que pode ser visto nas imagens das tabelas 5.1 e 5.2 é, principalmente, a qualidade em representação de distâncias nos exemplos que utilizam o espaço de cores LAB e a facilidade de remoção de fundo da abordagem RGB. Como a finalidade original do algoritmo de codebooks em RGB era uma segmentação binária, mesmo com adaptações este aspecto ainda é bastante presente nos resultados. Ao invés de expressar nuâncias nas semelhanças de cor, as distâncias apresentam um constraste muito maior.

Outro ponto interessante a ser notado é como a combinação de imagens de fundo pode, em ocasiões, prejudicar o modelo do fundo. Isso é causado principalmente pelo processo de ajuste de um codebook existente para acomodar um píxel novo durante a fase de treinamento, e pode ser observado melhor quando os fundos 5.7a e 5.7b são combinados na tabela 5.2. A mancha cinza observada ocorre quando os codebooks daquela região, que após o treinamento do primeiro frame representam as cores da imagem 5.7a, começam a considerar os píxeis da cor da imagem 5.7b semelhantes o suficiente os incorporarem. Desta forma, é executado o processo de inclusão de um píxel a um codebook descrito na seção 5.1 e o codebook original é modificado, distanciando-se da cor do fundo original.

Por consequência da natureza sequencial e combinativa do processo de treinamento, efeitos como este podem ocorrer em um conjunto de frames dependendo da ordem de treinamento. Na figura 5.8 pode ser vista a comparação de dois mapas de distâncias da mesma imagem

Tabela 5.1: Mapas de distâncias gerados para as imagens da figura 5.6 sob configurações diferentes das imagens da figura 5.7, no espaço de cores RGB.

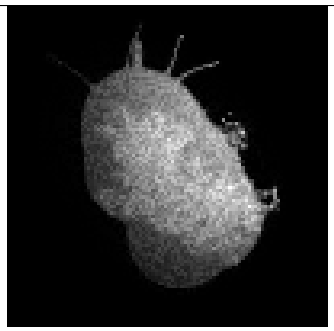
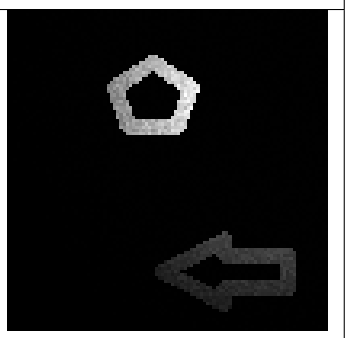
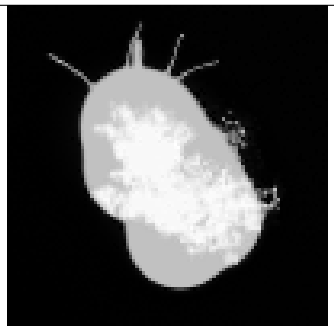
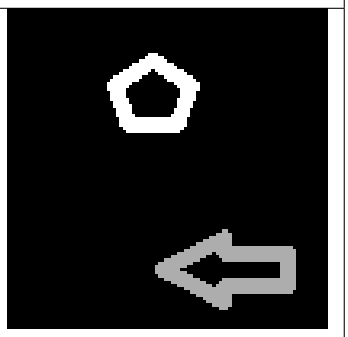
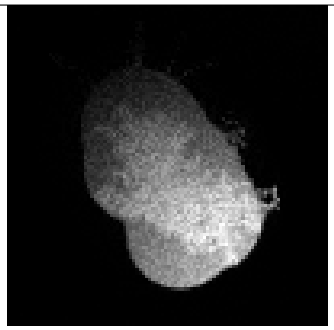
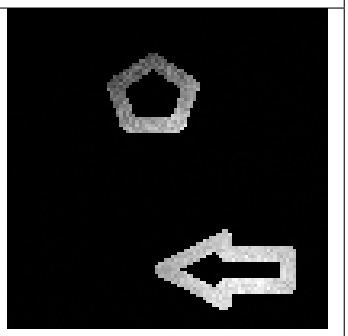
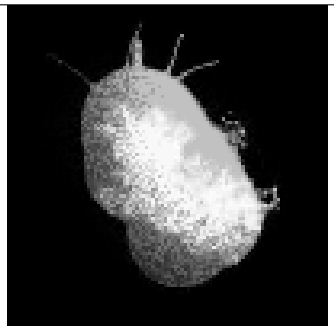
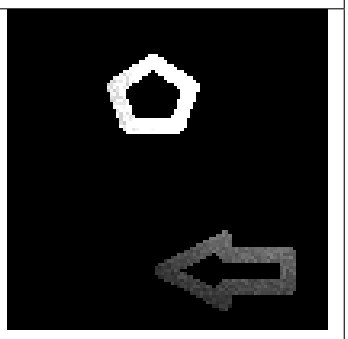
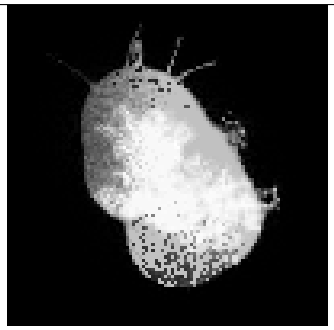
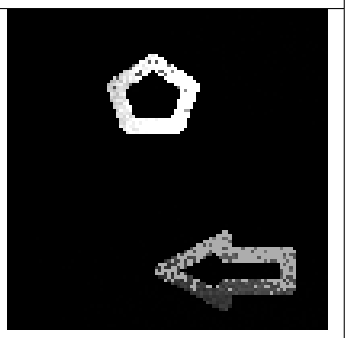
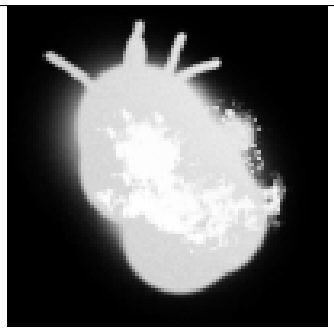
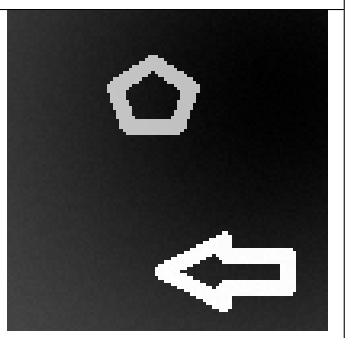
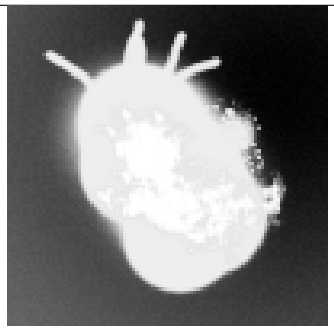
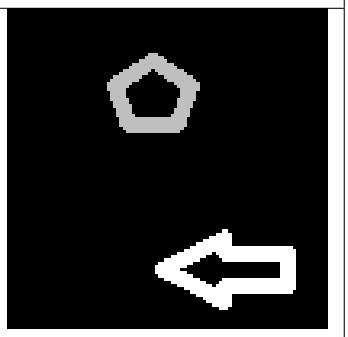
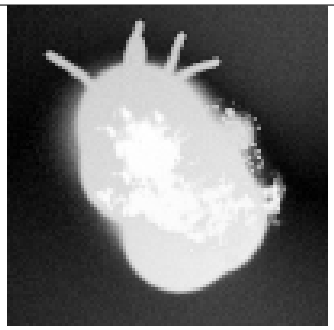
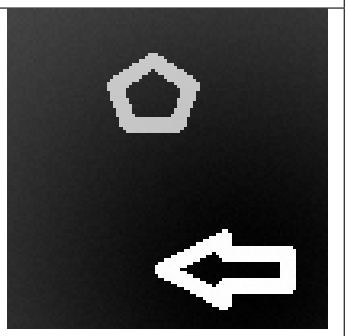
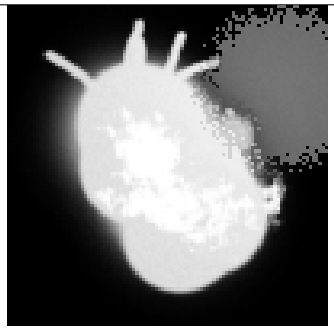
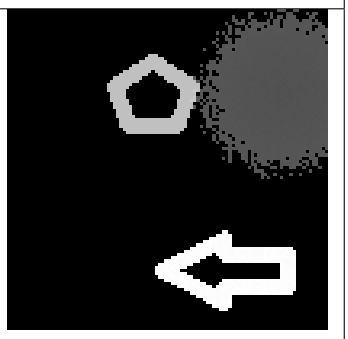
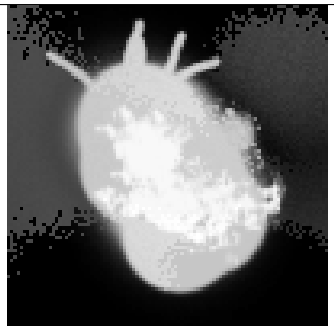
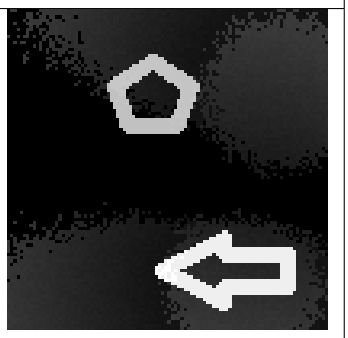
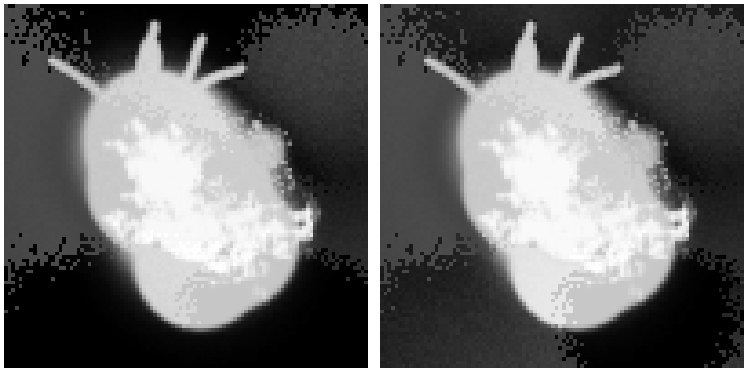
Fundo/Frente	5.6a	5.6b
5.7a		
5.7b		
5.7c		
5.7a, 5.7b		
5.7a, 5.7b, 5.7c		

Tabela 5.2: Mapas de distâncias gerados para as imagens da figura 5.6 sob configurações diferentes das imagens da figura 5.7, no espaço de cores LAB.

Fundo/Frente	5.6a	5.6b
5.7a		
5.7b		
5.7c		
5.7a, 5.7b		
5.7a, 5.7b, 5.7c		



(a) (b)

Figura 5.8: Comparação de dois mapas de distância com ordem de treinamento dos frames de fundo diferente. a) treinamento realizado com as imagens na ordem 5.7a, 5.7b e 5.7c. b) treinamento realizado na ordem 5.7b, 5.7a e 5.7c

de frente, usando as mesmas três imagens para criar um modelo do fundo, apenas em ordem diferente. É importante ressaltar que em casos reais de utilização, com vídeos do fundo para realizar o treinamento, a amostragem dos dados de entrada é muito maior e com menor variação, atenuando a ocorrência desse fenômeno.

5.3 Minimização de Energia

Técnicas de alpha matting que utilizam trimaps como entrada de dados geralmente delegam a responsabilidade de sua criação a um usuário, como comentado na seção 3.1. Um dos motivos principais para isto é que não existe apenas um trimap possível ou ideal para uma imagem; diferentes objetos podem ser considerados frente na mesma imagem, resultando em trimaps distintos. Desta forma, como a escolha do objeto a ser extraído é do usuário, todo processo de obtenção de trimaps é abstraído por praticidade.

No problema proposto para esse trabalho deseja-se sempre segmentar um ator de seu fundo, de modo que já foi escolhida uma abordagem baseada em treinamento e subtração de fundo para lidar com o processo de escolha. Na etapa de cálculo de distâncias as regiões que se distinguem do fundo já representam um esboço do objeto escolhido para ser extraído. No entanto, para produzir um trimap de fato é necessário realizar a segmentação ternária deste esboço para corretamente identificar regiões de frente, fundo e desconhecida.

Como visto na seção 3.1, a qualidade de um trimap afeta diretamente o resultado final de sua aplicação de alpha matting, e é definida pela precisão da demarcação das regiões. Idealmente, a região desconhecida deve ser tão fina quanto possível, cobrindo apenas píxeis onde há uma clara mescla de cores entre frente e fundo, o que implica em uma identificação precisa das cores

dos objetos, considerando inclusive sombra e textura. Este processo de rotulagem das regiões é bastante complicado e até um ser humano inteligente pode encontrar dificuldades em realizá-lo corretamente.

Uma das formas mais simples de resolver esse problema é determinar regiões de extrema diferença que são claramente frente ou fundo e o restante da imagem como desconhecido. Em seguida, analisa-se os píxeis desconhecidos próximos às regiões de frente e fundo e faz-se o questionamento: o quão mais semelhante este píxel é com as cores de frente do que com as regiões de fundo? Píxeis muito mais semelhantes à frente são, então, classificados como frente e vice-versa, de forma que ambas regiões se expandem. A partir de um certo ponto serão observados píxeis com semelhanças de cor similares entre frente e fundo, e estes devem ser deixados como desconhecidos, pois provavelmente apresentam mesclas de cores.

Esta solução simples representa uma maneira intuitiva de classificar as regiões de uma imagem de forma semelhante a como um usuário faria, abstraindo valores. Para uma implementação computacional, no entanto, esta precisa ser quantificada e expressada algoritmicamente. Desta forma, esse processo é modelado na forma de uma problema de minimização de energia, onde se busca encontrar a configuração de rótulos que minimiza uma equação de energia para um determinado grafo. Neste caso, a imagem é considerada como sendo um grafo reticulado em uma vizinhança 4-conexa, podendo pertencer a três classes diferentes: frente, fundo e desconhecido. Ao definir custos de energia, quantifica-se a solução descrita, podendo algoritmicamente serem decididos os limiares de semelhança onde um píxel deixa de ser fundo e passa a ser desconhecido.

Técnicas de minimização de energia são comumente aplicadas à problemas de visão computacional, especialmente envolvendo segmentação e clusterização (CREMERS et al., 2009). Sua principal vantagem é a flexibilidade em modelar problemas de classificação através de custos. Para situações onde existem mais de dois rótulos, no entanto, seu uso pode ser computacionalmente custoso. Como nesse trabalho não existe a necessidade de processamento em tempo real, sua utilização é ideal para modelar o processo de extração de trimaps com simplicidade.

A solução atual apresenta uma implementação das técnicas de graph-cuts (BOYKOV; VEKSLER; ZABIH, 2001) e maxflow (BOYKOV; KOLMOGOROV, 2004) para minimizaçao de energia, além da minimização de energia binária descrita em (KOLMOGOROV; ZABIH, 2002). Esta implementação minimiza funções de energia da forma descrita pela equação 5.6 para cada rótulo, onde existe um conjunto finito de pixels p (a imagem de entrada) e rótulos l (no nosso

caso, frente, fundo e desconhecido). As funções de custo desta equação são arbitrárias para qualquer minimização de energia, de modo que para cada aplicação específica deve-se projetar e definir estas funções.

$$E(l) = \sum_{p} D(p, l_p) + \sum_{p,q} V_{pq}(l_p, l_q) \tag{5.6}$$

A função $D(p, l_p)$ descreve o custo individual de se atribuir um rótulo l a um determinado píxel p independente, ou seja, sem considerar sua vizinhança. Geralmente são utilizadas funções específicas diferentes para cada rótulo. Para a aplicação desse trabalho foram definidas diversas funções de custo individual, porém todas baseadas na distância de cor entre o modelo do fundo e o quadro de entrada.

$$D(p, l_p) = \begin{cases} d_p^2 & l = 0 \\ \gamma^{Ind} & l = 1 \\ (1 - d_p)^2 & l = 2 \end{cases} \qquad 0 \leq d_p \leq 1 \tag{5.7}$$

A equação 5.7 apresenta um exemplo de função de custo individual baseada na distância. Nesta forma, d_p é a distância da cor entre o píxel de entrada e o modelo conhecido do fundo em uma determinada posição, normalizado na etapa anterior entre 0 e 1. O custo deste píxel ser rotulado como fundo ($l = 0$) corresponde ao quadrado da distância d_p encontrada, de modo que quanto mais um píxel assemelha-se com o modelo conhecido do fundo, menor é o custo deste ser considerado pertencente ao fundo. O termo de frente ($l = 2$) acentua esse aspecto ao utilizar o quadrado do inverso da distância d_p em relação ao espaço normalizado entre 0 e 1. O custo de se atribuir a um píxel o rótulo de desconhecido ($l = 1$) é invariável entre píxeis da mesma imagem, representado pela constante γ^{Ind}. Idealmente este valor deve ser suficientemente baixo para que áreas de transição possam ser classificadas como desconhecidas e suficientemente alto para minimizar a ocorrência de regiões desconhecidas.

O termo $V_{pq}(l_p, l_q)$ é chamado termo de suavização, representando um custo espacial em relação à atribuição de um rótulo l_p próximo à um rótulo l_q, com o objetivo de atenuar variações locais de rótulo. Este termo é calculado usando uma relação estática de custo de se ter rótulos diferentes adjacentes, multiplicado por um mapa da imagem que representa a propensão local de haver uma mudança de rótulo em uma determinada localidade.

A tabela 5.3 apresenta os custos de energia relativos entre rótulos vizinhos. Estes custos servem para modelar o formato esperado de trimaps, como visto na seção 3.1. Em um trimap, sempre espera-se que exista uma região desconhecida entre áreas de frente e fundo, pois transi-

Tabela 5.3: Custos de energia relativos entre rótulos vizinhos.

Rótulo	1	2	3
1	0	γ^{Comp}	∞
2	γ^{Comp}	0	γ^{Comp}
3	∞	γ^{Comp}	0

ções são sempre consideradas suaves. Desta forma, o custo energético de um píxel rotulado de frente ser vizinho à um rotulado de fundo é definido como muito grande (∞). O custo estático de transição entre rótulos do tipo frente e fundo para trimaps é definido por γ^{Comp}, que é uma constante arbitrária. Quanto menor o valor de γ^{Comp}, mais granular é a zona desconhecida.

Para representar a tendência local de mudança de rótulos, é extraída uma forma de laplaceanos de gaussianos da imagem, de modo a indicar regiões próximas a bordas, que seriam mais prováveis de sofrer mudanças entre frente, fundo e desconhecido.

Na configuração de custos de energia utilizada, existindo 3 píxeis adjacentes de rótulos l_p, l_q e l_r, não é possível obter um custo menor de energia entre l_p e l_q fazendo um atalho por l_r. Em termos formais $V_{pq}(l_p, l_p) + V_{pq}(l_q, l_q) \leq V_{pq}(l_p, l_q) + V_{pq}(l_q, l_p)$, mas $V_{pq}(l_p, l_p) + V_{pq}(l_q, l_r) > V_{pq}(l_p, l_r) + V_{pq}(l_q, l_p)$, o que segundo a terminologia de Kolmogorov (KOLMOGOROV; ZABIH, 2002) significa que o custo de energia binário de passos do algoritmo de troca é regular, mas de expansão é irregular. Na prática, isto implica na impossibilidade de utilizar um processo de minimização de energia por um algoritmo de expansão alfa, sendo necessário aplicar o algoritmo de trocas alfa. Isto é uma limitação, pois a utilização de trocas alfa não garante o mínimo global de energia e possui um desempenho menor. Na subseção 5.3.1 ambos algoritmos são descritos de forma breve.

Ao encontrar uma configuração de energia mínima segundo os parâmetros estabelecidos, um trimap é gerado, como pode ser visto na imagem 5.1. Este trimap segmenta a imagem em três cores: branco para píxeis de frente, cinza para píxeis desconhecidos e preto para píxeis de fundo. Um ponto importante a ser considerado é que a minimização de energia é extremamente sensível à variação de parâmetros, de modo que uma pequena mudança em um dos termos pode produzir resultados muito diferentes.

5.3.1 Algoritmos de minimização de energia

Os algoritmos de minimização de energia utilizam as mesmas funções de custo da equação 5.6, mas o processo utilizado para chegar ao estado de energia mínima é diferente. De uma forma geral, ambos constróem grafos ligando nodos píxeis e classes com pesos de arestas re-

presentando custos relativos de energia. Após processos iterativos onde se comparam valores de classes diferentes para todos os píxeis e suas relações, são realizados cortes de grafo que minimizam os pesos das arestas e, com isto, os custos de energia. As conexões cortadas ao final determinam a que classe um píxel pertence.

Como foi citado, os custos descritos na tabela 5.3 criam uma configuração energética na qual só é possível de se aplicar o algoritmo de trocas alfa. A substituição destes valores por valores comparativamente inferiores de forma a tornar $V_{pq}(l_p, l_p) + V_{pq}(l_q, l_r) \leq V_{pq}(l_p, l_r) + V_{pq}(l_q, l_p)$ permite a utilização de expansão alfa, mas perde-se a garantia de que píxeis de frente e fundo nunca sejam vizinhos.

Nas subseções a seguir a forma geral dos algoritmos de expansão e trocas alfa são descritos de forma simples para fins de entendimento. Uma explicação detalhada de seu funcionamento pode ser lida no seu artigo original por Kolmogorov (KOLMOGOROV; ZABIH, 2002). Ademais, as aplicações de cortes de grafos são descritas por Boykov ((BOYKOV; VEKSLER; ZABIH, 2001) e (BOYKOV; KOLMOGOROV, 2004)).

5.3.1.1 Algoritmo de expansão alfa

O algoritmo de expansão alfa considera uma classe α atual e tenta segmentar todos os píxeis pertencentes à α dos píxeis não pertencentes à α ($\bar{\alpha}$) através de cortes de grafo. Este algoritmo itera substituindo α por todos rótulos existentes possíveis até convergir.

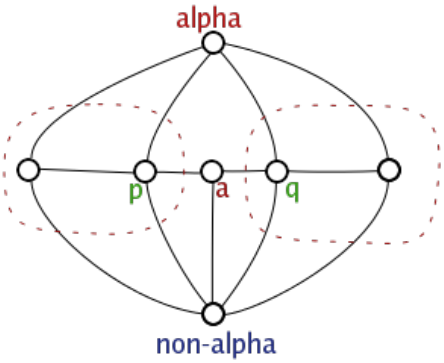


Figura 5.9: Representação do grafo entre dois píxeis p e q usado no algoritmo de expansão alfa. (LOMBAERT, 2006)

Para cada iteração, a região Ω_α de α pode apenas se expandir. A figura 5.9 mostra a configuração de uma parte do grafo considerando a relação entre dois nodos píxeis vizinhos p e q. Se estes nodos não forem pertencentes ao mesmo rótulo, é criado um nodo intermediário a que possa representar o "atalho" de custos citado anteriormente. Os pesos w das arestas são

definidos seguindo os custos da equação 5.6 da seguinte forma:

$$
\begin{aligned}
w(\alpha, p) &= D(p, \alpha) \\
w(\bar{\alpha}, p) &= \begin{cases} D(p, l_p) & p \notin \Omega_\alpha \\ \infty & p \in \Omega_\alpha \end{cases} \\
w(p, a) &= V(l_p, \alpha) \\
w(a, q) &= V(\alpha, l_q) \\
w(\alpha, \bar{\alpha}) &= V(l_p, l_q) \\
w(p, q) &= V(l_p, l_q)
\end{aligned}
\tag{5.8}
$$

O nodo p é considerado pertencente à α quando a aresta $p - \alpha$ estiver cortada, e pertencente à $\bar{\alpha}$ quando a aresta $p - \bar{\alpha}$ estiver cortada. Desta forma, o peso da aresta $p - alpha$ vai ser o custo relativo estático do píxel p receber o rótulo α. O peso da aresta $p - \bar{\alpha}$ depende se o nodo-píxel p já está inserido na região Ω_α, pois a mesma só pode ser expandida quando estiver sendo avaliada, resultando em um custo infinito para evitar a troca de rótulo.

5.3.1.2 Algoritmo de trocas alfa

Diferentemente da expansão alfa, o algoritmo de trocas alfa define duas classes α e β dentre as existentes e tenta realizar uma segmentação de nodos entre ambas. Para cada iteração, combinações diferentes de α e β são alternadas até a convergência ser atingida.

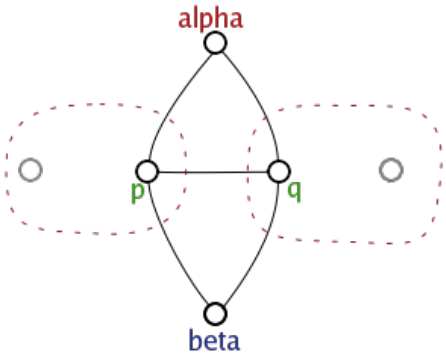


Figura 5.10: Representação do grafo entre dois píxeis p e q usado no algoritmo de trocas alfa. (LOMBAERT, 2006)

Na imagem 5.10 pode ser vista a configuração do grafo utilizado para as trocas alfa entre dois nodos píxeis vizinhos. Este grafo é mais simples do que o visto na figura 5.9, pois não é utilizado o nó intemediário a, e a ausência de arestas adicionais laterais indicam a exclusividade da comparação entre as classes α e β escolhidas, onde outros nós são desconsiderados. Os pesos w das arestas no algoritmo de troca alfa são definidos por:

$$
\begin{aligned}
w(\alpha, p) &= D(p, \alpha) + \sum_{q} V_{pq}(l_q, \alpha) \\
w(\beta, p) &= D(p, \beta) + \sum_{q} V_{pq}(l_q, \beta) \\
w(p, q) &= V(\alpha, \beta)
\end{aligned}
\tag{5.9}
$$

Em ambos os casos $w(\alpha, p)$ e $w(\beta, p)$, considera-se q como sendo um nodo píxel de uma vizinhança quatro-conexa ao redor de p, não pertencente às regiões Ω_α e Ω_β, enquanto p pertence à região Ω_α ou Ω_β. Da mesma forma que na expansão, o nó p é considerado pertencente à α quando a aresta $p - \alpha$ estiver cortada, e pertencente à β quando a aresta $p - \beta$ estiver cortada.

Os pesos das arestas nesta forma remetem diretamente à equação 5.6, pois cada nodo no grafo representa uma entidade com mais informação agregada, enquanto na expansão alfa custos são distribuídos em nós auxiliares adicionais.

5.4 Aplicação de Alpha Matting

Ao final da etapa anterior é gerado um vídeo onde cada quadro corresponde exatamente ao trimap do quadro análogo do vídeo de entrada. Deste modo, pode ser utilizada qualquer técnica de alpha matting que utilize um trimap e uma imagem como entrada para obter o mapa de transparências e a cor do plano de frente, como descrito no capítulo 3.

6 IMPLEMENTAÇÃO

A implementação das etapas descritas no capítulo 5 foi realizada na plataforma Matlab (MATLAB, 2011), com exceção da aplicação do alpha matting, que utiliza um programa implementado em C++ e GLSL. O motivo principal para a utilização do Matlab é a facilidade de prototipação e análise das técnicas desenvolvidas. Sua maior limitação é o baixo desempenho, que pode ser observado pelos altos tempos de processamento demonstrados no capítulo 7. Como discutido anteriormente, a solução desenvolvida apresenta um alto grau de paralelismo, o que permite que o fator desempenho seja solucionado futuramente com a utilização de processamento parelelo e/ou em GPU.

As técnicas de treinamento de fundo e cálculo de distância utilizando codebooks foram completamente programadas em Matlab, com exceção da conversão entre espaços de cores e da métrica ΔE^*_{ab}. Para estes fins foram utilizados os códigos-fonte disponibilizados por Gaurav Sharma (SHARMA; WU; DALAL, 2005) e Pascal Getreuer (GETREUER, 2011), respectivamente. Testes preliminares do algoritmo de codebooks usaram a implementação da biblioteca OpenCV (BRADSKI, 2000), bem como uma implementação própria alternativa com o auxílio da mesma. Ademais, os objetos que armazenam o modelo do fundo na forma de codebooks são facilmente armazenados em arquivos e carregados posteriormente dentro do contexto do matlab.

A abordagem de minimização de energia desenvolvida utiliza o wrapper de graph cuts para Matlab desenvolvido por Shai Bagon (BAGON, 2006), o qual provê uma interface para a aplicação de métodos de minimização de energia de autoria de Olga Veksler (BOYKOV; FUNKALEA, 2006). Deste modo, os custos de energia são resolvidos externamente ao Matlab.

De uma forma geral, a solução implementada não constitui uma aplicação singular e completa, sendo composta de diversos módulos e scripts separados para casos de testes diferentes. A entrada e saída de dados, especialmente de vídeos e imagens, é feita pelas bibliotecas providas

pelo Matlab, até a geração de trimaps. Existem dois conjuntos de funções do Matlab análogas, uma utilizando o espaço de cores RGB e a outra utilizando o espaço de cores CIELAB. As principais funções para ambos os modos são:

- cbImage{RGB,LAB}: gera um codebook baseado em um conjunto de imagens e o utiliza para calcular as distâncias de uma imagem de entrada, salvando a saída em outra imagem. Foi criada especificamente para testes com imagens sintéticas e fotográficas, ao invés de vídeos;

- getTrimapFromDistances: recebe um objeto de imagem do Matlab e aplica a minimização de energia sobre ele, retornando um objeto de imagem do Matlab. É usada principalmente internamente em outras funções para a fase de minimização de energia, mas pode ser chamada para calcular o trimap em uma imagem gerada pela função anterior;

- createCodeBookVideo{RGB,LAB}: recebe um vídeo de treinamento e um intervalo de frames e aplica o treinamento do fundo para os frames do vídeo nesse intervalo, salvado um objeto do Matlab com a estrutura de codebooks que modela o fundo. Sua principal finalidade é tornar o processo de testes mais prático ao permitir que somente o treinamento seja executado e armazenado;

- cbVideo{RGB,LAB}: permite várias utilizações diferentes, recebendo um vídeo de treinamento ou um arquivo de codebook gerado juntamente com um vídeo de entrada e gerando trimaps para um frame ou um conjunto de frames, salvando estes em arquivo de vídeo. É a função de maior importância e abrangência para a segmentação de vídeos.

No ambiente de desenvolvimento do Matlab estas funções são utilizadas na forma de scripts, sendo chamadas quando necessário através do console. Como o Matlab possui um chamado "espaço de trabalho" onde variáveis e objetos permanecem acessíveis pervasivamente, estas funções podem ser chamadas separadamente e em ordens variadas para realizar testes. Supondo por exemplo que tenhamos um arquivo chamado "foreground.png", representando uma imagem composta com um objeto a ser segmentado e três imagens "background1.png", "background2.png" e "background3.png". Para executar um teste que primeiro gere um mapa de distâncias da primeira imagem em relação a um modelo do fundo em CIELAB com as outras três imagens e, em seguida, calcula seu trimap, as funções seriam chamadas no console do matlab na seguinte sequência:

1. cbImageLAB(char('background1.png','background2.png','background3.png'), 'foreground.png','distancias.png')

2. imwrite(getTrimapFromDistances(imread('distancias.png')),'trimap.png','PNG')

A primeira chamada cria um vetor de strings com os nomes dos arquivos de imagens que serão usadas para o fundo e usa isso como entrada da função cbImageLAB, que ao ser executada calcula o mapa de distâncias e o salva na imagem 'distancias.png'. A função getTrimapFromDistances opera com objetos de imagem do Matlab, de modo que para passar seu parâmetro usa-se a função de leitura de imagens do Matlab imread e, posteriormente, imwrite para salvar a imagem de saída no formato "PNG" com o nome "trimap.png".

Um teste de vídeos utilizando dois vídeos "entrada.avi" e "fundo.avi", com o objetivo de analisar o efeito da quantidade de frames no resultado final, por exemplo, poderia ser feito da seguinte forma:

1. generateCodeBookVideoLAB('fundo.avi','codebook1-20',1,20)

2. generateCodeBookVideoLAB('fundo.avi','codebook1-200',1,200)

3. codebook1 = load('codebook1-20.MAT')

4. cbVideoLAB(,'entrada.avi','trimap1-20.avi', , ,1,0,codebook1)

5. codebook2 = load('codebook1-200.MAT')

6. cbVideoLAB(,'entrada.avi','trimap1-200.avi', , ,1,0,codebook2)

Inicialmente são gerados arquivos MAT contendo os codebooks com o modelo do fundo usando os primeiros 20 ou os primeiros 200 frames dos vídeos do fundo, através da função generateCodeBookVideoLAB e salvos com os nomes "codebook1-20.MAT" e "codebook1-200.MAT". Estes arquivos são carregados na memória para o "espaço de trabalho" do Matlab através da função load. Finalmente, a função cbVideoLab é chamada uma vez para cada modelo diferente do fundo, gerando vídeos de trimaps de saída diferentes. Os parâmetros em branco são relativos à possibilidade de se usar um vídeo de entrada diretamente ao invés de um objeto contendo o modelo do fundo. Os valores 1 e 0 indicam que o intervalo desejado para a geração dos trimaps é a partir do frame 1 (o Matlab indexa valores iniciando em 1 ao invés de 0) até o final do vídeo.

Para a aplicação de alpha matting, os trimaps gerados tanto em vídeo quanto em imagens devem ser passados como entrada para uma aplicação externa que irá efetuar o processo de cálculo de transparência e cores.

7 RESULTADOS

Devido a natureza desse trabalho, diversos tipos de testes são necessários para avaliar a eficácia da solução desenvolvida, primeiramente por seu objeto central ser vídeos digitais, que apresentam uma grande quantidade de dados e variabilidade e, em segundo lugar, pelo enorme conjunto de parâmetros presentes.

Neste capítulo serão apresentados os resultados de testes feitos sob configurações de parâmetros e entradas diferentes, discutindo suas implicações. Os conjuntos de parâmetros utilizados em um determinado teste são apresentados na forma de tabelas, como pode ser visto na tabela ilustrativa 7.1. KLCH se refere aos pesos usados no cálculo da distância ΔE^*_{ab} e são relativos à luminância, croma e matiz, respectivamente. Neste exemplo, os valores para as funções de custo exemplificados na seção 5.3 são transcritos no formato de parâmetros, onde $D(p, l_p)$ é a função de custo individual de um rótulo, descrita geralmente em termos da distância de cor de um píxel (d_p). Os diferentes rótulos de frente, desconhecido e fundo são representados pelos indicadores l_f, l_d e l_b, respectivamente.

Nas próximas seções este modelo de tabela será utilizado para especificar os parâmetros utilizados nos testes, quando necessário. Os parâmetros KLCH são exclusivamente relacionados à distâncias sobre um espaço CIELAB. Em testes onde são utilizados mapas de distâncias calculados sobre o espaço RGB estes valores não têm influência no resultado final.

Tabela 7.1: Exemplo de tabela de parâmetros.

	KLCH		$D(p, l_p)$	$V_{pq}(l_p, l_q)$	l_f	l_d	l_b
L	1	l_f	d_p^2	l_f	0	γ^{Comp}	∞
C	1	l_d	γ^{Ind}	l_d	γ^{Comp}	0	γ^{Comp}
H	1	l_b	$(1 - d_p)^2$	l_b	∞	γ^{Comp}	0

Tabela 7.2: Tabela de parâmetros para o teste das imagens da subseção 5.2.

	KLCH		$D(p, l_p)$	$V_{pq}(l_p, l_q)$	l_f	l_d	l_b
L	1	l_f	d_p^2	l_f	0	0.001	2
C	1	l_d	1.2	l_d	0.001	0	0.001
H	1	l_b	$(1 - d_p)^2$	l_b	2	0.001	0

7.1 Testes com imagens

Como discutido anteriormente, para a abordagem adotada neste trabalho vídeos são tratados como imagens individuais em sequência. Desta forma, é fundamental testar a aplicabilidade da solução desenvolvida primeiramente em imagens. Esta seção está dividida em duas categorias de imagens testadas: sintéticas e fotográficas.

7.1.1 Imagens sintéticas

Imagens sintéticas são imagens construídas em um software de edição de imagens ou por algum processo algorítmico, sem aquisição de informação externa. Este tipo de imagem é especialmente apropriado para se testar aspectos específicos da solução, pois é possível definir características únicas e condições ideais, podendo controlar os efeitos de ruído e iluminação, por exemplo.

As imagens utilizadas no exemplo de mapas de distância da subseção 7.4 são algumas das imagens sintéticas construídas para a realização de testes. Para uma avaliação inicial dos resultados, os mapas de distância gerados para a imagem 5.6a sob diferentes configurações de fundo são apresentados nas tabelas 7.3 e 7.4. Elas demonstram lado a lado os mapas de distância e os trimaps gerados a partir dos mesmos.

Os parâmetros usados para este teste são apresentados na tabela 7.2. Seus valores são próximos aos propostos na seção 5.3, definindo algumas variáveis que haviam permanecido em aberto. O parâmetro γ^{Ind}, que se refere ao custo individual de se atribuir o rótulo desconhecido l_d à um píxel é definido como 1.2, valor arbitrário que apresentou resultados bons para fins de teste. Os custos entre rótulos levam em conta que todas as distâncias são normalizadas no intervalo $(0, 1)$, de modo que ∞, um custo muito grande, assume o valor 2 e sua contrapartida γ^{Comp} é definida como 0.001.

O ineficácia dos resultados obtidos utilizando o espaço RGB é bastante evidente neste conjunto de imagens. A falta de suavidade na expressão das distâncias de cor, somada à sensibilidade a variações de luz e ruído faz com que os trimaps gerados sejam pouco adequados. Mesmo

Tabela 7.3: Trimaps gerados para os diferentes mapas de distância das imagens da tabela 5.1, no espaço de cores RGB.

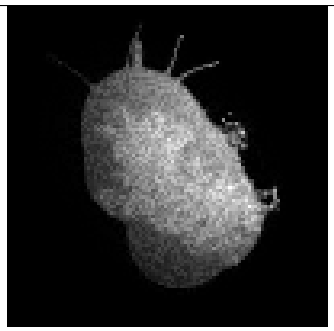
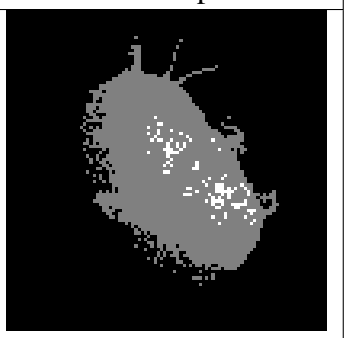
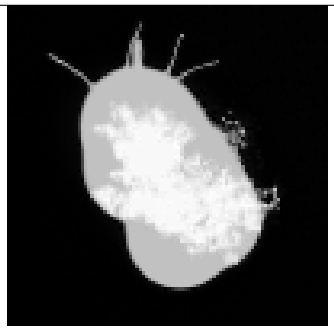
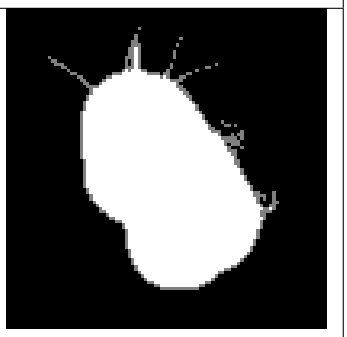
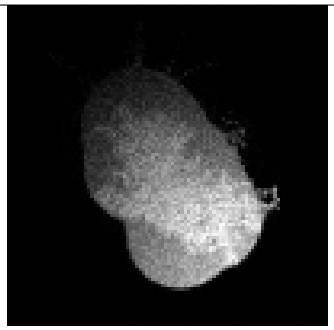
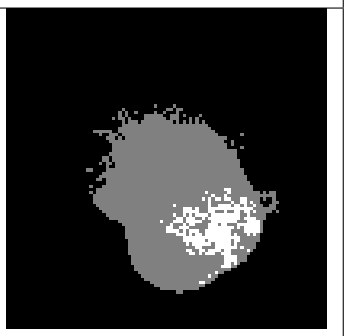
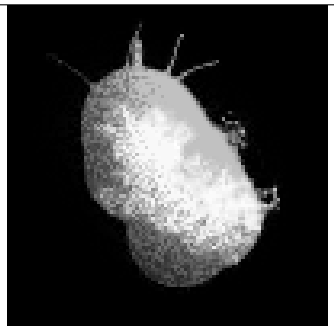
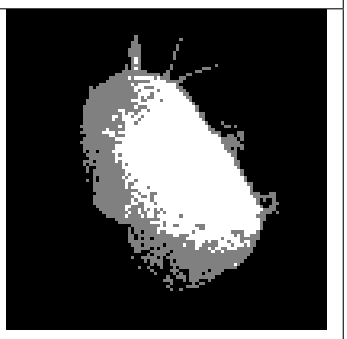
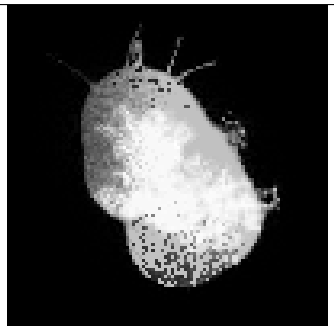
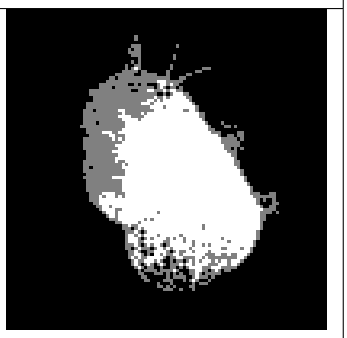
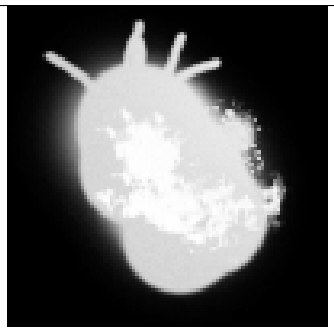
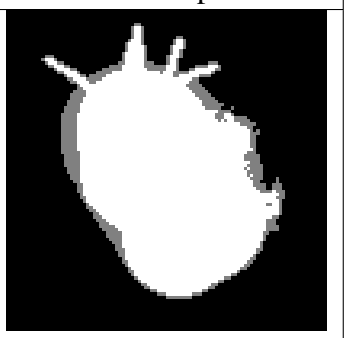
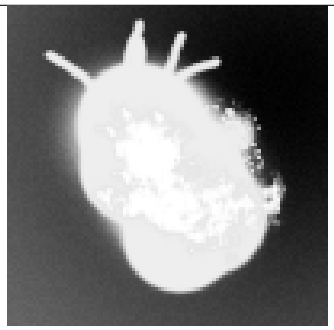
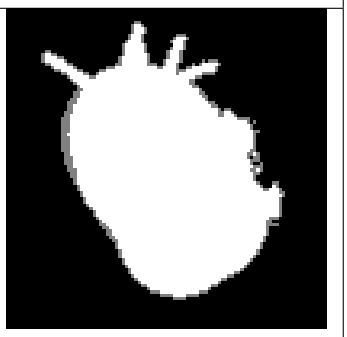
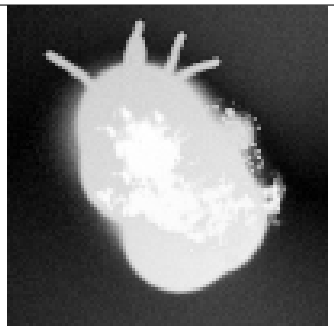
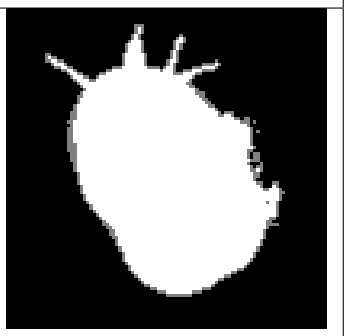
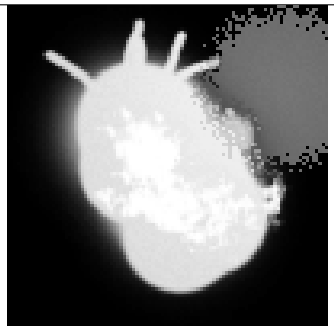
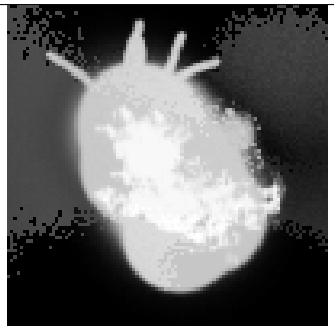
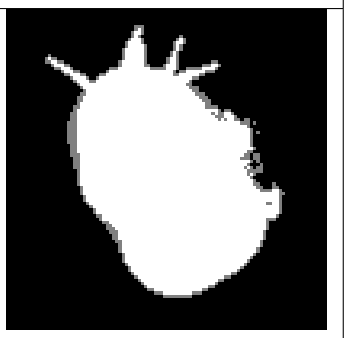
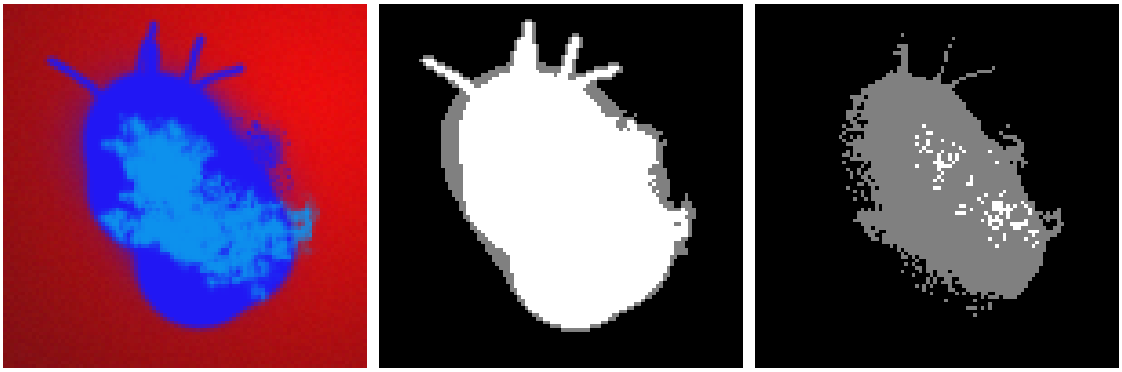
Fundo	Distâncias	Trimap
5.7a		
5.7b		
5.7c		
5.7a, 5.7b		
5.7a, 5.7b, 5.7c		

Tabela 7.4: Trimaps gerados para os diferentes mapas de distância das imagens da tabela 5.2, no espaço de cores LAB.

Fundo	Distâncias	Trimap
5.7a		
5.7b		
5.7c		
5.7a, 5.7b		
5.7a, 5.7b, 5.7c		



(a) (b) (c)

Figura 7.1: Comparação da imagem original com os trimaps gerados no caso de teste ideal. a) fundo original. b) trimap gerado com distâncias no espaço de cor CIELAB. c) trimap gerado com distâncias no espaço de cor RGB.

nas situações onde os mapas de distância gerados no espaço de cor CIELAB estão com menor qualidade, os trimaps gerados são muito mais condizentes com a proposta deste trabalho.

De uma forma geral, pode ser observada a tendência da minimização de energia em concentrar a distribuição de rótulos de frente em torno da maior concentração possível de distâncias altas. Isto é visível no trimap gerado sobre o espaço CIELAB para as imagem treinadas com o fundo 5.7b e 5.7c. Comparando com a imagem de fundo 5.7a acima, houve uma redução da zona desconhecida e expansão das áreas de frente e fundo em ambos os casos. Essa expansão é influenciada pela posição da distribuição de cores semelhantes, de modo que a imagem de fundo 5.7b, que possui uma grande área cinza no seu canto inferior direito apresenta uma expansão da área de frente para o canto superior direito, enquanto as imagens de fundos 5.7c e 5.7a, 5.7b e 5.7c são mais centralizadas. Este fenômeno ocorre pois os custos individuais de energia utilizam os quadrados das distâncias de píxeis, ou seja, valores maiores (ou menores, em contraste à valores medianos) possuem maior importância no processo de minimização de energia.

Este comportamento da minimização de energia dificulta a segmentação de múltiplos objetos desconexos, pois provavelmente um dentre os diversos objetos, aquele com a maior concentração de distâncias altas, seria escolhido como ponto de partida para a minimização. A partir deste ponto, seria mais difícil de propagar rótulos de frente através da área de rótulos de fundo e possivelmente muitos objetos seriam marcados apenas como regiões desconhecidas. Como discutido anteriormente, isto não é particularmente limitante para a proposta desse trabalho, pois existe a necessidade de se segmentar apenas um ator por cena.

Na imagem 7.1 é apresentada uma comparação da imagem original com os trimaps gera-

Tabela 7.5: Tabela de parâmetros para testes com imagens sintéticas de alta resolução.

	KLCH		$D(p, l_p)$	$V_{pq}(l_p, l_q)$	l_f	l_d	l_b
L	1	l_f	d_p^2	l_f	0	0.001	2
C	1	l_d	$0.5\bar{d_p}$	l_d	0.001	0	0.001
H	1	l_b	$(1 - d_p)^2$	l_b	2	0.001	0

dos no caso ideal, ou seja, com seu fundo exato. O trimap gerado com distâncias no espaço CIELAB apresenta um resultado muito satisfatório, delimitando áreas de nuância de cor como desconhecida e definindo corretamente a frente em partes estreitas. É possível observar também que na parte direita do objeto azul existem regiões com "respingos" de tinta azul, que foram em grande maioria marcadas como desconhecidas e em parte como fundo. Apesar desta classificação não corresponder totalmente à extração da imagem de frente, é importante notar que estes "respingos" não estão conectados diretamente ao objeto principal.

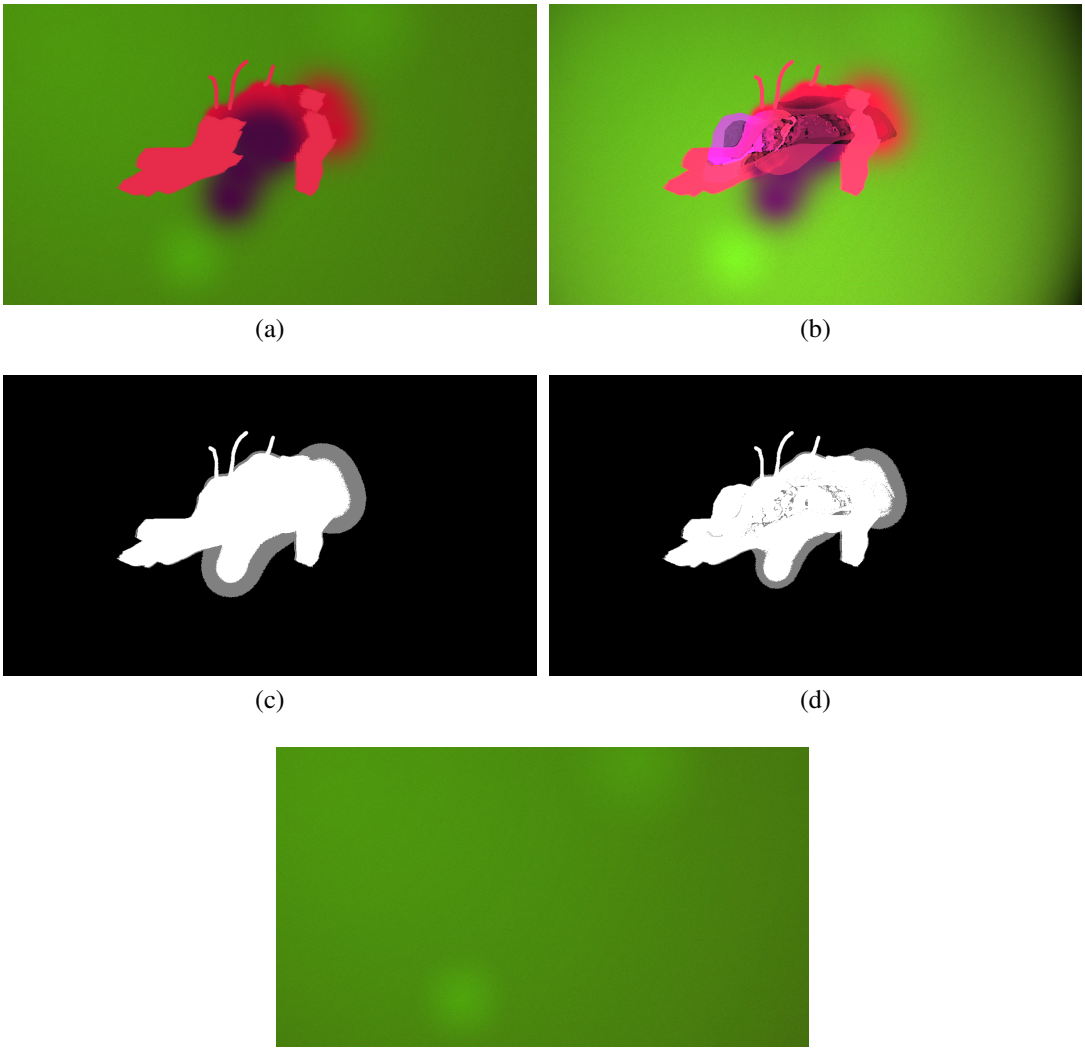
O mapa de distâncias calculado no espaço RGB é pouco compatível com a minimização de energia, havendo muita variação em píxeis próximos e prejudicando o resultado final. O resultado RGB mais adequado é o de fundo 5.7b (o fundo de um tom constante), e este ainda assim possui um detalhamento inferior à sua contrapartida CIELAB.

7.1.1.1 Imagens sintéticas de alta resolução

A resolução das imagens ou vídeos utilizados causa influências significativas na geração de seus trimaps. Isto ocorre pois ao aumentar a resolução de uma imagem sua área aumenta e as proporções de regiões de frente, fundo e desconhecidas crescem de formas diferentes. Ambas regiões de frente e fundo crescem com uma relação quadrática com o aumento de área, sendo que a região de fundo ainda apresenta um crescimento linearmente superior e a região desconhecida, em sua grande maioria perimetral, cresce de forma linear. O processo de minimização de energia precisa encontrar um balanço ideal entre estes três rótulos, de modo que parâmetros podem precisar serem modificados para compensar por esta diferença de crescimento.

A utilização de parâmetros que dependam de estatísticas da imagem, como o caso da configuração da tabela 7.5, permite uma flexibilidade muito maior sobre variações de resolução. Neste caso, o custo individual de se atribuir o rótulo desconhecido para um píxel é definido como $0.5\bar{d_p}$, ou seja, a metade da média dos valores de altura encontrados na imagem. Na imagem 7.2 é mostrado um conjunto de teste de imagens de resolução 720p e seus resultados utilizando esse conjunto de parâmetros.

Para este teste as distâncias foram calculadas apenas no espaço de cor CIELAB, pois a



(a) (b)

(c) (d)

(e)

Figura 7.2: Comparação de imagens de teste de alta resolução com parâmetro flexível e seus trimaps gerados. a) Imagem simples de alta resolução. b) imagem de alta resolução com adição de efeitos de composição, ruído e iluminação. c) trimap gerado com distâncias no espaço de cor CIELAB para a imagem (a). d) trimap gerado com distâncias no espaço de cor CIELAB para a imagem (b). e) fundo usado para o treinamento de ambos os testes, corresponde ao fundo da imagem (a).

eficácia da utilização de distâncias calculadas no espaço de cor RGB é bastante limitada. Neste caso, a imagem 7.2a, com a imagem de fundo 7.2e e seu trimap gerado 7.2c configuram um caso ideal de extração de fundo. Já a imagem 7.2b teve a adição de diversos efeitos de composição de imagem, uma camada de ruído e iluminação adicional, sofrendo até vigneting no seu lado direito. O objetivo destas modificações é testar efeitos adversos no resultado final, que são especialmente críticos em altas resoluções. Como pode ser observado no trimap resultante 7.2d, a segmentação é bastante satisfatória, com exceção da identificação de diversas micro-regiões desconhecidas no interior do objeto central.

Este resultado demonstra a aplicabilidade da solução em resoluções altas, considerando ajustes necessários de parâmetros.

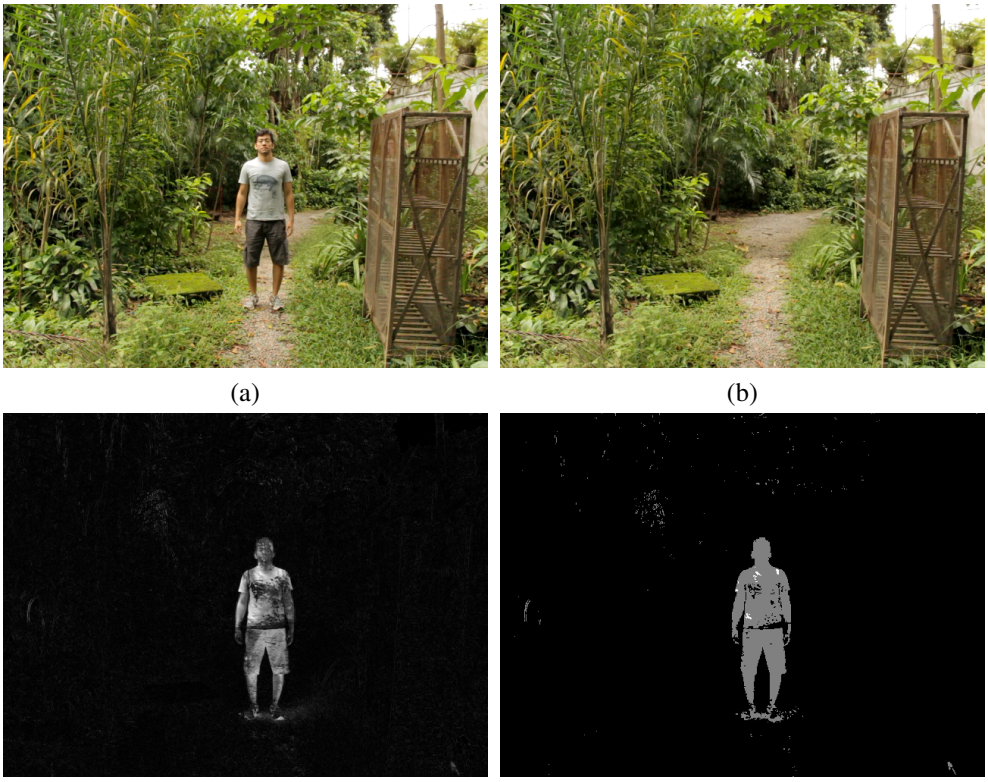
7.2 Imagens fotográficas

A utilização do termo imagens fotográficas nessa seção refere-se à imagens obtidas através de algum dispositivo de captura, como uma câmera digital ou scanner. A utilização de imagens fotográficas da mesma forma demonstrada nos testes de imagens sintéticas, com uma imagem de frente e uma imagem de fundo para a segmentação faz muito pouco sentido no contexto desse trabalho. Isso ocorre especialmente devido à natureza de imagens capturadas, altamente suscetíveis a variações do ambiente entre duas capturas consecutivas. Como o foco desse trabalho é a segmentação de vídeos, esse problema é solucionado no âmbito de vídeos através de um treinamento do fundo que utiliza diversas imagens. Para fins ilustrativos, essa seção apresentará alguns testes sobre imagens fotográficas.

Nas figuras 7.3 e 7.4 são apresentados dois conjuntos de imagens na qual se pode observar alguns dos problemas mencionados. Ambos os pares de frente e fundo foram obtidos usando uma câmera fotográfica digital ao longo de alguns segundos. Para cada teste, apesar da câmera permanecer estática e filmando o mesmo cenário, diversas variações ocorrem neste período de tempo. Os parâmetros usados foram os descritos na tabela 7.5.

A figura 7.3 apresenta um fundo complexo, com muitas folhagens se movimentando. Como pode ser visto no seu mapa de distâncias na imagem 7.3c, este movimento causa mudanças no fundo conhecido, que é apenas composto pela imagem 7.3b, gerando uma espécie de ruído. O trimap resultante é bastante inadequado, pois não existe uma concentração ideal de distâncias altas encontradas.

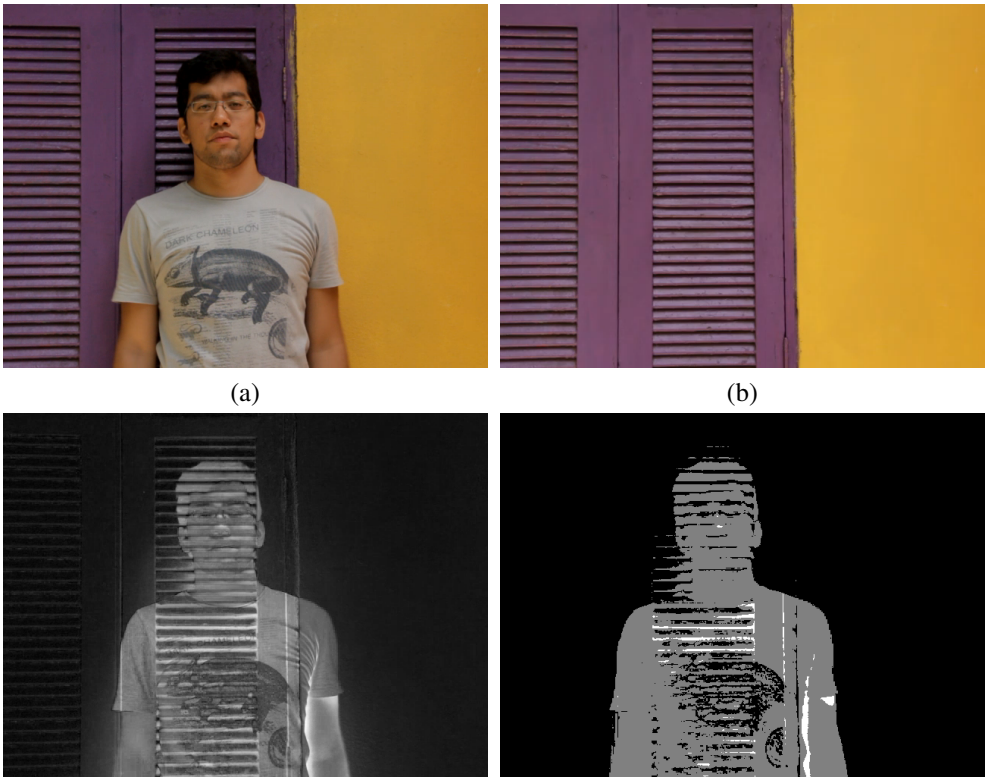
O efeito da variação de iluminação é demonstrado na figura 7.4. Neste exemplo, pode-



(a) (b)

(c) (d)

Figura 7.3: Demonstração do efeito de variações no fundo em imagens fotográficas. a) imagem de entrada. b) Fundo usado. c) mapa de distâncias calculado usando o espaço de cor LAB. d) Trimap resultante.



(a) (b)

(c) (d)

Figura 7.4: Demonstração do efeito de variações na iluminação em imagens fotográficas. a) imagem de entrada. b) Fundo usado. c) mapa de distâncias calculado usando o espaço de cor LAB. d) Trimap resultante.

se notar que a imagem de fundo 7.4b está ligeiramente mais clara que a imagem de entrada 7.4a, afetada por iluminação natural. No mapa de distâncias da imagem 7.4c é possível ver claramente uma espécie de "negativo" da janela pertencente ao fundo, que, idealmente, deveria estar completamente escura. Novamente, o resultado obtido é de baixa qualidade.

Mesmo utilizando estes mapas de distância inadequados é possível obter resultados relativamente melhores através de mudanças nos parâmetros utilizados. Estes ajustes, no entanto, são em sua maioria ad hoc e só se aplicam para casos específicos. A imagem 7.5 representa o cálculo de trimaps de ambos os casos anteriores utilizando os parâmetros da tabela 7.6, que foram especialmente ajustados para corrigir alguns dos problemas do primeiro conjunto de imagens. Pode ser observado que esta mudança de parâmetros, apesar de mudar drasticamente o trimap gerado para o segundo caso de teste, ainda assim apresenta resultados insatisfatórios.

7.3 Testes com vídeos

Na seção anterior diversos aspectos da solução desenvolvida foram avaliados e discutidos a partir de testes realizados com imagens. Muitas das questões levantadas e particularidades se

Tabela 7.6: Tabela de parâmetros ajustada para melhor se adequar ao caso de teste da imagem 7.3.

	KLCH		$D(p, l_p)$	$V_{pq}(l_p, l_q)$	l_f	l_d	l_b
L	1	l_f	d_p^2	l_f	0	0.001	2
C	1	l_d	$0.3\bar{d_p}$	l_d	0.001	0	0.001
H	1	l_b	$((1 - d_p) * 0.15)^2$	l_b	2	0.001	0

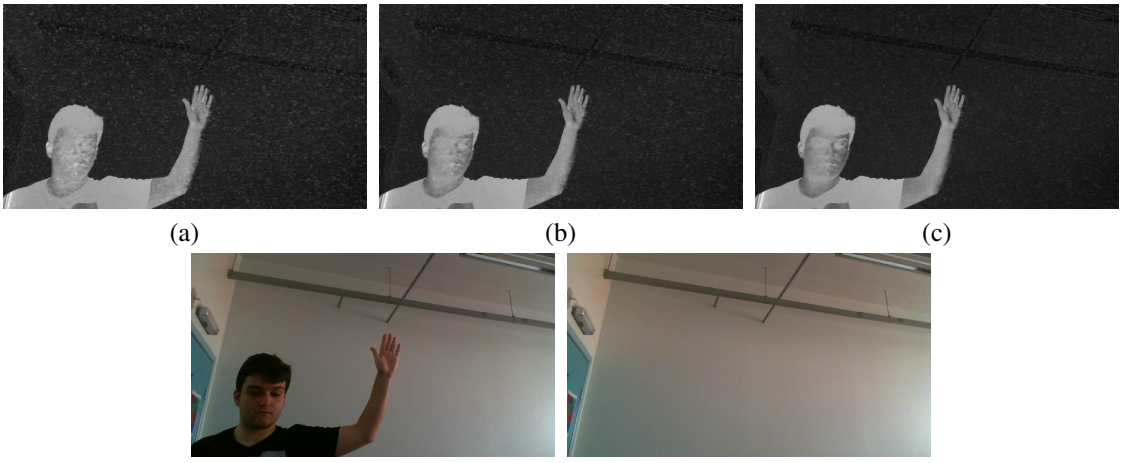
(a) (b)

Figura 7.5: Trimaps gerados para ambos os cenários de teste usando os parâmetros da tabela 7.6, ajustada especificamente para o conjunto de imagens de entrada e fundo da figura 7.3.

mantêm ao serem aplicadas a vídeos. A maior diferença ao se utilizar vídeos é a amostragem, pela quantidade de quadros, e variância das amostras, pois as diversas formas de ruído, artefatos de compressão e interferência luminosa citadas anteriormente se aplicam. Ao utilizar o modelo de fundo desenvolvido, é possível atenuar estes efeitos até um certo ponto.

Na implementação desenvolvida atualmente, no entanto, a utilização de sequências longas de vídeo para o treinamento do fundo é proibitiva devido ao alto custo computacional envolvido. Este custo é ainda mais agravado em vídeos de resoluções altas. O maior fator limitante nesta questão é a capacidade de memória, pois o armazenamento simultâneo de todos os codebooks para um vídeo pode rapidamente ocupar mais de 4GB de memória, praticamente toda memória física da maioria dos computadores modernos, sendo necessária a utilização da memória virtual. Por este motivo, os testes realizados com vídeos foram mais suscintos, tentando focar em avaliar a aplicabilidade da solução desenvolvida.

Um parâmetro importante que até este momento não foi modificado nos testes é o KLCH referente ao cálculo das distâncias de cor na métrica E^*_{00}. Este parâmetro consiste de três valores que regulam a influência das componentes de luminância, crominância e matiz, respectivamente no cálculo da distância de cor. Para este trabalho, o mais importante é reduzir a influência de variações de iluminação na identificação de cores semelhantes, de modo que para isto é preciso



(a) (b) (c)

(d) (e)

Figura 7.6: Comparação de mapas de distâncias entre o quadro da imagem (d) e um modelo do fundo visto na imagem (e), sob diferentes parâmetros KLCH. a) KLCH = {0.8,1,1}. b) KLCH = {1,1,1}. c) KLCH = {0.8,1,1}.

que o valor do parâmetro L seja superior aos valores de C e H, segundo a equação 5.4.

A figura 7.6 apresenta uma comparação de mapas de distância para uma imagem calculados sob parâmetros de KLCH diferentes. Em primeiro lugar, observa-se que no frame do vídeo de frente da imagem 7.6d existe uma diferença de iluminação em relação ao do quadro de fundo 7.6e, o que é intencional, para observar o efeito destas variações. As imagens 7.6a, 7.6b e 7.6c são mapas de distâncias calculados sob três configurações diferentes: influência maior da iluminação, influência igual de todos os componentes e influência menor da iluminação. É possível notar que na medida que a influência da iluminação é diminuída, a imagem fica mais nítida e existe menor variação de distâncias em áreas semelhantes.

Devido ao excessivo tempo de se realizar testes em altas resoluções, diversos testes foram realizados com versões reduzidas de vídeos. A figura 7.7 apresenta uma sequência de quadros de um vídeo em resolução 240p, juntamente com seus mapas de distâncias e trimaps calculados com os parâmetros da tabela 7.7. Neste exemplo pode ser observado claramente o efeito de ruído e artefatos causados pela compressão e baixa amostragem. Este tipo de ruído, que afeta o trimap resultante, na realidade acaba tendo pouco efeito no resultado final após a aplicação do alpha matting, pois regiões desconhecidas isoladas acabam sendo descartadas. O maior problema nesse caso é a dificuldade de se obter uma transição suave mais apropriada através de uma região desconhecida.

Ao utilizar resoluções maiores, é possível se obter uma melhor precisão no resultado, mas além do custo computacional elevado outras dificuldades emergem. Os resultados de um teste

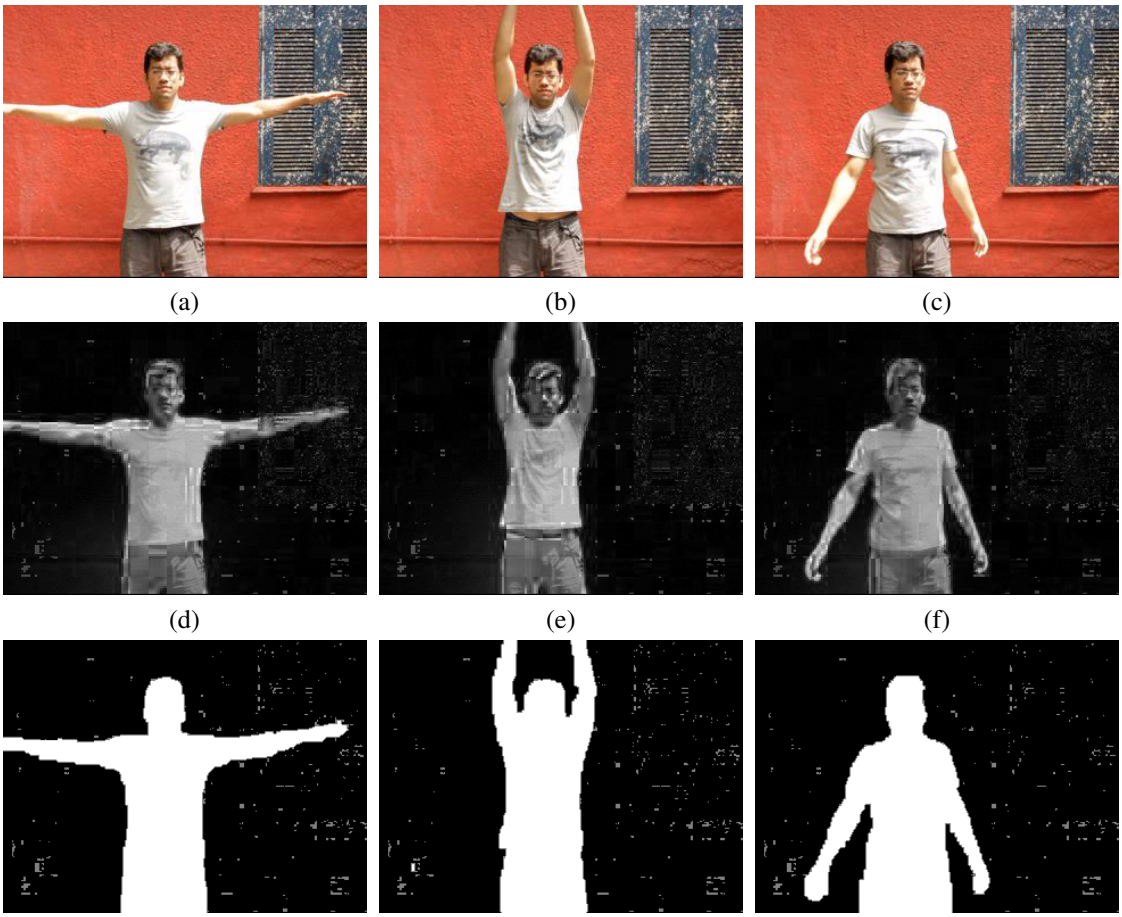


Figura 7.7: Exemplo de teste de vídeo realizado em resolução 240p. a-c) Quadros de entrada. d-f) Mapas de distância. g-i) Trimaps gerados.

Tabela 7.7: Parâmetros usados para os testes da imagem 7.7.

	KLCH		$D(p, l_p)$	$V_{pq}(l_p, l_q)$	l_f	l_d	l_b
L	1	l_f	$(1.4d_p)^2$	l_f	0	0.03	2
C	0.3	l_d	$0.05\bar{d}_p$	l_d	0.03	0	0.03
H	0.3	l_b	$((1-d_p)0.25)^2$	l_b	2	0.03	0

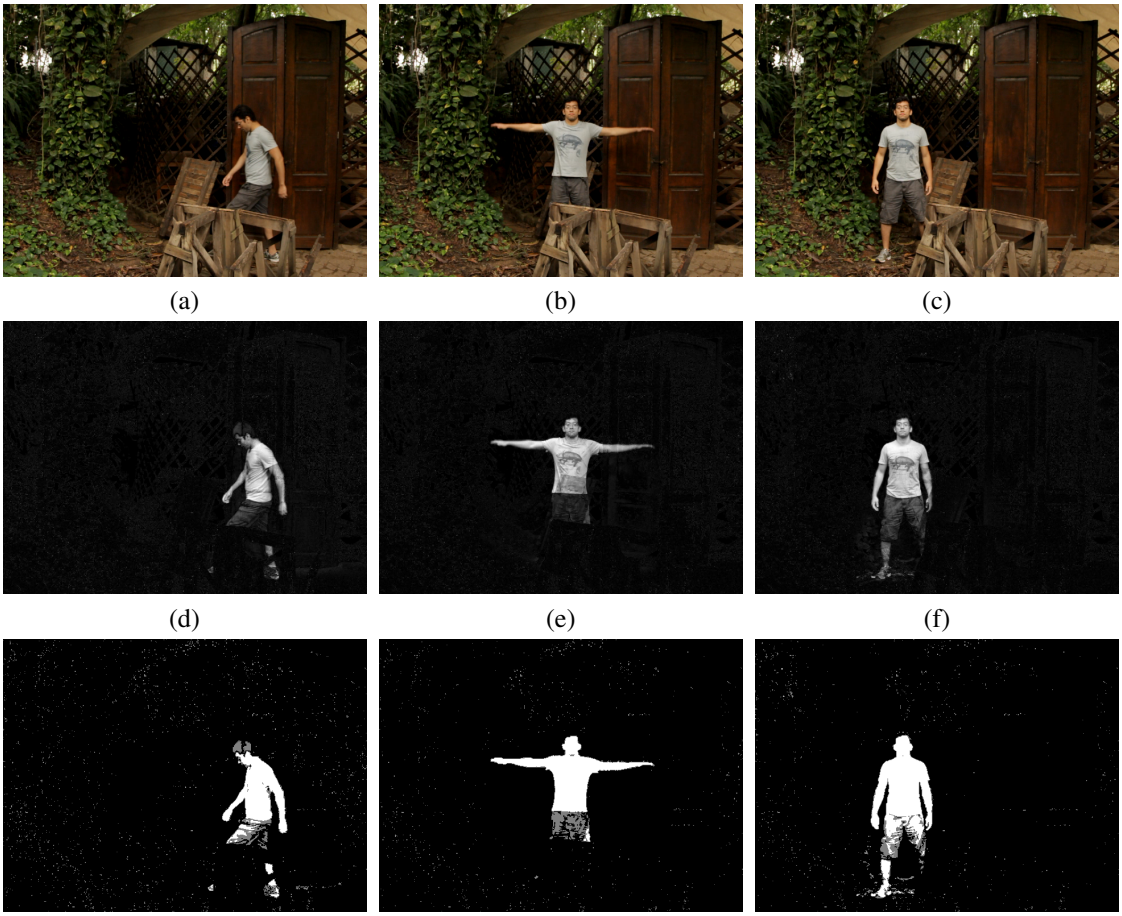


Figura 7.8: Exemplo de teste de vídeo com fatores adversos. a-c) Quadros de entrada. d-f) Mapas de distância. g-i) Trimaps gerados.

Tabela 7.8: Parâmetros usados para os testes da imagem 7.8.

	KLCH	$D(p, l_p)$		$V_{pq}(l_p, l_q)$	l_f	l_d	l_b
L	1	l_f	$(1.3d_p)^2$	l_f	0	0.001	2
C	0.42	l_d	$0.02\bar{d}_p$	l_d	0.001	0	0.001
H	0.42	l_b	$((1-d_p)0.185)^2$	l_b	2	0.001	0

de vídeo que ressalta algumas dessas dificuldades, além de problemas de se trabalhar com segmentação de vídeos discutidas no capítulo 2 pode ser visto na figura 7.8. Este teste foi realizado com um modelo do fundo treinado com 60 quadros em resolução 480p ao longo de um segundo e utilizando os parâmetros da tabela 7.8.

O primeiro problema a ser notado é a grande quantidade de distâncias médias encontradas em regiões que são claramente fundo, o que é atribuído à baixa amostragem de treinamento em relação à variância da cena. Mesmo em regiões da imagem onde o fundo é estático é possível observar variações nos três quadros. Por se tratar de uma cena com iluminação natural, influência de vento e fundo complexo, 60 frames ao longo de um segundo não são suficientes para modelar corretamente o fundo nessa resolução. Ademais, pode ser observado que a presença do ator modifica a distribuição de distâncias do fundo nas suas proximidades, devido à sombra e à reflexão de luz.

Na primeira coluna uma parte da perna do ator é obstruída por um objeto, o que faz com que seja necessário de se segmentar duas partes separadamente, algo que não é adequado para essa abordagem. Como o objeto obstruinte apresentou distâncias muito baixas neste caso, a segmentação apresentou resultados melhores do que o esperado para este caso, corretamente identificando as partes separadas. Nas colunas seguintes, mesmo com uma obstrução o ator ainda é considerado um objeto íntegro em termos de segmentação.

É possível verificar nos três quadros como certas tonalidades de cores do ator se mesclam com o fundo, especialmente suas roupas e seu cabelo. Sua bermuda, que é de um tom mais escuro em relação à sua camiseta acaba apresentando distâncias muito baixas das cores do fundo, fazendo com que o processo de minimização de energia concentre os rótulos de fundo apenas na sua parte superior. Na primeira e terceira coluna, no entanto, a presença de áreas de distâncias maiores pertencentes às pernas do ator faz com que sua bermuda seja mais facilmente classificada como frente. Na segunda coluna, onde suas pernas são completamente obstruídas a segmentação da bermuda é claramente inferior.

Alguns dos problemas citados podem ser resolvidos com a utilização de um fundo menos

(a) (b) (c)

Figura 7.9: Teste de vídeo sobre fundo simplificado. a) Quadro de entrada. b) Mapa de distâncias. c) Trimap gerado.

Tabela 7.9: Parâmetros usados para os testes das imagens 7.9 e 7.10.

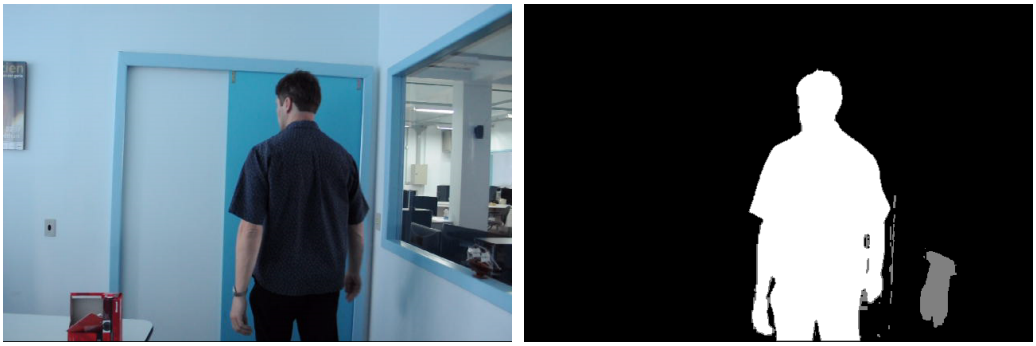
	KLCH		$D(p,l_p)$	$V_{pq}(l_p,l_q)$	l_f	l_d	l_b
L	1	l_f	$(1.25d_p)^2$	l_f	0	0.001	2
C	0.42	l_d	$0.05d_p$	l_d	0.001	0	0.001
H	0.42	l_b	$((1-d_p)0.35)^2$	l_b	2	0.001	0

complexo e que apresente tons distintos das vestimentas do ator. No teste apresentado na figura 7.9, que é uma versão em maior resolução da cena vista na figura 7.7 isto é demonstrado. Nesse caso, foi realizado um treinamento de apenas 30 quadros ao longo de um segundo e usando os parâmetros da tabela 7.9, o que foi suficiente para se modelar boa parte do fundo vermelho, mas não da textura da janela.

Três defeitos são evidentes nesse caso: a proximidade do ator da parede causa um padrão de interferência no fundo, aumentando as distâncias encontradas; a tonalidade vermelha do fundo se mescla com os lábios do ator; e a forte incidência de luz solar sobre a parte direita do rosto do ator cria uma complexa variação de tons que dificulta a identificações de regiões como semelhantes durante a minimização de energia. Todos esses defeitos são menos evidentes na versão de resolução menor da figura 7.9, especialmente em relação à confusão de cores da face do ator com o fundo.

Muitos dos defeitos encontrados nos mapas de distâncias, e especialmente nos trimaps resultantes poderiam ser atenuados com a utilização de operações de pré-processamento, como redução de ruído ou pós-processamento, como aberturas e fechamentos. Para isso ser possível, seria necessária a intervenção de um usuário que indique quais operações a serem executadas e seus parâmetros; ou que sejam feitas suposições sobre a natureza da cena do vídeo para que operações sejam executadas automaticamente.

A primeira alternativa é contrária à proposta desse trabalho, que visa a automação do processo de segmentaçào, reduzindo ao máximo o envolvimento de usuários. A segunda alterna-



(a) (b)

Figura 7.10: Teste de vídeo sob condições ideais. a) Quadro de entrada. b) Trimap gerado.

tiva agrega complexidade ao problema, pois implica na identificação de elementos adicionais no vídeo, o que não é uma tarefa trivial. Um exemplo disso seria a aplicação de operações de fechamento para a redução da presença de pequenas regiões desconhecidas espalhadas pelo trimap resultante e para remover buracos em faces. Uma cena onde um ator apresenta uma indumentária complexa, como um vestido de noiva que apresenta fendas, ou que está em uma pose irregular e na qual sejam executados fechamentos provavelmente irá apresentar resultados indesejados.

Na figura 7.10 um caso ideal é apresentado: o ator está de costas, evitando o efeito de reflexos na face; o ambiente possui iluminação interna que não incide diretamente sobre a cena e o fundo apresenta apenas um tom homogêneo. O vídeo de entrada utilizado possui a resolução 640x424 e foram usados os parâmetros da tabela 7.9 para realizar o cálculo de distâncias e minimização de energia. Pequenos erros são perceptíveis devido à sombra incidente sobre a parede, mas como estas regiões foram classificadas apenas como desconhecidas isso não afeta o processo de alpha matting.

Os espaços formados entre as pernas do ator e entre seus braços e o corpo são áreas difíceis de serem segmentados corretamente, pois existem mudanças muito bruscas entre frente/fundo, que podem causar defeitos no processo de minimização. Ademais, este caso é um bom exemplo onde a aplicação de uma operação de pós-processamento como fechamento realizada com pouco critério poderia ser prejudicial. De todo modo, o resultado obtido neste exemplo é bastante satisfatório, segmentando o ator com bastante precisão.

7.4 Aplicação de Alpha Matting

Após a obtenção dos trimaps, como demonstrado na seção anterior, é possível fazer a aplicação de uma técnica de alpha matting para realizar a extração dos elementos e o seu cálculo

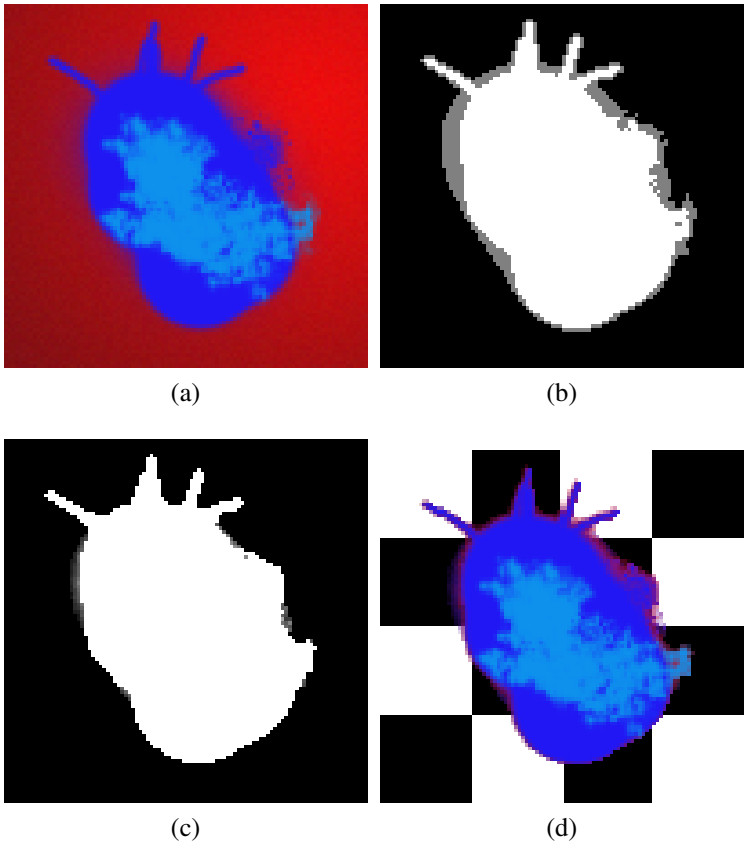


Figura 7.11: Aplicação de alpha matting nos testes de imagens de baixa resolução. a) Imagem original. b) Trimap gerado. c) Alpha matte estimado. d) Imagem extraída.

de transparência. Para todos os testes de alpha matting realizados nesse trabalho foi utilizada a técnica de Shared Matting (GASTAL; OLIVEIRA, 2010).

Nas figuras 7.11 e 7.12 são apresentados os resultados da aplicação de alpha matting sobre os trimaps gerados para as imagens sintéticas da seção 7.1. O mapa de transparência estimado para a imagem de baixa resolução possui uma precisão bastante reduzida, o que se deve ao fato da técnica de alpha matting utilizada ser baseada em amostragem. Deste modo, o resultado obtido com a imagem sintética de alta resolução, onde existe uma quantidade muito maior de píxeis para se fazer uma estimativa, é de boa qualidade.

O cálculo de alpha matting para os trimaps gerados no teste de vídeo da figura 7.7 são apresentados na figura 7.13. A resolução do vídeo nesse caso é 240p, de modo que a estimativa do alpha matte é um pouco prejudicada pela baixa amostragem. Como pode ser visto no caso da coluna central, a presença de artefatos de compressão no trimap pode ser passada para a extração final do plano de frente. Ademais, como o contorno mais externo do ator neste caso é quase completamente opaco, seu alpha matte parece ser quase completamente branco, mas na realidade são valores altos diferentes em escala de cinza.

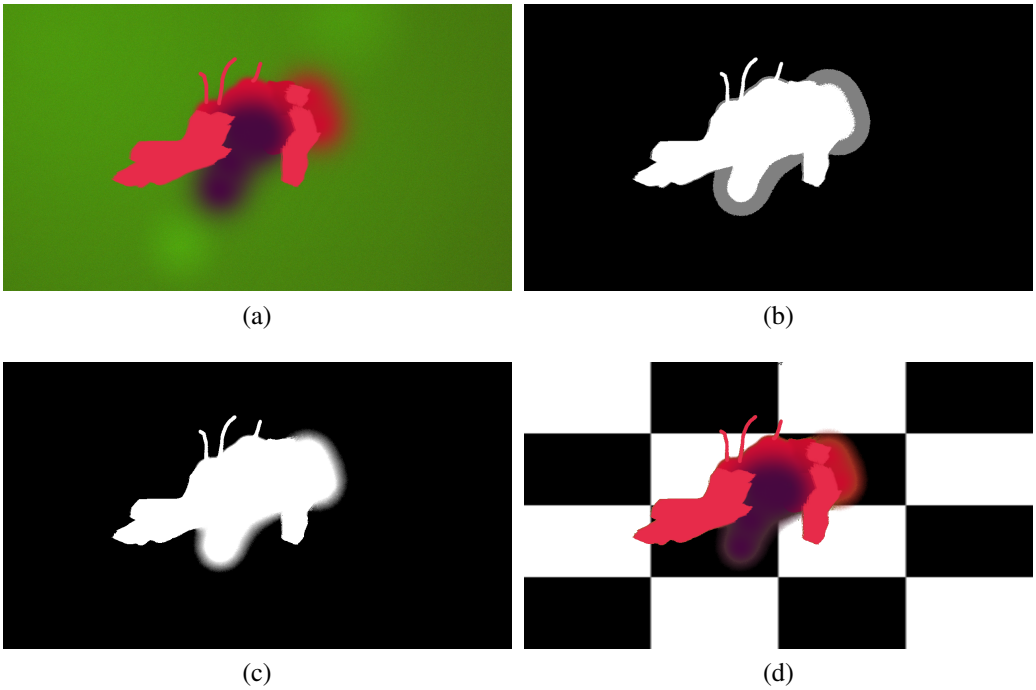


Figura 7.12: Aplicação de alpha matting nos testes de imagens de alta resolução. a) Imagem original. b) Trimap gerado. c) Alpha matte estimado. d) Imagem extraída.

Nos casos onde o trimap gerado foi inadequado a baixa qualidade dos resultados finais é evidente, como demonstram as figuras 7.14 e 7.15. No primeiro caso, representando o teste realizado sob condições adversas, o tom escuro do fundo fez com que o cabelo do ator fosse excluído do resultado final, bem como boa parte de sua bermuda. No segundo caso, os danos maiores foram em relação à baixa qualidade do trimap na região da sua face, sendo que a interferência luminosa na parede de fundo prejudicou a extração de sua bermuda. Como estes vídeos foram gravados em uma resolução maior (420p), mesmo apresentado resultados ruins é possível perceber uma precisão maior em contornos e delimitações das partes extraídas.

Finalmente, os resultados da aplicação de alpha matting sobre o teste realizado sob condições ideais são apresentados na figura 7.16. Neste caso podem ser vistas falhas no espaço entre os braços e o corpo do ator, devido às dificuldades de segmentação já discutidas. Também podem ser percebidos no alpha matte da imagem 7.16c alguns artefatos do processo de alpha matting. Essa ocorrência provavelmente pode ser justificada pela presença de regiões desconhecidas muito estreitas entre áreas de frente e fundo.

7.5 Tempos de Execução

Os testes apresentados nesse capítulo foram realizados sob várias configurações em diversos computadores diferentes. Por esse motivo, é especialmente difícil de se fazer uma generaliza-

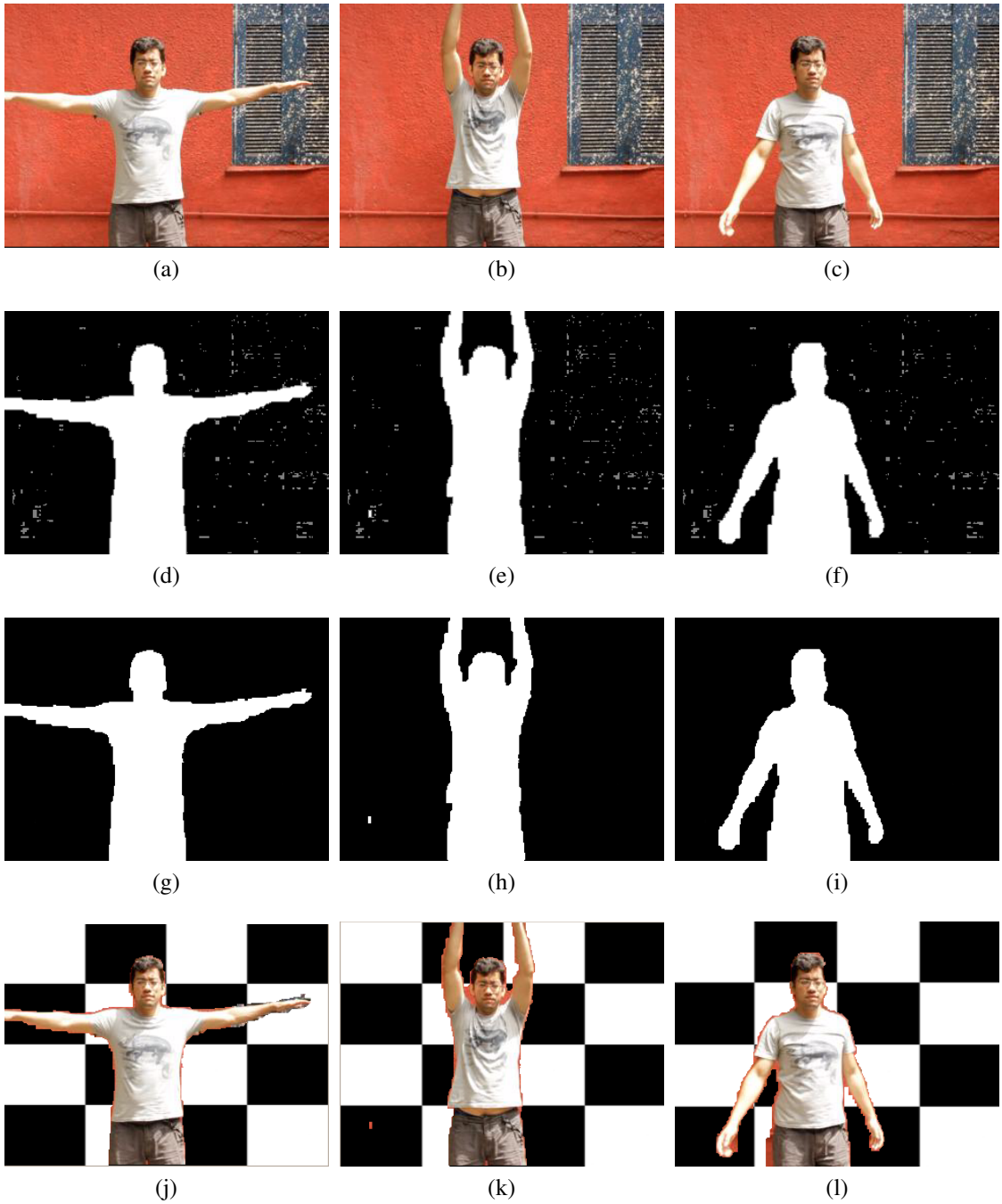
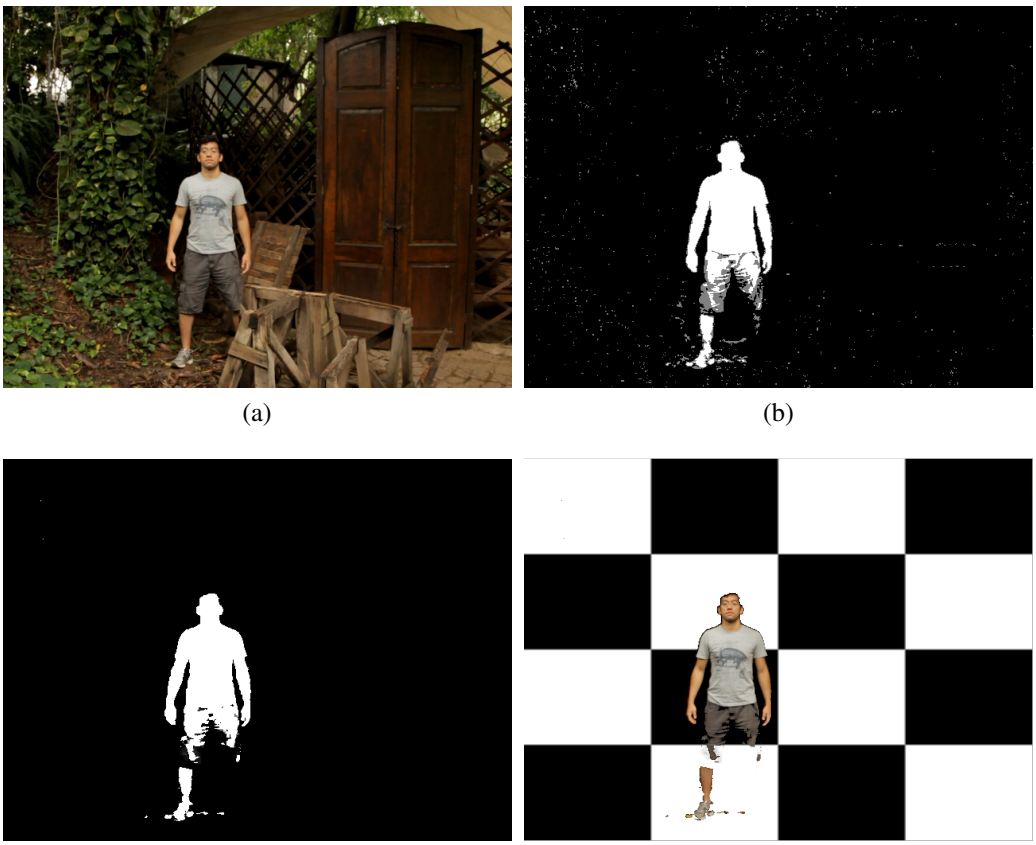


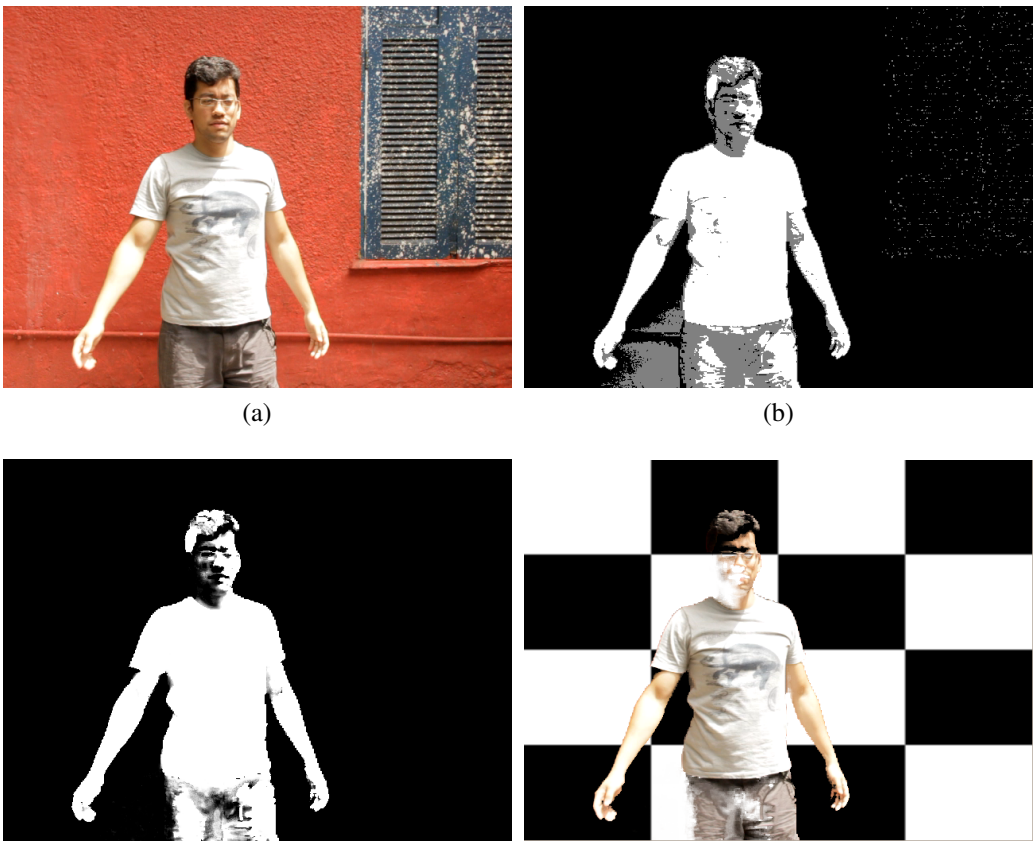
Figura 7.13: Aplicação de alpha matting no teste de vídeo realizado em resolução 240p. a-c) Quadros de entrada. d-f) Trimaps gerados. g-i) Alpha mattes estimados. j-l) Imagens extraídas.



(a) (b)

(c) (d)

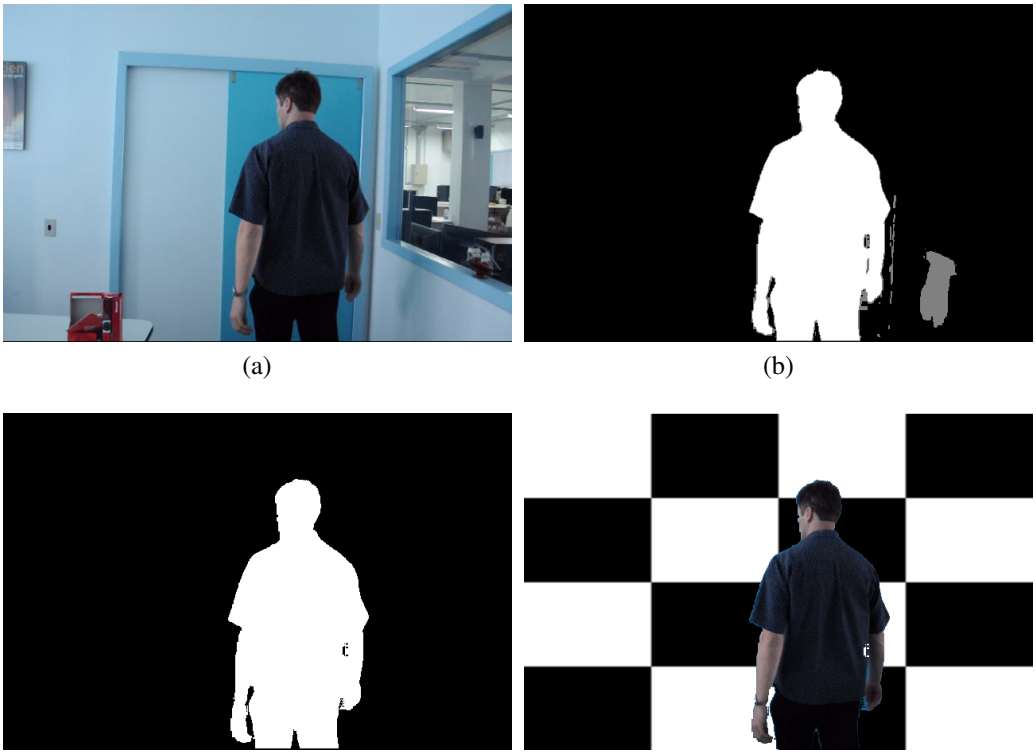
Figura 7.14: Aplicação de alpha matting no caso de teste sob condições adversas. a) Imagem original. b) Trimap gerado. c) Alpha matte estimado. d) Imagem extraída.



(a) (b)

(c) (d)

Figura 7.15: Aplicação de alpha matting no teste da imagem 7.9. a) Imagem original. b) Trimap gerado. c) Alpha matte estimado. d) Imagem extraída.



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

Figura 7.16: Aplicação de alpha matting no teste da imagem 7.9. a) Imagem original. b) Trimap gerado. c) Alpha matte estimado. d) Imagem extraída.

ção dos tempos de execução da solução desenvolvida. Nesta seção, são apresentadas relações entre diversos parâmetros dos testes e tempos de execução com a finalidade de delimitar uma estimativa do real custo computacional envolvido nas técnicas implementadas.

7.5.1 Testes com Imagens

Todos os testes com imagens presentes nesse capítulo foram executados em um notebook HP Pavillion com um processador Intel Core 2 Quad Q9000 2.0 Ghz 64 bits e 4GB de memória RAM, utilizando o software Matlab 2011b (MATLAB, 2011). O modelo implementado para imagens utiliza dois passos de execução, como visto no capítulo 6: primeiramente são passados como entrada uma imagem de frente e um conjunto de imagens de fundo e, gerando um mapa de distâncias e em seguida a minimização de energia é realizada para gerar o trimap. Deste modo, o tempo de cada um dos passos é avaliado separadamente.

A tabela 7.10 apresenta tempos de execução de imagens de baixa resolução (100x100), como vistos nos testes das tabelas 7.3 e 7.4. Nesta tabela está expressa a relação entre imagens usadas para o conjunto de treinamento e as duas etapas de execução. A etapa de cálculo do mapa de distâncias nesse caso também engloba a geração dos codebooks de treinamento, motivo pela

Tabela 7.10: Tabela de tempos de execução de testes em imagens de resolução 100x100.

Imagens/Etapa	Cálculo do mapa de distâncias	Minimização de energia
1	7.41s	0.28s
2	9.53s	0.26s
3	12.08s	0.31s
4	15.11s	0.32s

Tabela 7.11: Tabela de tempos de execução de testes em imagens de resolução 1280x720.

Imagens/Etapa	Cálculo do mapa de distâncias	Minimização de energia
1	7.2m	5.49s
2	14.56m	6.01s
3	19.83m	6.05s

qual é possível ver um crescimento de tempo notável com a adição de imagens de treinamento. O processo de minimização de energia é praticamente inafetado pela quantidade de imagens usadas, de modo que seu tempo é aproximadamente constante em todos casos.

Para imagens de resolução altas, como as da figura 7.2 (1280x720p), existe um aumento significativo no tempo de execução, de modo que foram realizados testes somente com até 3 imagens de treinamento de fundo. Os resultados podem ser vistos na tabela 7.11, onde novamente é presente um grande crescimento de tempo com o aumento do conjunto de treinamento, enquanto o tempo da minimização de energia permanece praticamente inalterado.

7.5.2 Testes com Vídeos

O processo executado para se obter o trimap de um vídeo apresenta um passo a mais, pois o modelo do fundo é realizado em separado ao cálculo de distâncias, com o objetivo de economizar tempo entre execuções diferentes sobre o mesmo fundo. Deste modo, após o tempo inicial de treinamento este passo pode ser ignorado. Os nomes das etapas estão reduzidos nas tabelas para melhor disposição dos dados, sendo que o termo quadros corresponde à quantidade de quadros usado para o treinamento do fundo e os tempos apresentados para o cálculo de distâncias e geração do trimap correspondem ao tempo de processamento de um quadro do vídeo.

Os tempos apresentados nessa seção foram obtidos ao executar a solução em um computador Intel Core 2 Quad Q6600 2.4 Ghz 64 bits, com 4 GB de memória RAM no software Matlab 2011b para o caso de resoluções baixas e em um computador Intel Xeon ES-2620 2.0 Ghz 64 bits, com 16 GB de memória RAM e o software Matlab 2012b para o caso de resoluções altas.

As tabelas 7.12 e 7.13 apresentam os tempos de execução de testes para vídeos em resoluções 240p e 480p, respectivamente. Estes dados demonstram claramente o alto custo com-

Tabela 7.12: Tabela de tempos de execução de testes em vídeos de resolução 320x240.

Quadros/Etapa	Codebooks	Distâncias	Trimap
1	52.39s	38.15s	2.11s
10	23.42m	1.69m	2.09s
30	1.93h	2.55m	2.28s
60	7.66h	3.80m	2.04s

Tabela 7.13: Tabela de tempos de execução de testes em vídeos de resolução 640x480.

Quadros/Etapa	Codebooks	Distâncias	Trimap
1	1.83m	1.32m	4.83s
10	1.51h	3.72m	4.49s
30	8.76h	8.18m	5.12s
60	21.95h	9.66m	5.33s

putacional envolvido na solução implementada, especialmente no processo de treinamento do fundo. De uma forma geral, o crescimento de complexidade é linear em relação à quantidade de píxeis e quadrático em relação ao número de quadros usados para o treinamento. Isso ocorre pois quanto mais quadros são usados para o treinamento, mais codebooks são criados por píxel, e mais testes de semelhança de cor precisam ser realizados. Espera-se que esse crescimento seja reduzido, atingindo um limiar quando os codebooks existentes modelarem completamente o fundo e codebooks novos não precisarem ser criados. O cálculo de distâncias também é afetado pelo aumento do número de quadros de treinamento, mas novamente a minimização de energia é sensível apenas à mudanças na resolução.

Dois fatores importantes devem ser comentados: primeiramente, os processadores utilizados apresentam uma grande quantidade de núcleos. Na implementação atual utilizando Matlab apenas um núcleo foi utilizado em todos os casos de teste, de modo que praticamente só a frequência dos núcleos pode ser considerada. Em segundo lugar, a execução do treinamento de vídeos em 240p aproximadamente a partir de 30 quadros e em 480p aproximadamente a partir de 10 quadros consumiam uma quantidade de memória superior à 4GB, forçando o sistema a utilizar a memória virtual e diminuindo drasticamente o desempenho.

8 CONCLUSÃO

Este trabalho teve como objetivo principal a concepção de uma solução para segmentação de vídeos, para ser utilizada em aplicações de storytelling interativo baseado em vídeo. Para este fim, foi feita uma ampla pesquisa nas áreas de segmentação de vídeos e imagens, alpha matting e espaços de cores. Dentre as características buscadas, foi dada especial importância à automatização do processo. A abordagem resultante visa a geração de trimaps para serem utilizados em alpha matting, através de um processo de minimização de energia, baseado na distância de cores entre um vídeo de entrada e um modelo conhecido do fundo.

Os resultados obtidos com imagens e vídeos demonstram o potencial da solução desenvolvida, apesar desta necessitar de ajustes. De uma forma geral, na maioria dos casos de testes com vídeos os trimaps gerados são capazes de extrair um objeto ou ator de seu fundo utilizando alpha matting, mas ainda não são ideais para aplicações de storytelling interativo baseado em vídeo. Para tal, primeiramente são necessários mais testes com resoluções altas e com configurações de cena e atores diferentes.

A maior dificuldade encontrada neste trabalho foi a grande variabilidade de dados que precisa ser considerada ao se trabalhar com vídeos. Como a automação do processo é um dos objetivos principais do trabalho, elementos diferentes como: ruído, sombras, variações na iluminação e movimentos no fundo entre outros precisam ser tratados automaticamente - e simultaneamente -, o que não é uma tarefa trivial.

Como comentado anteriormente, alguns resultados poderiam se beneficiar muito com a aplicação de técnicas de pré e pós processamento, mas isto iria requerer que se fizesse suposições sobre a natureza do vídeo e que em última instância um usuário realizasse entrada de dados adicionais. Por este motivo, decidiu-se limitar o controle externo apenas a variações de parâmetros, fazendo com que a solução apresente resultados mais brutos, porém autênticos.

Outras dificuldades encontradas no trabalho foram o alto custo computacional e de memória

da aplicação implementada em Matlab e a dificuldade em se obter bons vídeos para testes. A gravação de vídeos que se enquadrassem no escopo da proposta foi problemática, pois o Laboratório de Computação Aplicada, onde este trabalho foi realizado, não possui instalações e equipamentos adequados. Estes problemas tiveram um impacto principalmente na finalização do trabalho, causando uma certa demora no processo de testes.

8.1 Contribuições

O estado atual da solução desenvolvida apresentou diversos resultados interessantes, tendo como contribuição principal uma abordagem para a geração automática de trimaps para alpha matting. Esta contribuição é importante pois atualmente trimaps são a principal forma de entrada utilizada para alpha matting, no entanto, sua criação é considerada um processo manual e pouco tratado na literatura. Nesse trabalho não só são avaliadas diversas questões relacionadas à geração de trimaps como é apresentada uma forma de gerar trimaps automaticamente para vídeos atendendo um certo escopo de requisitos. Ademais, também podem ser listadas como contribuições desse trabalho:

- **A concepção de um processo de segmentação de vídeos para storytelling interativo baseado em vídeo**: foi proposto nesse trabalho um processo de segmentação para vídeos - baseado na utilização de alpha matting - para solucionar o problema principal de storytelling digital baseado em vídeo, que era a extração de atores de filmagens para serem compostos em novas cenas;

- **Um modelo de fundo para cálculo de distâncias de cor baseado em codebooks**: para este trabalho foi adaptado um método de segmentação frente/fundo existente para poder realizar cálculos de distância de cor ao invés de segmentações binárias;

- **A utilização de distâncias perceptuais de cor para subtração de fundo**: a combinação de um modelo de fundo baseado em codebooks com a métrica de cores E^*_{00} permitiu uma melhor estimativa de distâncias entre cores, especialmente podendo ajustar a influência de componentes como luminância; e

- **A modelagem do problema de geração de trimaps como um problema de minimização de energia**: neste trabalho as características desejadas para um trimap foram traduzidas para um contexto de problemas de minimização de energia através da estimativa de custos baseados em distâncias de cor.

8.2 Limitações e Trabalhos Futuros

Como discutido nesse capítulo, esse trabalho apresenta resultados satisfatórios, mas não constitui uma solução ideal para os problemas propostos, servindo primariamente como uma pesquisa inicial e delimitando uma arquitetura geral. As principais limitações encontradas atualmente são:

- O alto custo computacional da solução implementada;
- A sensibilidade a variações na câmera, como trepidações e mudanças de foco, especialmente em relação ao vídeo de treinamento;
- A sensibilidade a condições de filmagem inadequadas, como pouca ou muita iluminação e sombras;
- A sensibilidade a cores de frente semelhante ao fundo, como roupas do ator que se mesclam com o fundo em certas posições; e
- A grande influência dos parâmetros no resultado final.

Deste modo, existem alguns aspectos a serem melhorados em trabalhos futuros. Esta seção apresenta alguns pontos relevantes e promissores para a continuidade da pesquisa e desenvolvimento.

8.2.1 Implementação com paralelismo

No capítulo 6 é justificado o motivo para a implementação na plataforma MATLAB bem como suas limitações, sendo o desempenho uma das mais críticas. No entanto, a natureza dos métodos para treinamento de fundo e cálculo de distâncias, especialmente, permite um alto grau de paralelismo, de modo que estas poderiam ser facilmente adaptadas para uma arquitetura paralela como GPUs ou clusters. Como atualmente este é o maior gargalo da implementação, espera-se que o desempenho melhore drasticamente realizando estas mudanças.

Dentre os trabalhos futuros sugeridos, este é provavelmente o que pode ser desenvolvido com maior facilidade, por se tratar em grande parte de um trabalho de implementação e programação. Caso este seja realizado primeiramente e a arquitetura seja portada para outra plataforma diferente do MATLAB, apesar do ganho em tempo para realizar testes é possível que se torne mais difícil o processo de desenvolver e prototipar novas técnicas, pois arquiteturas paralelas são menos práticas e flexíveis.

8.2.2 Classes de minimização de energia adaptáveis

Nesse trabalho a abordagem de minimização desenvolvida buscou criar uma relação estrita entre as classes de rótulos e as regiões de frente, fundo e desconhecidas de um trimap. Isso trouxe muitas dificuldades pois em alguns casos havia uma grande quantidade de variações e elementos diferentes em vídeo, que precisavam ser finamente delineados em apenas três classes com poucos parâmetros. Alguns exemplos disso são quando um ator veste uma roupa altamente texturizada ou com divisões de cores distintas e até sombras sobre superfícies. Nesses casos, existe uma grande variação de tipos de regiões de distância, como áreas pequenas com grandes distâncias que deveriam ser pertencentes ao fundo ou áreas grandes com baixas distâncias que deveriam ser pertencentes à frente.

Uma das formas de solucionar este problema é modelar o processo de minimização de energia para enquadrar classes de rótulos adicionais, que posteriormente seriam transformados em um trimap. Ao analisar um mapa de alturas, que é normalizado relativo às amostras encontradas em um espaço de tons de cinza, é possível identificar distribuições particulares, sendo as modas particularmente importantes, pois indicariam possíveis classes de rótulos adicionais. Essas classes poderiam corresponder a sombras, a uma cor diferente na roupa de um ator ou alguma espécie de ruído no fundo.

Ajustar classes de minimização de energia de acordo com a distribuição estatística das distâncias permitiria um processo de minimização mas preciso, além de facilitar a identificação de píxeis transparentes, pois regiões desconhecidas seriam minimizadas localmente em relação às classes vizinhas. Posteriormente, essas classes seriam convertidas em seu equivalente em um trimap: frente, fundo ou desconhecido. O rótulo de sombras simplesmente seria substituído pela cor preta, represententando seu enquadramento no fundo e duas classes diferentes: uma para a calça do ator outra para a camisa, seriam substituídas por branco, por exemplo.

8.2.3 Múltiplas etapas de minimização de energia

Outra solução possível para o problema descrito na subseção anterior seria a realização de múltiplos processos de minimização de energia sob parâmetros diferentes, ao invés de classes diferentes em uma só aplicação da minimização. A idéia é basicamente a mesma, isolar certos elementos presentes em um quadro que, apesar de realmente pertencerem à frente, fundo ou região desconhecida, são distintos o suficiente para dificultar a minimização de energia.

Neste abordagem, cada aplicação de minimização de energia seria realizada com parâme-

tros diferentes, gerando diversas imagens com rotulagens diferentes. O resultado final, então, poderia ser estimado utilizando operações de processamento para formar um trimap no formato branco, preto e cinza. O número de classes utilizado em cada etapa pode ser variado, podendo inclusive haver uma intersecção entre as classes adaptáveis da subseção anterior, dependendo do objetivo. Realizar diversas minimizações de energia binárias em série, no entanto, parece ser a forma mais simples e eficiente em termos de custo computacional de se implementar essa adaptação.

8.2.4 Ajuste automático de parâmetros

Como demonstrado ao longo desse trabalho, os parâmetros utilizados tanto para o cálculo de distâncias quanto para a minimização de energia afetam drasticamente o resultado obtido. Encontrar um conjunto de parâmetros ideal para cada caso pode ser uma tarefa trabalhosa e tediosa, de modo que formas automáticas de estimativa de parâmetros são necessárias.

No capítulo 7 foram usados alguns parâmetros variáveis, como a média das distâncias, por exemplo. O ajuste automático de parâmetros seria o próximo passo nesse sentido, analizando estatisticamente as distâncias ou até algum aspecto do vídeo de entrada original para determinar os melhores valores de forma automática. Caso não seja encontrada uma relação matemática direta que modele as configurações de parâmetros, existe ainda a possibilidade de se utilizar redes neurais ou Support Vector Machines como ajustadores de função.

8.2.5 Mapa de custos de mudança de rótulos adequado

O termo de suavização $V_{pq}(l_p, l_q)$ usado no processo de minimização de energia leva em consideração não apenas o custo estático de variação entre classes, mas também um mapa da imagem que indica o custo específico de haver mudanças naquele ponto. O mapa utilizado atualmente é apenas uma aplicação de uma operação de laplaceanos de gaussianos sobre o mapa de distâncias, como é comum em muitas técnicas de minimização de energia envolvendo imagens. Essa abordagem faz com que se tenha uma maior facilidade de mudanças de rótulos próximo à bordas na imagem. É possível, no entanto, estimar um mapa mais adequado que leve em consideração outros aspectos da imagem.

8.2.6 Modelos diferentes de cor e fundo

Para calcular distâncias de cor em relação a um fundo conhecido, o método desenvolvido utilizando codebooks no espaço de cor CIELAB se provou muito superior ao utilizando RGB.

No entanto, ainda existem alguns pontos que podem ser explorados, especialmente devido ao baixo desempenho dessa abordagem.

Atualmente, o espaço modelado por um codebook possui muito pouca informação espacial, praticamente representando apenas um ponto no espaço de cor. Uma possível melhora seria utilizar um modelo estatístico baseado em gaussianas tridimensionais, onde os parâmetros são ajustados de acordo com os pontos que são selecionados como pertencente a ela. A intensidade dessa função gaussiana seria utilizada para ajustar a distância real da cor de um ponto em relação ao codebook.

Outra possibilidade é a utilização de outros modelos de cor que aprentem métricas interessantes de cálculo de similaridade de cor. Um exemplo é o espaço de cor desenvolvido por Chong (CHONG; GORTLER; ZICKLER, 2008), que ajusta um espaço tridimensional euclidiano para melhor descrever distâncias de cor ignorando iluminação. Neste trabalho, o espaço CIEXYZ é adaptado através de transformações de modo que a distância euclidiana entre dois pontos corresponde à sua distância de cor perceptual. A qualidade dos resultados de distância obtida é inferior ao apresentada pela métrica E^*_{00}, mas muito mais eficiente e prático.

8.3 Considerações Finais

Este trabalho foi realizado em conjunto com o laboratório Vision Lab da Pontifícia Universidade Católica do Rio de Janeiro e foi financiado pela CAPES através do projeto RH-TVD # 133/2008.

As implementações e testes desenvolvidos foram realizadas com o auxílio dos colegas Leonardo C. Campagnolo e Guilherme G. Schardong.

O código-fonte do Shared Matting (GASTAL; OLIVEIRA, 2010) desenvolvido por Eduardo S. L. Gastal foi cordialmente cedido pelo prof. Manuel M. Oliveira da Universidade Federal do Rio Grande do Sul.

REFERÊNCIAS

AGARWALA, A. Keyframe-based tracking for rotoscoping and animation. In: ACM SIGGRAPH, 2004. **Proceedings...** [S.l.: s.n.], 2004.

ALPHA Matting Evaluation Website. www.alphamatting.com.

BAGON, S. Matlab Wrapper for Graph Cuts. **www.wisdom.weizmann.ac.il/ bagon**, [S.l.], 2006.

BAI, X.; SAPIRO, G. Geodesic Matting: a framework for fast interactive image and video segmentation and matting. **Int. J. Comput. Vision**, Hingham, MA, USA, v.82, n.2, p.113–132, Apr. 2009.

BLACK, M. J.; ANANDAN, P. The robust estimation of multiple motions: parametric and piecewise-smooth flow fields. **Computer Vision and Image Understanding**, [S.l.], v.63, p.75–104, 1996.

BOBICK, A.; DAVIS, J. An appearance-based representation of action. In: IEEE INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION, 1996. **Anais...** [S.l.: s.n.], 1996.

BOYKOV, Y.; FUNKA-LEA, G. Graph cuts and efficient n-d image segmentation. **International Journal of Computer Vision**, [S.l.], v.70, p.109–131, 2006.

BOYKOV, Y.; KOLMOGOROV, V. An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, [S.l.], v.26, n.9, p.1124 –1137, sept. 2004.

BOYKOV, Y.; VEKSLER, O.; ZABIH, R. Fast Approximate Energy Minimization via Graph Cuts. **IEEE Trans. Pattern Anal. Mach. Intell.**, Washington, DC, USA, v.23, n.11, p.1222–1239, Nov. 2001.

BRADSKI, G. The OpenCV Library. **Dr. Dobb's Journal of Software Tools**, [S.l.], 2000.

C. JULIà A. SAPPA, F. L. J. S.; LOPEZ, A. Motion segmentation from feature trajectories with missing data. In: IBERIAN CONFERENCE ON PATTERN RECOGNITION AND IMAGE ANALYSIS, 2007. **Anais...** [S.l.: s.n.], 2007.

CAVAZZA, M.; CHARLES, F.; MEAD, S. J. Character-Based Interactive Storytelling. **IEEE Intelligent Systems**, Piscataway, NJ, USA, v.17, n.4, p.17–24, July 2002.

CHONG, H. Y.; GORTLER, S. J.; ZICKLER, T. A perception-based color space for illumination-invariant image processing. **ACM Trans. Graph.**, New York, NY, USA, v.27, n.3, p.61:1–61:7, Aug. 2008.

CHRISTOPH RHEMANN CARSTEN ROTHER, J. W. M. G. P. K. P. R. A Perceptually Motivated Online Benchmark for Image Matting. In: CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION (CVPR), 2009. **Anais...** [S.l.: s.n.], 2009.

CHUANG, Y. Video matting. In: ACM SIGGRAPH, 2002. **Proceedings...** [S.l.: s.n.], 2002.

ELGAMMAL A HARWOOD D, D. L. **Non-parametric model for background subtraction**. 2000.

ENERGY MINIMIZATION METHODS IN COMPUTER VISION AND PATTERN RECOGNITION, 7TH INTERNATIONAL CONFERENCE, EMMCVPR 2009, BONN, GERMANY, AUGUST 24-27, 2009. PROCEEDINGS, 2009. **Anais...** Springer, 2009. (Lecture Notes in Computer Science, v.5681).

FU, K. S.; MUI, J. K. A survey on image segmentation. **Pattern Recognition**, [S.l.], v.13, p.3–16, 1981.

GASTAL, E. S. L.; OLIVEIRA, M. M. Shared Sampling for Real-Time Alpha Matting. **Computer Graphics Forum**, [S.l.], v.29, n.2, p.575–584, May 2010. Proceedings of Eurographics.

GETREUER, P. **Colorspace Transformations**. 2011.

GOH, A.; VIDAL, R. Segmenting motions of different types by unsupervised manifold clustering. In: IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION, 2007. **Anais...** [S.l.: s.n.], 2007.

H. SHEN L. ZHANG, B. H.; LI, P. A map approach for joint motion estimation, segmentation, and super resolution. **IEEE Transactions on Image Processing,**, [S.l.], 2007.

HARALICK, R. M.; SHAPIRO, L. G. Image segmentation techniques. **Computer Vision, Graphics, and Image Processing**, [S.l.], v.29, n.1, p.100–132, Jan. 1985.

HORPRASERT, T. **A statistical approach for real-time robust background subtraction and shadow detection**. 1999.

J. WANG P. BHAT, A. C. M. A.; COHEN, M. Interactive video cutout. In: ACM SIGGRAPH, 2005. **Proceedings...** [S.l.: s.n.], 2005.

J. ZHANG F. SHI, J. W.; LIU, Y. 3d motion segmentation from straight-line optical flow. **Multimedia Content Analysis and Mining**, [S.l.], 2007.

JAVED O SHAFIQUE K, S. M. **A hierarchical approach to robust background subtraction using color and gradient information**. 2002.

KIM, K.; CHALIDABHONGSE, T. H.; HARWOOD, D.; DAVIS, L. Real-time foreground-background segmentation using codebook model. **Real-Time Imaging**, London, UK, UK, v.11, n.3, p.172–185, June 2005.

KOLMOGOROV, V.; ZABIH, R. What Energy Functions can be Minimized via Graph Cuts? In: EUROPEAN CONFERENCE ON COMPUTER VISION, 2002. **Anais...** [S.l.: s.n.], 2002. v.26(2), p.147–159.

KOPRINSKA, I.; CARRATO, S. Temporal video segmentation: a survey. **Signal Processing: Image Communication**, [S.l.], v.16, n.5, p.477 – 500, 2001.

LEE DS HULL JJ, E. B. **A Bayesian framework for Gaussian mixture background modeling.** 2003.

LEVIN, A.; RAV ACHA, A.; LISCHINSKI, D. Spectral Matting. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, [S.l.], v.30, n.10, p.1699 –1712, oct. 2008.

LI, Y. S. Y.; SHUM, H. Y. Video object cut and paste. In: ACM SIGGRAPH, 2005. **Proceedings...** [S.l.: s.n.], 2005.

LIMA, E. de; FEIJO, B.; FURTADO, A.; POZZER, C.; CIARLINI, A. Automatic Video Editing for Video-Based Interactive Storytelling. In: MULTIMEDIA AND EXPO (ICME), 2012 IEEE INTERNATIONAL CONFERENCE ON, 2012. **Anais...** [S.l.: s.n.], 2012. p.806 –811.

LIMA, E. E. S.; POZZER, C. T.; DALLA FAVERA, E. C.; d'ORNELLAS, M. C.; CIARLINI, A.; FEIJÓ B.AND FURTADO, A. L. Support Vector Machines for Cinematography Real-Time Camera Control in Storytelling Environments. In: VIII BRAZILIAN SYMPOSIUM ON COMPUTER GAMES AND DIGITAL ENTERTAINMENT - SBGAMES 2009, 2009. **Anais...** [S.l.: s.n.], 2009.

LINDEBERG, T.; EKLUNDH, J. Scale detection and region extraction from a scale-space primal sketch. In: COMPUTER VISION, 1990. PROCEEDINGS, THIRD INTERNATIONAL CONFERENCE ON, 1990. **Anais...** [S.l.: s.n.], 1990. p.416 –426.

LOMBAERT, H. **Energy minimization with Graph Cuts**. 2006.

M. KONG J.-P. LEDUC, B. G.; WICKERHAUSER, V. Spatio-temporal continuous wavelet transforms for motion-based segmentation in real image sequences. In: INTERNATIONAL CONFERENCE ON IMAGE PROCESSING, 1998. **Proceedings...** [S.l.: s.n.], 1998.

MACADAM, D. L. Visual Sensitivities to Color Differences in Daylight. **J. Opt. Soc. Am.**, [S.l.], v.32, n.5, p.247–273, May 1942.

MATEAS, M.; STERN, A. Structuring content in the Façade interactive drama architecture. In: FIRST ARTIFICIAL INTELLIGENCE AND INTERACTIVE DIGITAL ENTERTAINMENT CONFERENCE, 2005. **Proceedings...** [S.l.: s.n.], 2005. p.93–98.

MATLAB. **version 7.13.0.564 (R2011b)**. Natick, Massachusetts: The MathWorks Inc., 2011.

MITTAL A, P. N. **Motion-based background subtraction using adaptive kernel density estimation.** 2004.

ONG E. P., L. W. T. B. J. . E. M. **"Fast Automatic Video Object Segmentation for Content-Based Applications."Advances in Image and Video Segmentation**. [S.l.]: IGI Global, 2006. p.140–160.

PORIKLI F, T. O. **Human bodytracking by adaptive background models and mean-shift analysis**. 2003.

PORTEOUS, J.; BENINI, S.; CANINI, L.; CHARLES, F.; CAVAZZA, M.; LEONARDI, R. Interactive storytelling via video content recombination. In: MULTIMEDIA, 2010, New York, NY, USA. **Proceedings...** ACM, 2010. p.1715–1718. (MM '10).

PORTER, T.; DUFF, T. Compositing digital images. **SIGGRAPH Comput. Graph.**, New York, NY, USA, v.18, n.3, p.253–259, Jan. 1984.

POZZER, C. T. **Um Sistema para Geração, Interação e Visualização 3D de Histórias para TV Interativa**. 2005. Tese (Doutorado) — Pontifícia Universidade Católica do Rio de Janeiro.

R. STOLKIN A. GREIG, M. H.; GILBY, J. An em/e-mrf algorithm for adaptive model based tracking in extremely poor visibility. **Image and Vision Computing**, [S.l.], 2008.

RASMUSSEN, C.; HAGER, G. D. Probabilistic data association methods for tracking complex visual objects. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, [S.l.], 2001.

RUZON, M. A.; TOMASI, C. Alpha Estimation in Natural Images. **2012 IEEE Conference on Computer Vision and Pattern Recognition**, Los Alamitos, CA, USA, v.1, p.1018, 2000.

SALM, A. V. der; MARTÃNEZ, M.; FLIK, G.; BONGA, S. W. Effects of husbandry conditions on the skin colour and stress response of red porgy, Pagrus pagrus. **Aquaculture**, [S.l.], v.241, p.371 – 386, 2004.

SHARMA, G.; WU, W.; DALAL, E. N. The CIEDE2000 color-difference formula: implementation notes, supplementary test data, and mathematical observations. **Color research and application**, [S.l.], v.30, n.1, p.21–30, 2005.

SMITH, T.; GUILD, J. The C.I.E. colorimetric standards and their use. **Transactions of the Optical Society**, [S.l.], v.33, n.3, p.73, 1931.

STAUFFER C, G. W. **Adaptive background mixture models for real-time tracking**. 1999.

SUN, J.; JIA, J.; TANG, C.-K.; SHUM, H.-Y. Poisson matting. **ACM Trans. Graph.**, New York, NY, USA, v.23, n.3, p.315–321, Aug. 2004.

SZELISKI, R.; SHUM, H. Y. Creating full view panoramic mosaics and environment maps. In: ACM SIGGRAPH, 1997. **Proceedings...** [S.l.: s.n.], 1997.

TOYAMA K KRUMM J, B. B. M. B. **Wallflower**: principles and practice of background maintenance. 1999.

URSU, M.; KEGEL, I.; WILLIAMS, D.; THOMAS, M.; MAYER, H.; ZSOMBORI, V.; TUOMOLA, M.; LARSSON, H.; WYVER, J. ShapeShifting TV: interactive screen media narratives. **Multimedia Systems**, [S.l.], v.14, p.115–132, 2008. 10.1007/s00530-008-0119-z.

WANG; COHEN. Image and video matting: a survey. **Found. Trends. Comput. Graph. Vis.**, Hanover, MA, USA, v.3, n.2, p.97–175, Jan. 2007.

WANG, L.; GONG, M.; ZHANG, C.; YANG, R.; ZHANG, C.; YANG, Y.-H. Automatic Real-Time Video Matting Using Time-of-Flight Camera and Multichannel Poisson Equations. **International Journal of Computer Vision**, [S.l.], v.97, p.104–121, 2012. 10.1007/s11263-011-0471-x.

WREN CR AZARBAYEJANI A, D. T. P. A. Pfinder: realtime tracking of the human body. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, [S.l.], 1997.

WYSZECKI, G.; STILES, W. **Color Science**: concepts and methods, quantitative data and formulae. [S.l.]: John Wiley & Sons, 2000. (Wiley classics library).

ZAPPELLA, L.; LLADó, X.; SALVI, J. Motion Segmentation: a review. In: ARTIFICIAL INTELLIGENCE RESEARCH AND DEVELOPMENT: PROCEEDINGS OF THE 11TH INTERNATIONAL CONFERENCE OF THE CATALAN ASSOCIATION FOR ARTIFICIAL INTELLIGENCE, 2008., 2008, Amsterdam, The Netherlands, The Netherlands. **Proceedings...** IOS Press, 2008. p.398–407.

ZHANG, Y. **Advances in image and video segmentation**. [S.l.]: IRM Press, 2006.

ZHANG, Y.-J. **An Overview of Image and Video Segmentation 1 Chapter I An Overview of Image and Video Segmentation**. [S.l.]: IRM Press, 2006. p.16–30.