

IFT-SLIC: Geração de Superpixels com Base em Agrupamento Iterativo Linear Simples e Transformada Imagem-Floresta

Eduardo Barreto Alexandre

DISSERTAÇÃO
AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
DA
UNIVERSIDADE DE SÃO PAULO
PARA
DEFESA
DE
MESTRADO EM CIÊNCIAS DA COMPUTAÇÃO

Programa: MAC

Orientador: Prof. Dr. Paulo André Vechiatto de Miranda

O desenvolvimento deste trabalho recebeu auxílio financeiro da CAPES

São Paulo, maio de 2017

IFT-SLIC: Geração de Superpixels com Base em Agrupamento Iterativo Linear Simples e Transformada Imagem-Floresta

Esta versão da dissertação contém as correções e alterações sugeridas pela Comissão Julgadora durante a defesa deste trabalho realizada em 29/06/2017.

Comissão Julgadora

Prof. Dr. Paulo André Vechiatto de Miranda - orientador - IME-USP
Prof. Dr. Alexandre Xavier Falcão - IC-Unicamp
Prof. Dr. Jurandy G. Almeida Jr. - ICT-Unifesp

Agradecimentos

Primeiramente agradeço à USP pela estrutura e ensino de qualidade, o qual fez e fará grande diferença em minha vida. Aos meus pais amados, Antonio Carlos e Denise e minha irmã Isabela, que foram de enorme importância para meu desenvolvimento e superação através do amor, carinho, apoio e incentivo.

Também agradeço imensamente ao meu orientador Prof. Paulo Miranda por todos os valiosos ensinamentos, pela compreensão, empenho e dedicação na realização desse trabalho. Assim como aos meus ex-orientadores Prof. Antonio Carlos Sobieranski e Prof. Eros Comunello que me abriram os caminhos e o interesse acadêmico.

Sou muito grato aos meus queridos avós, Sydney Barreto e Dionice, as minhas tias Eugênia, Vera e Malu, ao meu tio Edson e as minhas primas Marta, Clara e Marina que me acolheram em suas casas onde encontrei muita atenção, força e uma extensão do meu lar.

Agradeço aos meus amigos Eliane e Gustavo que me ajudaram muito na revisão dessa dissertação. Aos meus amigos do IME, Aline Borges, Thilo Koch, Lucy Mansilla e Hans Harley por compartilharem tantas experiências inesquecíveis durante esses anos e a todos os amigos que diretamente ou indiretamente me ajudaram.

Sou muito grato à banca de qualificação e de defesa onde fez parte o Prof. Roberto Hirata Jr., Prof. Marcel Jackowski, Prof. Alexandre Falcão e Prof. Jurandy Almeida Jr., pelas correções e sugestões que ajudaram a enriquecer muito este trabalho. Também agradeço ao Prof. Roberto Hirata Jr. que me ajudou a ingressar na USP ao me apresentar ao meu orientador, e aos Professores Alexandre Falcão e Ananda Shankar Chowdhury e John E. Vargas Muñoz pela parceria e colaboração no desenvolvimento de artigos referentes a este trabalho.

Por fim, e não menos importante, agradeço à CAPES pelo apoio financeiro permitindo minha dedicação com maior enfoque e tranquilidade.

Todos vocês foram fundamentais para a realização desse sonho.

Resumo

ALEXANDRE, E. B. **IFT-SLIC: Geração de Superpixels com Base em Agrupamento Iterativo Linear Simples e Transformada Imagem-Floresta.** 2017. Defesa (Mestrado) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2017.

A representação de imagem baseada em superpixels tem se tornado indispensável na melhoria da eficiência em sistemas de Visão Computacional. Reconhecimento de objetos, segmentação, estimativa de profundidade e estimativa de modelo corporal são alguns importantes problemas nos quais superpixels podem ser aplicados. Porém, superpixels podem influenciar a qualidade dos resultados do sistema positiva ou negativamente, dependendo de quanto bem eles respeitam as fronteiras dos objetos na imagem. Neste trabalho, é proposto um método iterativo para geração de superpixels, conhecido por IFT-SLIC, baseado em sequências de Transformadas Imagem-Floresta, começando com uma grade regular de sementes. Um procedimento de recomputação de pixels sementes é aplicado a cada iteração, gerando superpixels conexos com melhor aderência às bordas dos objetos presentes na imagem. Os superpixels obtidos via IFT-SLIC correspondem, estruturalmente, a árvores de espalhamento enraizadas nessas sementes, que naturalmente definem superpixels como regiões de pixels fortemente conexas. Comparadas ao Agrupamento Iterativo Linear Simples (SLIC), o IFT-SLIC considera os custos dos caminhos mínimos entre pixels e os centros dos agrupamentos, em vez de suas distâncias diretas. Funções de conexidade não monotonicamente incrementais são exploradas em neste método resultando em melhor desempenho. Estudos experimentais indicam resultados de extração de superpixels superiores pelo método proposto em comparação com o SLIC. Também é analisada a efetividade do IFT-SLIC, em termos de medidas de eficiência e acurácia, em uma aplicação de segmentação do céu em fotos de paisagens. Os resultados mostram que o IFT-SLIC é competitivo com os melhores métodos do estado da arte e superior a muitos outros, motivando seu desenvolvimento para diferentes aplicações.

Palavras-chave: Agrupamento Iterativo Linear Simples, Transformada Imagem-Floresta, Superpixel, Segmentação não supervisionada.

Abstract

ALEXANDRE, E. B. **IFT-SLIC: Superpixel generation based on Simple Linear Iterative Clustering and Image Foresting Transform.** 2017. Defesa (Mestrado) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2017.

Image representation based on superpixels has become indispensable for improving efficiency in Computer Vision systems. Object recognition, segmentation, depth estimation, and body model estimation are some important problems where superpixels can be applied. However, superpixels can influence the quality of the system results in a positive or negative manner, depending on how well they respect the object boundaries in the image. In this work, we propose an iterative method for superpixels generation, known as IFT-SLIC, which is based on sequences of Image Foresting Transforms, starting with a regular grid for seed sampling. A seed pixel recomputation procedure is applied per each iteration, generating connected superpixels with a better adherence to objects borders present in the image. The superpixels obtained by IFT-SLIC structurally correspond to spanning trees rooted at those seeds, that naturally define superpixels as regions of strongly connected pixels. Compared to Simple Linear Iterative Clustering (SLIC), IFT-SLIC considers minimum path costs between pixel and cluster centers rather than their direct distances. Non-monotonically increasing connectivity functions are explored in our IFT-SLIC approach leading to improved performance. Experimental results indicate better superpixel extraction by the proposed approach in comparation to that of SLIC. We also analyze the effectiveness of IFT-SLIC, according to efficiency, and accuracy on an application – namely sky segmentation. The results show that IFT-SLIC can be competitive to the best state-of-the-art methods and superior to many others, which motivates it's further development for different applications.

Palavras-chave: Simple Linear Iterative Clustering, Image Foresting Transform, Superpixel, Unsupervised Segmentation.

Sumário

Lista de Abreviaturas	xi
Lista de Figuras	xiii
Lista de Tabelas	xvii
1 Introdução	1
1.1 Objetivos e Contribuições	2
1.2 Trabalhos Relacionados	5
1.3 Segmentação de Objetos	8
1.4 Organização do Trabalho	9
2 Conceitos Gerais	11
2.1 Imagem Digital	11
2.1.1 Multidimensionalidade	11
2.1.2 Imagem Multi-banda	12
2.1.3 Imagem de Rótulos	12
2.1.4 Objeto	12
2.1.5 Segmentação e Superpixels	12
2.2 Grafos	12
2.2.1 Relação de Adjacência	12
2.2.2 Subgrafo	12
2.2.3 Dígrafo	13
2.2.4 Grafo Ponderado	13
2.2.5 Caminho	13
2.2.6 Ciclo	13
2.2.7 Grafo Conexo	13
2.2.8 Corte no Grafo	13
2.2.9 Árvore e Floresta	13
2.2.10 Imagem como Grafo	14
2.3 Avaliação de Métodos de Segmentação	15
3 Conceitos Específicos	19
3.1 Função de Conexidade	19
3.1.1 Função Monotonicamente Incremental	20
3.2 Mapa de Predecessores	20

3.3	Floresta de Espalhamento	20
3.4	IFT	20
3.4.1	Ambiguidade	22
3.4.2	Algoritmo	22
3.4.3	Funcionamento	22
3.4.4	Exemplos de Segmentação	28
3.5	DIFT	29
3.5.1	Algoritmo	29
3.5.2	Teste de Predecessor	32
3.6	SLIC	33
4	IFT-SLIC	35
4.1	Algoritmo	36
4.2	Propriedades Teóricas	36
4.2.1	Convergência com Funções não Monotonicamente Incrementais	38
4.3	Detalhes de Implementação	39
5	Resultados	43
5.1	Bases de Imagens	44
5.2	Metodologia	45
5.3	Análise de Parâmetros	46
5.3.1	SLIC	46
5.3.2	IFT-SLIC	51
5.4	Comparativos	55
5.5	Eficiência	68
5.6	Aplicação de Alto Nível	70
5.7	Conclusão	72
5.8	Trabalhos Futuros	76
	Referências Bibliográficas	77

Listas de Abreviaturas

IFT	Transformada Imagem-Floresta (<i>Image Foresting Transform</i>)
DIFT	Transformada Imagem-Floresta Diferencial (<i>Differential Image Foresting Transform</i>)
SLIC	Agrupamento Iterativo Linear Simples (<i>Simple Linear Iterative Clustering</i>)
RM	Ressonância Magnética
3T	3 Tesla
SF	Floresta de Espalhamento (<i>Spanning Forest</i>)
FP	Falso Positivo (<i>False Positive</i>)
FN	Falso Negativo (<i>False Negative</i>)
TP	Verdadeiro Positivo (<i>True Positive</i>)
TN	Verdadeiro Negativo (<i>True Negative</i>)
MI	Monotonicamente Incremental (<i>Monotonically Increasing</i>)
NMIF	Função Não Monotonicamente Incremental (<i>Non-Monotonically Increasing Function</i>)
FIFO	Primeiro a Entrar, Primeiro a Sair (<i>First-In-First-Out</i>)
LIFO	Último a Entrar, Primeiro a Sair (<i>Last-In-First-Out</i>)

Listas de Figuras

1.1	Resultados de segmentações com o SLIC e IFT-SLIC usando 4 (a, b), 50 (c, d), 72 (e, f) e 97 (g, h) superpixels. (a), (c) e (g) contém falhas de segmentação, flechas indicam essas falhas em (c) e (g) para fácil identificação. (e) apresenta uma segmentação sem falhas no SLIC, porém (g) demonstra que as falhas podem reaparecer ao se adicionar novos superpixels.	4
1.2	Exemplo da limitação de distribuição de sementes no SLIC. (a) Imagem contendo um objeto de interesse na sua região central. (b) Uma maior concentração de sementes é considerada na região central a fim de melhorar a segmentação do objeto, com correspondente redução da densidade de sementes no fundo homogêneo. É importante notar que cada semente é representada por um círculo e sua região máxima ($2S \times 2S$) pelo quadrado em sua volta. Por fim, (c) demonstra em vermelho todas as regiões da imagem que estão fora do alcance de uma semente a serem conquistadas.	5
2.1	(a-c) Exemplos de adjacência circular. (d-f) Exemplos de adjacência retângular.	14
2.2	(a-b) Imagem de entrada e seu gabarito. (c) Um exemplo de segmentação por um algoritmo qualquer. (d) A classificação dos pixels da segmentação entre pixels do objeto e fundo corretamente segmentados, TP e TN respectivamente, e entre pixels do objeto e fundo incorretamente segmentados, FN e FP respectivamente.	17
3.1	Uma SF com 3 árvores <i>a</i> , <i>b</i> e <i>c</i>	21
3.2	Exemplos de políticas de desempate.	24
3.3	(a) Um grafo com duas raízes iniciais com valor 0 e os custos de cada aresta. (b-c) Duas possíveis SF de caminhos ótimos através da função f_{max} (Equação 3.2). (d) Zona de empate no grafo (cor cinza).	25
3.4	Exemplo de execução da IFT com função de conexidade f_{max} em um grafo. Os nós com cor de fundo laranja correspondem aos nós presentes na fila de prioridade Q a cada iteração. Cor de fundo verde é usada para indicar nós que foram removidos de Q em definitivo. Os valores dentro dos nós representam os custos dos melhores caminhos encontrados.	28
3.5	Exemplo de segmentação supervisionada, onde S_o e S_f representam as sementes do objeto e do fundo respectivamente. (a) Exemplos de caminhos ótimos ($\langle S_o, \dots, a \rangle$ e $\langle S_f, \dots, b \rangle$) e caminhos não ótimos ($\langle S_o, \dots, b \rangle$ e $\langle S_f, \dots, a \rangle$). (b) Segmentação final onde a borda branca representa os limites entre as SFs de S_o e S_f	29
3.6	(a-b) Imagem de entrada e as sementes selecionadas de forma não supervisionada. (c) A segmentação final onde cada semente gerou sua própria SF de caminhos ótimos.	30

3.7 (a) Exemplo de uma SF de caminhos ótimos com duas árvores com raízes em a e b , onde $\langle a, \dots, s, t, \dots, u \rangle$ é um caminho ótimo. Em (b), a semente c é adicionada. Já em (c), o caminho $\langle c, \dots, s \rangle$ é encontrado como um caminho ótimo mais barato que $\langle a, \dots, s \rangle$, porém o custo atual de t ($C(t)$) é igual ao custo de $f(\langle c, \dots, s, t \rangle)$. O teste de predecessor garante nesse caso que todos os pixels com caminhos ótimos passando por s , como é o caso de $\langle t, \dots, u \rangle$, serão reavaliados e possivelmente reatribuídos para a semente c . (d) Uma possível segmentação final da DIFT com a adição da semente c .	33
4.1 Exemplo da percentagem de sementes em forma decimal a serem computadas diminuindo na base de imagens Grabcut a cada iteração.	40
4.2 Exemplo da percentagem de sementes em forma decimal a serem computadas diminuindo na base de imagens Liver a cada iteração.	40
4.3 Exemplo da percentagem de sementes em forma decimal a serem computadas diminuindo na base de imagens Tálus a cada iteração.	41
5.1 Imagens da base de imagens Grabcut, com seus respectivos gabinetes.	44
5.2 Imagens da base de imagens Liver, com seus respectivos gabinetes.	45
5.3 Imagens da base de imagens Talus, com seus respectivos gabinetes.	45
5.4 (a-b-c) Imagem de entrada, seu gabarito e imagem de rótulos. (d-e) Dividem o superpixel entre regiões A e B , na qual A representa a área do superpixel contida no fundo e B na frente. (f) A segmentação resultante.	46
5.5 (a-b) Imagem de entrada e seu gabarito. Os superpixels de 40×40 são computados por: (c) SLIC e (e) IFT-SLIC. Foi atribuído para cada superpixel o rótulo mais frequente do gabarito ocorrendo em seu interior: (d) O resultado do SLIC tem $Dice = 0.9659$, e (f) A IFT-SLIC tem $Dice = 0.9694$.	47
5.6 Demonstração do uso da etapa de perturbação das sementes.	48
5.7 Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Grabcut.	48
5.8 Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Liver.	49
5.9 Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Talus.	49
5.10 Comparação da média de acurácia do uso do parâmetro p ativado ou desativado no SLIC para as bases de imagens Grabcut (a), Liver (b) e Talus (c), além de sua média geral (d).	50
5.11 Demonstração do uso do parâmetro m .	51
5.12 Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Grabcut.	52
5.13 Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Liver.	53
5.14 Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Talus.	53
5.15 Comparação da média de acurácia do uso do parâmetro p ativado ou desativado para as bases de imagens Grabcut (a), Liver (b) e Talus (c), além de sua média geral (d).	54

5.16 (a) Processo de seleção do centro de um agrupamento usado pelo SLIC. No exemplo, o centro acaba caindo em uma região fora do agrupamento. Em (b-c) é apresentada a solução usada pelo IFT-SLIC. Em (b) há uma varredura pelo nó pertencente ao agrupamento mais próximo do centro definido em (a). Após o nó ser encontrado, ele passa a ser o novo centro do agrupamento e portanto dentro do agrupamento como mostrado em (c).	55
5.17 Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Grabcut.	56
5.18 Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Liver.	57
5.19 Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Talus.	57
5.20 Demonstração da combinação dos parâmetros <i>alfa</i> e <i>beta</i>	58
5.21 As curvas de acurácia média de segmentação da base de imagens Grabcut para superpixels de diferente tamanhos.	59
5.22 As curvas de acurácia média de segmentação da base de imagens Liver para superpixels de diferente tamanhos.	59
5.23 As curvas de acurácia média de segmentação da base de imagens Talus para superpixels de diferente tamanhos.	60
5.24 Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Grabcut.	61
5.25 Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Liver.	62
5.26 Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Talus.	62
5.27 Exemplos de segmentação da base de imagens Grabcut. A pontuação Dice do SLIC (c) é 0.98817 e do IFT-SLIC (d) é 0.990667.	63
5.28 Exemplos de segmentação da base de imagens Liver. A pontuação Dice do SLIC (c) é 0.974979 e do IFT-SLIC (d) é 0.975518.	64
5.29 Exemplos de segmentação da base de imagens Tálu. A pontuação Dice do SLIC (c) é 0.854148 e do IFT-SLIC (d) é 0.961013.	65
5.30 Curvas obtidas pela comparação entre acurácia e compacidade da variação dos parâmetros do SLIC e IFT-SLIC na base de imagens Grabcut.	66
5.31 Curvas obtidas pela comparação entre acurácia e compacidade da variação dos parâmetros do SLIC e IFT-SLIC na base de imagens Liver.	67
5.32 Curvas obtidas pela comparação entre acurácia e compacidade da variação dos parâmetros do SLIC e IFT-SLIC na base de imagens Talus.	68
5.33 Exemplo de estabilização da pontuação (Dice) dos superpixels no IFT-SLIC em cada iteração para a base de imagens Grabcut.	69
5.34 Exemplo de estabilização da pontuação (Dice) dos superpixels no IFT-SLIC em cada iteração para a base de imagens Liver.	69
5.35 Exemplo de estabilização da pontuação (Dice) dos superpixels no IFT-SLIC em cada iteração para a base de imagens Tálu.	70

5.36	Tempo de execução e acurácia correspondente para a base de imagens Grabcut.	70
5.37	Tempo de execução e acurácia correspondente para a base de imagens Liver.	71
5.38	Tempo de execução e acurácia correspondente para a base de imagens Tálus.	71
5.40	Imagens da base de imagens Sky, com seus respectivos gabaritos.	72
5.39	Demonstração das etapas do algoritmo de segmentação do céu.	73
5.41	Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Sky.	74
5.42	Exemplos de segmentação da base de imagens Sky. Na primeira linha, a pontuação Dice do SLIC (c) é 0.991623 e do IFT-SLIC (d) é 0.995687. Na segunda, a pontuação Dice do SLIC (c) é 0.884502 e do IFT-SLIC (d) é 0.99737.	75

Lista de Tabelas

2.1	Matriz de confusão	15
5.1	Tabela de parâmetros, SLIC varia o parâmetro m de 1 até 80, IFT-SLIC varia o parâmetro alfa de 0.0025 até 0.2 com passos de 0.0025	43

Capítulo 1

Introdução

Particionar uma imagem em regiões distintas, que, idealmente, correspondem a diferentes objetos do mundo real, é um passo importante para a análise de conteúdo de imagens. Entretanto, a etapa de segmentação pode tornar-se extremamente complicada devido à imensa variabilidade de cor, iluminação e textura que se manifesta em uma imagem. Essa situação é agravada quando o número de regiões a serem classificadas na imagem é também desconhecido.

No caso de segmentação de imagem supervisionada, o usuário fornece uma rotulação parcial da imagem, pintando alguns pixels dela pertencentes aos diferentes objetos (pixels sementes). Essa rotulação parcial é então propagada para os demais pixels da imagem, seguindo algum critério para a partição ótima da imagem. Já no caso da segmentação não supervisionada, seu particionamento é extraído diretamente dos padrões presentes na imagem.

A super-segmentação (*over-segmentation*) não supervisionada de uma imagem, comumente chamada de superpixels, é uma forma conveniente de partitionar uma imagem em regiões relevantes que podem, em conjunto, representar objetos (Panagiotakis *et al.*, 2013; Rauber *et al.*, 2013). Esse agrupamento pode reduzir significativamente o tempo de processamento dos algoritmos, substituindo a estrutura rígida de grade de pixels (Achanta *et al.*, 2012). Um superpixel pode ser definido como uma região compacta de pixels similares e conexos, que representam, localmente, uma mesma estrutura da imagem. A medida de similaridade empregada pode ser definida de várias maneiras, usando intensidade, cor, textura e posição como características. Uma vez que os pixels contidos no mesmo superpixel são considerados iguais por definição, primitivas de superpixels têm algumas vantagens sobre as primitivas de pixels, como eficiência computacional, já que o número de primitivas é reduzido no nível de superpixel. Isso traz grandes oportunidades para aliviar a complexidade de sistemas de Visão Computacional (Cuadros *et al.*, 2012; Çigla e Alatan, 2010).

Superpixels podem ser usados em uma grande variedade de aplicações: segmentação de imagens médicas (Wu *et al.*, 2014), segmentação do céu em fotos de paisagem (*sky segmentation*) (Kostolansky, 2016), segmentação de movimento (*motion segmentation*) (Ayvaci e Soatto, 2009), segmentação de objetos em múltiplas classes (*multi-class object segmentation*) (Fulkerson *et al.*, 2009a; Yang *et al.*, 2012), detecção de objetos (*object detection*) (Shu *et al.*, 2013), detecção de saliência espaço-temporal (*spatiotemporal saliency detection*) (Liu *et al.*, 2014), rastreamento de alvos (*target tracking*) (Yang *et al.*, 2014), estimativa de modelo corporal (*body model estimation*) e estimativa de profundidade (*depth estimation*) (Zitnick e Kang, 2007).

Apesar do ganho de eficiência, a representação de uma imagem via superpixels pode afetar, positivamente ou negativamente, a eficácia dos algoritmos. Por isso, é crucial que os superpixels respeitem os limites dos objetos na imagem, de tal forma que um objeto possa ser precisamente definido por um conjunto de superpixels. Um bom algoritmo de geração de superpixel deve possuir as seguintes propriedades desejáveis (Achanta *et al.*, 2012):

P.1 Capacidade de aderir às bordas dos objetos na imagem: Os métodos devem respeitar e preservar as estruturas locais apresentadas na imagem, uma vez que o objetivo de um superpixel é representar parte de algum objeto na imagem;

P.2 Flexibilidade na escolha do número de superpixels gerados: Os métodos devem, idealmente, permitir a personalização do número desejado de superpixels, a fim de evitar segmentação insuficiente, isto é, a segmentação da imagem em poucas regiões, onde um mesmo superpixel poderia erroneamente conter partes de diferentes objetos;

P.3 Eficiência: Superpixels precisam ser gerados da forma mais rápida possível para que eles não adicionem muita sobrecarga para o resto do *pipeline* limitando seus benefícios; além disso, eles devem ser simples de estender para imagens multidimensionais (ex: imagens tridimensionais de ressonância magnética (RM));

P.4 Segmentação dada por uma imagem de rótulos: Os superpixels não devem sobrepor-se uns aos outros. Cada pixel deve ser atribuído a um único superpixel;

P.5 Compacidade: Superpixels devem ter tamanho e forma uniforme. A capacidade de controlar a compacidade dos superpixels é importante. Superpixels regulares e compactos são muitas vezes desejáveis, pois tendo dimensões limitadas e poucos vizinhos, obtém-se um grafo de superpixels mais interpretável, a partir do qual pode-se extrair medidas locais mais relevantes. Para estender diferentes operações de processamento de imagem como convolução (filtragem do nível de pixel) para o nível de superpixel, uma distribuição regular de superpixels com tamanho e vizinhança mais uniforme é absolutamente necessária. Observe que essas operações são sempre executadas no nível de pixel com um tamanho de janela fixo (mesmo número de vizinhos para qualquer pixel).

1.1 Objetivos e Contribuições

Neste trabalho, é proposto um arcabouço de geração de superpixels, baseado em sequências de Transformada Imagem-Floresta (*Image Foresting Transform IFT*) (Falcão *et al.*, 2004). Cada iteração deste executa uma IFT a partir de um conjunto distinto de sementes, gerando uma sequência de resultados de segmentação que melhoram ao longo das iterações até a sua convergência. Dado que o algoritmo proposto foi inspirado em um dos mais populares algoritmos de geração de superpixel, chamado Agrupamento Iterativo Linear Simples (*Simple Linear Iterative Clustering SLIC*) (Achanta *et al.*, 2010, 2012), em sua primeira publicação foi adotado o nome de IFT-SLIC (Barreto-Alexandre *et al.*, 2015). Uma vez que os superpixels correspondem estruturalmente a árvores de espalhamento (*spanning trees*) enraizadas nas suas sementes, posteriormente foi adotado o nome de Floresta de Espalhamento Iterativa (*Iterative Spanning Forest ISF*)¹. Neste trabalho, é adotada a seguinte convenção: o termo ISF será usado para designar o arcabouço de geração de superpixels, que permite a escolha de uma função de custo de caminho mais adequada para cada aplicação, e IFT-SLIC para indicar uma particular instância deste arcabouço, com uma função de custo específica. Neste trabalho será dada maior ênfase no método IFT-SLIC.

O SLIC adapta, essencialmente, o algoritmo *k-means* para a geração de superpixel. Uma vez que o *k-means* baseia-se nas distâncias diretas entre pixel e centros dos agrupamentos, os pixels semelhantes podem não agrupar em uma região conexa, mesmo localmente. Esse problema é tratado no SLIC por meio de um pós-processamento, porém, seu uso resulta na criação de novos agrupamentos, impossibilitando a garantia da geração do número fixo de superpixels inicialmente estabelecido. No ISF, a função de distância é dada pelo custo do caminho mínimo em um grafo derivado da imagem, de tal forma que superpixels são naturalmente definidos como regiões compactas de pixels fortemente conexos. Esse resultado, não só melhora a qualidade dos superpixels, mas também reduz a geração de superpixel à escolha de uma função de caminho de custo apropriada para uma dada aplicação, além de sempre garantir um número fixo de superpixels.

Devido à limitação do uso do *k-means* discutida acima, o SLIC necessita de uma restrição adicional para garantir a compacidade de seus superpixels; essa restrição é aplicada durante o cálculo das distâncias de cada pixel, de modo que apenas centros de agrupamentos dentro de uma

¹<http://www.ic.unicamp.br/~afalcao/isf-superpixels/>

área $2\mathcal{S} \times 2\mathcal{S}$ ao redor do pixel em análise são considerados, onde $\mathcal{S} = \sqrt{N/k}$, N é o número de pixels da imagem e k o número de superpixels inicialmente estabelecido. Esta restrição, além de auxiliar na compacidade do superpixel, evita a geração de regiões desconexas distantes de seu centro, porém ocasiona em várias falhas de segmentação quando o número de superpixels é reduzido. Figura 1.1² mostra esse comportamento do SLIC; observa-se várias falhas de segmentação do SLIC quando o número de superpixels não é o suficiente para sobrepor essa restrição, é o caso nas Figuras 1.1a-c onde o número de superpixels usados são 4 e 50 respectivamente, apenas com 72 superpixels o SLIC é capaz de gerar uma segmentação sem falhas (Figura 1.1e), porém mesmo com o aumento de superpixels não há uma garantia de uma melhora na segmentação (Figura 1.1g). As Figuras 1.1b,d,f,h mostram os resultados das segmentações da mesma imagem usando IFT-SLIC, na qual o método é capaz de obter uma segmentação sem falhas com apenas 4 superpixels, ou seja com um número bem menor de superpixels em comparação com o SLIC (Figura 1.1e), além de mantê-la independentemente do número de superpixels usados.

Outro problema devido a essa restrição no SLIC é a obrigatoriedade de uma densidade homogênea de sementes da imagem devido à necessidade de existir sementes vizinhas conectadas dentro da área máxima de $2\mathcal{S} \times 2\mathcal{S}$, ou seja, caso a maioria das sementes em uma imagem estiverem concentradas em uma pequena região, o resto da imagem conterá poucas sementes distribuídas esparsamente ocasionando em regiões órfãs, a Figura 1.2 demonstra este efeito. O IFT-SLIC, por outro lado, não traz nenhuma restrição espacial, permitindo todos os pixels da imagem serem conquistados por algum superpixel independentemente de sua posição e distância das sementes. No arcabouço do ISF³, uma seleção heterogênea de sementes iniciais pode ser usada, em que a densidade de sementes muda de acordo com a entropia da região, de modo a concentrar um maior número de superpixels em regiões mais relevantes da imagem.

No contexto da segmentação não supervisionada de imagens, os métodos baseados na IFT geralmente consideram apenas funções monotonicamente incrementais⁴ (Rocha *et al.*, 2009). O método IFT-SLIC também inova por considerar uma Função Não Monotonicamente Incremental (*Non-Monotonically Increasing Function* NMIF) em segmentação não supervisionada. Estudos envolvendo NMIF similares, em segmentação supervisionada, mostraram que elas se moldam às características particulares das imagens de forma mais eficaz e são mais adaptáveis ao lidar com problemas de falta de homogeneidade (Mansilla *et al.*, 2013a).

Historicamente, as NMIF foram evitadas, dado que os caminhos gerados pela IFT com NMIF podem não ter otimalidade garantida de acordo com a definição apresentada por Falcão *et al.* (2004). Estudos anteriores consideravam apenas o uso de NMIF para calcular uma aproximação da distância Euclideana (Falcão *et al.*, 2002; Torres *et al.*, 2004) e em reconstrução superior local⁵, embora outros estudos por Herman e Carvalho (2001) sugerem vantagens práticas do uso de NMIF no contexto da segmentação interativa de imagens (Herman e Carvalho, 2001; Miranda *et al.*, 2008).

Mais recentemente, Strand *et al.* (2013) propôs o método por *Minimum Barrier Distance*, ao qual, em cenários digitais, leva a uma NMIF. Esse método vem mostrando bons resultados em comparações empíricas no contexto de segmentação supervisionada (Mansilla *et al.*, 2013a). Também em outros trabalhos recentes, foi provado que algumas NMIF geram resultados ótimos de acordo com outros critérios de otimalidade, como uma função de energia para perseguição ótima de bordas (Miranda *et al.*, 2011, 2012), ou uma energia de corte em grafo para segmentação ótima por região (Mansilla e Miranda, 2013a,b; Mansilla *et al.*, 2013a; Miranda e Mansilla, 2014). Isso reacendeu a pesquisa envolvendo NMIF.

Atualmente, métodos baseados na IFT com NMIF têm sido utilizados com sucesso no contexto de segmentação supervisionada de imagens (Mansilla e Miranda, 2013a,b; Mansilla *et al.*, 2013a; Miranda e Mansilla, 2014). As funções não monotonicamente incrementais compõem uma classe

²Os valores 4, 50, 72 e 97 foram usados devido ao SLIC não respeitar a propriedade P.2 gerando números de superpixels diferentes dos desejados, dificultando assim na escolha deles.

³<http://www.ic.unicamp.br/~afalcao/isf-superpixels/>

⁴Uma excessão seria a IFT Euclideana que não é monotonicamente incremental.

⁵Anotações de aula da disciplina “MO815: Processamento de Imagens usando Grafos” <http://www.ic.unicamp.br/~afalcao/mo815-grafos/index.html>.

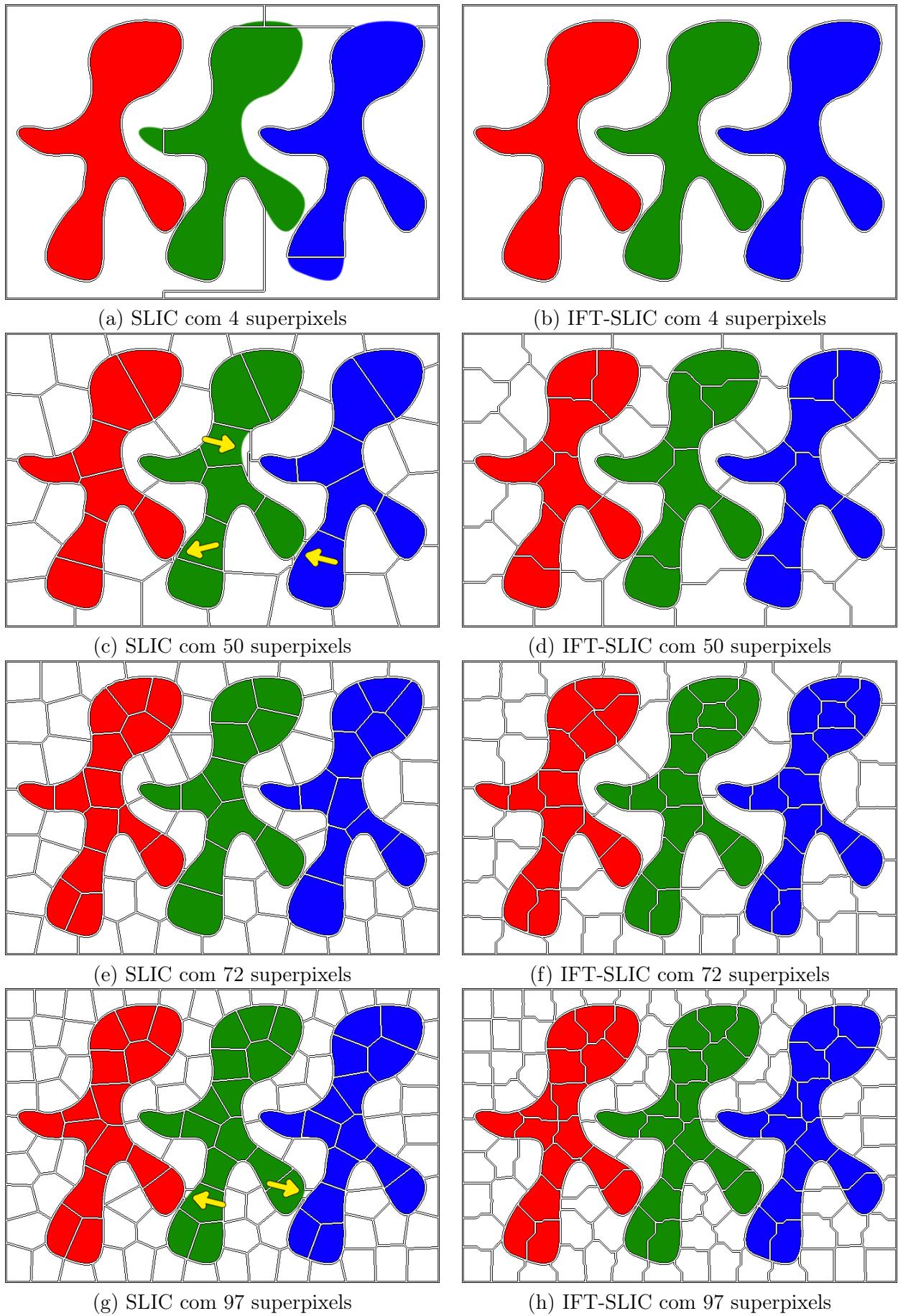


Figura 1.1: Resultados de segmentações com o SLIC e IFT-SLIC usando 4 (a, b), 50 (c, d), 72 (e, f) e 97 (g, h) superpixels. (a), (c) e (g) contém falhas de segmentação, flechas indicam essas falhas em (c) e (g) para fácil identificação. (e) apresenta uma segmentação sem falhas no SLIC, porém (g) demonstra que as falhas podem reaparecer ao se adicionar novos superpixels.

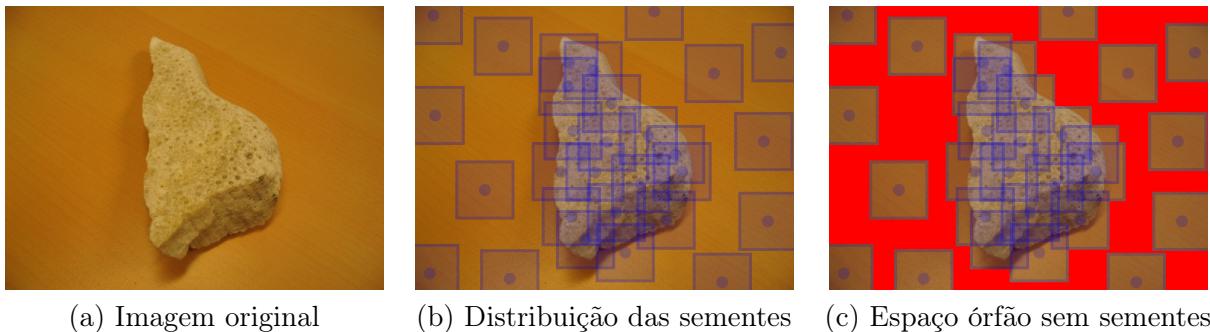


Figura 1.2: Exemplo da limitação de distribuição de sementes no SLIC. (a) Imagem contendo um objeto de interesse na sua região central. (b) Uma maior concentração de sementes é considerada na região central a fim de melhorar a segmentação do objeto, com correspondente redução da densidade de sementes no fundo homogêneo. É importante notar que cada semente é representada por um círculo e sua região máxima ($2S \times 2S$) pelo quadrado em sua volta. Por fim, (c) demonstra em vermelho todas as regiões da imagem que estão fora do alcance de uma semente a serem conquistadas.

de funções de conexidade menos restritas, o que permitiu avanços como a incorporação de informação de polaridade de borda (Mansilla e Miranda, 2013a; Miranda e Mansilla, 2014), o uso de restrições de forma (Mansilla e Miranda, 2013b; Mansilla *et al.*, 2013b) e restrições de conexidade (Mansilla e Miranda, 2016; Mansilla *et al.*, 2016). Essas restrições permitem customizar a segmentação para um dado objeto alvo, de acordo com suas propriedades de alto nível. No contexto de segmentação não supervisionada de imagens, porém, praticamente ainda não existiam estudos da aplicação de NMIF.

1.2 Trabalhos Relacionados

Vários métodos de segmentação não supervisionada de imagens usando grafos consideram um grafo esparsa $G = (V, A)$, em que os nós no conjunto V são os pixels, e as arestas em A são definidas entre pixels vizinhos. Os pesos das arestas são dados por uma função de dissimilaridade entre as características dos pixels vizinhos (ex: magnitude do gradiente de intensidades), ou pela sua noção dual dada por uma função de afinidade/similaridade.

Para um dado grafo derivado da imagem, uma relação binária entre pixels contida no produto cartesiano $V \times V$ que satisfaz as propriedades de reflexividade, simetria e transitividade, induz naturalmente a uma segmentação da imagem, pois uma relação de equivalência permite particionar o grafo em classes de equivalência. Por exemplo, para um grafo não direcionado, considere a relação binária $a \xrightarrow{\kappa} b$ tal que existe um caminho interligando os pixels a e b , contendo apenas arestas com pesos abaixo de um dado limiar κ . As partições geradas por essa relação binária correspondem aos componentes conexos do grafo, após a remoção de todas as arestas com pesos maiores ou iguais a κ . Variando o parâmetro κ , é possível construir uma hierarquia das partições. Infelizmente, tais abordagens em geral não geram boas segmentações, apesar de sua elegância e simplicidade.

Alguns métodos mais elaborados seguem uma estratégia divisiva (*top-down clustering* ou *divisive clustering*), como por exemplo os métodos de segmentação por *Mean Cut* (Wang e Siskind, 2003; Wang e Siskind, 2001), *Normalized Cut* (Carballido-Gamio *et al.*, 2004; Shi e Malik, 2000; Çigla e Alatan, 2010), ou pela remoção de arestas inconsistentes da árvore de Espalhamento Mínima (Duda e Hart, 1973). Esses métodos são naturalmente hierárquicos. Uma árvore das partições é construída pelo processo de sucessivas divisões binárias.

Outros métodos hierárquicos mais simples utilizam uma estratégia de agrupamento aglomerativo, em que o histórico de fusões é, em geral, representado na forma de um dendrograma. São exemplos dessa classe os métodos *Average Linkage*, *Complete Linkage*, *Single Linkage* (Duda e Hart, 1973), e o método proposto por Felzenszwalb e Huttenlocher (2004). Alguns autores consideram também o uso de métodos computacionalmente mais caros, tais como os baseados em teoria de

redes complexas usados por Cuadros *et al.* (2012), que são viabilizados pelo emprego de superpixels. Inicialmente, cada pixel (ou superpixel) é considerado como um grupo individual, e grupos são recursivamente fundidos até produzir um bom agrupamento final. A cada passo são juntados os dois grupos com a menor distância entre agrupamentos. O problema desses métodos é que nas primeiras etapas, as decisões são tomadas com base apenas em características locais da imagem sem levar em conta informações de mais alto nível.

Métodos supervisionados podem, muitas vezes, também serem adaptados para segmentação não supervisionada mediante o emprego de um critério adequado para seleção automática de sementes. Por exemplo, a versão não supervisionada do IFT-*Watershed* de Marcadores (Lotufo e Falcão, 2000; Lotufo *et al.*, 2002), sobre uma imagem de magnitude de gradiente, corresponde a um IFT-*Watershed* na qual os marcadores são selecionados em alguns mínimos regionais da imagem de gradiente, com um rótulo distinto para cada mínimo regional (Audigier e Lotufo, 2007)⁶. Variações que contornam o problema da super-segmentação podem ser obtidas através do preenchimento de bacias de acordo com critérios de altura, área, ou volume, a partir de filtros conexos (Falcão *et al.*, 2001; Salembier e Serra, 1995; Salembier *et al.*, 1998). Esse mesmo procedimento poderia ser estendido, tendo como base vários outros métodos supervisionados, no entanto, alguns métodos como o de Fluxo Máximo-Corte Mínimo (Boykov e Funka-Lea, 2006; Boykov e Jolly, 2001) se tornam um problema NP-difícil no caso de múltiplos rótulos, enquanto que no caso do método dos Passeios Aleatórios (Grady, 2006), tem-se que o número de sistemas de equações lineares a serem resolvidos cresce de modo proporcional ao número de rótulos distintos, inviabilizando a solução.

Um problema recorrente em todas as abordagens discutidas acima, quando aplicadas sobre um grafo esparsa com pesos dados pela dissimilaridade entre pixels vizinhos (ex: magnitude do gradiente de intensidades), ou o seu complemento, é que, em geral, a segmentação não apresenta bons resultados em partes finas dos objetos presentes na imagem. Também o tratamento de transições em forma de rampa entre regiões depende de uma boa escolha de um fator de escala no cálculo do gradiente (ex: tamanho do kernel de convolução), porém a escala ótima pode variar para diferentes regiões da imagem.

Outros tipos de métodos em grafos consideram as relações topológicas entre os componentes conexos binários obtidos pela decomposição por limiarização da imagem de entrada, também conhecida como árvore de Componentes (Salembier *et al.*, 1998). Uma implementação eficiente dessa árvore é conhecida como *Max-Tree*, pois as folhas são sempre os máximos regionais da imagem, e o seu caso dual é conhecido como *Min-Tree* (Najman e Couprie, 2006). Mais recentemente foi proposta uma representação conjunta da *Max-Tree* e *Min-Tree* chamada *Tree of Shapes* (Ballester *et al.*, 2003; Géraud *et al.*, 2013). Métodos de segmentação não supervisionada de imagens podem ser obtidos por meio da análise das várias formas codificadas na *Tree of Shapes* (Xu *et al.*, 2012). No entanto, uma vez que os objetos correspondem a componentes conexos binários obtidos por limiarização, esses métodos baseados na *Tree of Shapes* não fornecem bons resultados no caso de imagens com fortes problemas de inhomogeneidade.

Dado que vários métodos em grafos discutidos anteriormente podem ser aplicados nos mais diferentes tipos de grafos, alguns autores⁷, recentemente, têm aplicado técnicas de operadores conexos em conjunto com grafos direcionados levando em conta o seu uso de sucesso em trabalhos como Miranda e Mansilla (2014) e Boykov e Funka-Lea (2006). Outros têm aplicado técnicas em grafos modelados diretamente no espaço de características (Rocha *et al.*, 2009). Nesse contexto, técnicas como *Mean Shift* (Cheng, 1995) são muito populares dentro da comunidade de Visão Computacional (Comaniciu e Meer, 2002). Esse método consiste em deslocar as amostras para regiões de maior densidade seguindo a subida do gradiente, porém, sem calcular explicitamente as densidades. O *Mean Shift* é essencialmente uma versão dual no contínuo do *Watershed* por paradigma de queda de chuva⁸. O método descrito por Rocha *et al.* (2009) é essencialmente uma versão discreta

⁶A versão supervisionada considera os marcadores como sendo um conjunto de pixels rotulados fornecidos pelo usuário.

⁷<http://hal.archives-ouvertes.fr/hal-00869727/>

⁸No *Watershed* por paradigma de queda de chuva (*drop of water*), considera-se a imagem como um relevo topográfico, no qual a intensidade dos pixels indica a altitude dos pontos no relevo. Para cada pixel da imagem, considera-se

do *Mean Shift* baseado em grafos no espaço de características, porém com cálculo explícito das densidades nos nós do grafo, e usando a versão dual da IFT-*Watershed* que segue o paradigma por imersão, de modo a tirar proveito de toda a literatura da comunidade de Morfologia Matemática. De forma similar, outro método popular chamado *Quick Shift* cria uma árvore de pontos de dados de vizinhos mais próximos, que aumentam o valor de densidade para atingir um domo de uma função de densidade. Ambos os métodos já foram usados para a geração de superpixels (Fulkerson *et al.*, 2009b). No entanto, métodos executados no espaço de características não resolvem os problemas de imagens com forte inomogeneidade.

Os métodos de segmentação baseados em grafos abordados por Felzenszwalb e Huttenlocher (2004) e Shi e Malik (2000) também podem ser usados para a extração de superpixels. O método de Felzenszwalb e Huttenlocher (2004) usa árvores de Espalhamento Mínimo (*Minimum Spanning Tree*) enquanto Shi e Malik (2000) se baseia em cortes normalizados. No entanto, têm-se observado que Felzenszwalb e Huttenlocher (2004) produz superpixels com formas e tamanho muito irregulares e (Shi e Malik, 2000) é um dos métodos mais lentos para a extração de superpixels.

Outro algoritmo não especializado usado para superpixels é o clássico *Watershed* por Beucher e Meyer (1993). Como o próprio nome sugere, a ideia é criar várias bacias hidrográficas através da simulação de um processo de inundação começando pelos mínimos locais do gradiente da imagem (Lotufo *et al.*, 2002). Cada bacia hidrográfica representa um componente conexo na segmentação, o que, consequentemente, representa um superpixel também. O problema com o algoritmo acima descrito é que o mesmo não oferece nenhuma maneira de controlar diretamente o tamanho ou compacidade dos superpixels, violando as propriedades P.2 e P.5. No entanto, a propriedade P.2 poderia ser resolvida pelo uso de valores de extinção calculados através da análise de uma árvore de componentes (Silva e Lotufo, 2011). Outros métodos tratam o agrupamento de dados como uma Floresta de Caminhos ótimos (*Optimum-Path Forest*) (Rocha *et al.*, 2009). Isto corresponde a uma definição dual da IFT-*Watershed* (Lotufo *et al.*, 2002), mas rodando em um grafo diferente e começando pelos valores máximos locais de uma função de densidade.

Outros autores concentram-se especificamente na produção de superpixels, Levinstein *et al.* (2009) apresenta um algoritmo baseado em Fluxo Geométrico (*Geometric-Flow*) chamado Turbopixel. Esse algoritmo organiza seus superpixels em uma estrutura semelhante a uma grade. Superpixels são gerados por uma curva de evolução de um conjunto de pontos de sementes colocadas regularmente na imagem. Usando algumas restrições, esse processo obtém superpixels que preenchem todas as propriedades de um superpixel. No entanto, de acordo com Achanta *et al.* (2012), o método Turbopixel é um dos algoritmos mais lentos examinados e exibe relativamente uma fraca adesão às bordas. Alguns autores Moore *et al.* (2008), Moore *et al.* (2009) e Moore *et al.* (2010), geram superpixels em uma determinada ordem geométrica que cria uma grade regular real. A vantagem de ter uma grade é que os superpixels gerados têm a mesma relação aos seus vizinhos como simples pixels, simplificando a sua adaptação aos métodos que tomam vantagens na análise de vizinhança. Essa estrutura de grade é diferente da mostrada por Levinstein *et al.* (2009) que carece de uma vizinhança bem definida.

A título de exemplo, é importante notar que os dois tipos de problemas mencionados anteriormente (transições em forma de rampa, e inomogeneidade de campo) são frequentes em imagens de RM do cérebro humano. A inomogeneidade aumenta a sobreposição entre os padrões de intensidade de objeto e fundo, no espaço de características, dificultando a segmentação de tecidos e demais estruturas cerebrais. Nesse contexto, técnicas específicas de correção de inomogeneidade podem ajudar a amenizar o problema (Axel *et al.*, 1987; Brinkmann *et al.*, 1998; Sled *et al.*, 1998), porém

uma gota de água caindo sobre o relevo naquele ponto. A água flui ao longo de um caminho até chegar a algum mínimo local. Pontos associados a um mesmo mínimo local recebem um mesmo rótulo. De modo similar no *Mean Shift*, cada amostra se desloca pelo espaço de características, trocando a sua posição pela média dos pontos vizinhos mais próximos, o que tem o mesmo efeito de se deslocar para regiões de mais alta densidade de pontos para um kernel correspondente apropriado. Cada amostra fica associada a um máximo local da densidade. Esses métodos são versões duais no discreto e no contínuo, respectivamente, em que o inverso da intensidade dos pixels atua de modo análogo a densidade no *Mean Shift*, e vice-versa.

essas correções nunca são perfeitas⁹. Tendo as NMIF demonstrado bons resultados na segmentação supervisionada dessas imagens RM de 3 Tesla (3T) (Mansilla *et al.*, 2013a), pretende-se estender esses resultados para o caso não supervisionado.

1.3 Segmentação de Objetos

O uso de superpixels normalmente ocorre como uma etapa de pré-processamento para depois ser usado em algum contexto de uma aplicação de mais alto nível. Destas aplicações, muitas possuem uma complexidade elevada, impossibilitando o seu uso em tempo real. Muitas vezes, essa complexidade se origina de sua execução ao nível de pixel. Mesmo com imagens pequenas, o número de pixels continua elevado. Por exemplo, para uma imagem de dimensões 250×250 , 62500 pixels teriam que ser avaliados.

O superpixel surge então como uma solução interessante para esse problema sem que haja a necessidade de grandes alterações na aplicação original, apenas o seu uso já possibilitaria uma redução considerável no número de “pixels” avaliados. Além disso, diferentemente de um pixel que é apenas uma representação não natural no espaço discreto desprovido de qualquer informação de contexto, um superpixel, por sua vez, sempre representará um objeto ou parte de um objeto contido na imagem em função de informações disponíveis para sua geração, textura, intensidade, contorno, cores, etc.

Ren e Malik (2003) aproveitam essas vantagens no desenvolvimento de seu modelo de classificação para segmentações. Além de minimizar a complexidade do modelo, o seu uso também facilita na sua etapa de agrupamento para a formação dos segmentos finais da segmentação nos quais várias características são retiradas diretamente dos superpixels como a similaridade intra e inter-regional de textura, a similaridade intra e inter-regional de luminosidade e continuidade curvilínea.

Já Mori *et al.* (2004) apresenta um trabalho que ataca o desafio da segmentação de pessoas. Diferentemente de outros trabalhos semelhantes que simplificam esse problema com métodos de mais baixo nível como segmentações a nível de pixel e subtração de fundo. Nesse artigo, informações de contexto são usadas como auxílio na segmentação. Essas informações constituem a separação do corpo humano em pequenas partes, como mão, antebraço, tronco, perna, etc e suas configurações individuais. A detecção correta de uma mão significa que na sua vizinhança haverá um antebraço, permitindo uma segmentação correta em cenários com mais ruídos de fundo e poses diversas.

Esse método começa com uma segmentação normal a nível de pixel usando *Normalized Cuts* (Shi e Malik, 2000) de uma imagem de borda usando *Canny* (Canny, 1986) a partir de uma imagem de entrada. Dessa segmentação, possíveis partes do corpo são detectadas e são então usadas para a geração de possíveis configurações parciais do corpo segundo certas regras como escala relativa e simetria das roupas.

O próximo passo é a detecção das partes faltantes e validação das configurações parciais para a geração final da configuração completa, e, consequentemente, sua segmentação. Para a detecção das partes faltantes, as partes já detectadas da configuração são usadas. Para cada parte, uma máscara em formato de retângulo é posicionada em cada uma das juntas dela e estendida em todas as direções até a detecção da parte vizinha. Ou seja, sendo que uma mão foi detectada na configuração parcial, a máscara estendida a partir de sua junta teria o papel de detectar o antebraço. Com o antebraço detectado, a próxima parte a ser procurada seria o resto do braço e assim sucessivamente.

Essa etapa deve ser rodada para todas as partes para cada uma das configurações parciais,

⁹Estudos com imagens de 3T mostram que máscaras iniciais e cuidadosas de cérebro são necessárias para melhorar os resultados da correção de inhomogeneidade (Boyes *et al.*, 2008). No entanto, experimentos com imagens de 3T demonstram que as máscaras obtidas por técnicas de *Skull Stripping* são por sua vez também afetadas pela inhomogeneidade de campo (Cappabianco *et al.*, 2012), sendo difícil quebrar esse ciclo de dependências. Ademais, experimentos mostram que métodos como, *Homogeneous Unsharp Masking* e *Nonparametric Nonuniform Intensity Normalization*, causam alterações indevidas em imagens sem problemas de inhomogeneidade, enquanto que o comportamento esperado nesse caso para um método ideal de correção seria conservar a imagem inalterada (Banerjee e Maji, 2013). Também muitas vezes a avaliação é conduzida usando apenas experimentos com imagens sintéticas (Banerjee e Maji, 2013), que não refletem bem os problemas em imagens reais.

resultando em uma busca muito complexa e ineficiente. A solução encontrada pelo autor foi no uso de superpixels. Enquanto as etapas anteriores são executadas, um mapa de superpixels é obtido em paralelo. Quando a etapa de detecção de partes faltantes e validação das configurações parciais ocorre, ao invés de aplicar a máscara em todas as direções possíveis, a mesma só é aplicada em cada um dos superpixels vizinhos desta parte, reduzindo drasticamente a complexidade geral do método sem a perda de acurácia.

[Kostolansky \(2016\)](#) apresenta uma aplicação de alto nível utilizando superpixels com o objetivo de detecção do céu em imagens. Seu algoritmo começa com a geração de superpixels da imagem de entrada. Um processo de mescla de superpixels é aplicado gerando uma segmentação. Os superpixels são mesclados a partir da distância dos superpixels vizinhos com um limiar calculado. Por fim, o maior segmento resultante presente na primeira linha da imagem é considerado como sendo o céu.

Diferente das outras aplicações abordadas acima, [Kostolansky \(2016\)](#) utiliza o SLIC como gerador de superpixels. Além disso, seu algoritmo é simples e de fácil implementação. Esses pontos o tornam um ótimo candidato para uma melhor análise das vantagens do arcabouço em comparação com o SLIC no escopo de aplicações de alto nível. Essa análise e detalhamento da aplicação são apresentadas na Seção 5.6.

1.4 Organização do Trabalho

No Capítulo 1.2 são apresentados trabalhos da literatura sobre segmentação não supervisionada, superpixels e suas características. Conceitos básicos sobre imagem digital e grafos necessários à pesquisa são discutidos no Capítulo 2. No Capítulo 3 são discutidos os métodos relacionados diretamente com o projeto. Já no Capítulo 4 é apresentado o método desenvolvido IFT-SLIC e seus detalhes de implementação. Os experimentos, contribuições e conclusões estão contidos no Capítulo 5.

Capítulo 2

Conceitos Gerais

Nesta seção são apresentadas as noções básicas necessárias para o correto entendimento desta pesquisa.

2.1 Imagem Digital

Uma imagem digital \hat{I} é representada por um par (D_I, \vec{I}) em que D_I é o domínio da imagem que corresponde a um conjunto de pontos contidos em Z^n , sendo n o número de dimensões da imagem. Cada elemento $t \in D_I$ pode ser denominado pixel (*picture element*) ou, mais genericamente, spel (*space element*) e \vec{I} é um mapeamento vetorial que associa a cada pixel $t \in D_I$ um conjunto de k valores inteiros $\vec{I}(t) = (I_1(t), I_2(t), \dots, I_k(t))$ representando alguma propriedade física, sendo k o número de bandas.

Uma imagem 2D, $\hat{I} = (D_I, \vec{I})$, em tons de cinza possui apenas uma banda, ou seja, $\forall t \in D_I$, $\vec{I}(t) = (I_1(t))$, portanto a mesma pode ser interpretada como $\hat{I} = (D_I, I)$. $I(t)$ ou $I_1(t)$ contém alguma propriedade física de cada pixel $t \in D_I$. Essa propriedade pode ser obtida através de amostragem e quantização de uma função contínua $I_c(x, y)$. Exemplos de $I(t)$ são brilho ou intensidade, no caso de uma foto monocromática e densidade de tecido, no caso de tomografias.

Uma imagem em tons de cinza pode ser representada através de uma matriz de tamanho $N \times M$, na qual valores altos são demonstrados visualmente por tons tendendo ao branco e valores baixos por tons tendendo ao preto.

Embora o valor de um pixel seja normalmente um inteiro, certas transformações matemáticas requerem um armazenamento de números reais ou complexos. Apesar disso, no que tange a sua visualização, esses valores são requantizados em valores inteiros. Um exemplo de transformação matemática geradora de uma imagem com pixels de números complexos é a Transformada de Fourier ([Fourier, 1822](#)). Sua transformada $\mathcal{F}(\hat{I})$ de uma imagem em tons de cinza $\hat{I} = (D_I, I)$ gera uma imagem complexa $\hat{F} = (D_I, \vec{F})$, na qual $\vec{F} = (F_1, F_2)$ contém a parte real e imaginária.

2.1.1 Multidimensionalidade

Uma imagem pode ser bidimensional ou multidimensional. No caso de uma imagem bidimensional, seu domínio $D_I \subset Z^2$ é definido pelas coordenadas $x = 0, 1, \dots, n_x - 1$ e $y = 0, 1, \dots, n_y - 1$.

Imagens multidimensionais têm seu domínio definido como $D_I \subset Z^n$, para $n > 2$. Uma imagem tridimensional é um exemplo comum de imagem multidimensional em que $n = 3$ com o objetivo de representar uma região volumétrica, suas coordenadas são $x = 0, 1, \dots, n_x - 1$, $y = 0, 1, \dots, n_y - 1$ e $z = 0, 1, \dots, n_z - 1$ e cada elemento $p = (x, y, z) \in D_I$ é denominado voxel (*volumetric pixel*). Imagens médicas exemplificam o caso de multidimensionalidade, uma sequência espacial de fatias tomográficas é um exemplo de imagem tridimensional. Adicionando a noção temporal nessa sequência obtém-se uma imagem tetradimensional e, em ambos os casos, a propriedade física quantizada é a densidade dos tecidos na tomografia de Raios-X.

2.1.2 Imagem Multi-banda

Uma imagem $\hat{I} = (D_I, \vec{I})$ é multi-banda (também conhecida como multi-paramétrica ou multi-espectral) quando para cada pixel $t \in D_I$, o número de propriedades físicas quantificadas é maior que 1, ou seja, $\vec{I}(t) = (I_1(t), I_2(t), \dots, I_k(t))$, para $k > 1$.

2.1.3 Imagem de Rótulos

Dada uma imagem $\hat{I} = (D_I, \vec{I})$ qualquer, uma imagem de rótulos de \hat{I} é definida como uma imagem de tons de cinza $\hat{R} = (D_I, R)$ na qual para cada elemento $t \in D_I$, há um valor inteiro $R(t)$ associado a t denominado rótulo. Uma máscara é um exemplo de uma imagem binária de rótulos em que pixels brancos representam a parte de interesse de uma imagem enquanto pixels pretos representam o fundo (a parte a ser ignorada). Outro exemplo é uma segmentação de \hat{I} em várias regiões, nas quais cada parte possui um identificador único na imagem de rótulos.

2.1.4 Objeto

Um objeto \mathcal{O} é um subconjunto de pixels de D_I de uma imagem $\hat{I} = (D_I, \vec{I})$, composto por um ou mais componentes conexos. A borda $\mathcal{B} \subseteq \mathcal{O}$ de um objeto é formada pelo subconjunto de elementos de \mathcal{O} que possuem ao menos um pixel adjacente no exterior ($D_I \setminus \mathcal{O}$).

2.1.5 Segmentação e Superpixels

Uma imagem \hat{I} pode ser dividida em várias partições $\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_k$ tal que $\bigcup_{j=1}^k \mathcal{P}_j = D_I$. Quando essas partições visam representar diferentes objetos da imagem obtém-se uma segmentação de \hat{I} .

As partições \mathcal{P}_j de uma segmentação são normalmente referenciadas como superpixels quando seus algoritmos de geração tentam se adequar as propriedades **P.1** à **P.5** apresentadas anteriormente.

2.2 Grafos

Um grafo G é representado pelo par (V, A) , no qual V é um conjunto finito não vazio composto de nós e A por sua vez é um conjunto de pares de elementos distintos de V denominados arestas. $|V|$ representa o número total de nós no grafo e $|A|$ o número total de arestas.

2.2.1 Relação de Adjacência

Um nó v é adjacente a um nó u caso exista um arco $(u, v) \in A$. Essa relação de adjacência entre nós pode ser assimétrica, v pode ser adjacente a u sem que u seja adjacente a v , ou simétrica, $(u, v) \in A$, se e somente se $(v, u) \in A$.

Uma relação de adjacência \mathcal{A} é uma relação binária entre nós em um grafo $G = (V, A)$ determinando seus arcos, ou seja, cada nó $u \in V$ tem associado a si um conjunto de todos os nós v adjacentes, definindo uma vizinhança representada por $\mathcal{A}(u) = \{v \in V \text{ tal que } (u, v) \in A\}$, logo, $v \in \mathcal{A}(u)$ indica que um nó v é adjacente a u em G .

2.2.2 Subgrafo

Um grafo $G_2 = (V_2, A_2)$ é um subgrafo de $G_1 = (V_1, A_1)$ quando $V_2 \subseteq V_1$ e $A_2 \subseteq A_1$. Se $V_2 = V_1$, então G_2 é chamado de subgrafo gerador ou subgrafo de espalhamento.

2.2.3 Dígrafo

Um grafo $G = (V, A)$ é chamado dígrafo quando o seu conjunto de arestas A possui pares ordenados de elementos de V , ou seja, as arestas (u, v) e (v, u) são consideradas distintas, representando uma relação de direção ou orientação entre os pares de A de tal forma que (u, v) significa uma direção $u \rightarrow v$. Assim, uma aresta $e = (u, v)$ é dita divergente de u e convergente a v , enquanto que em grafos não direcionados, a aresta e é dita incidente a u e v . Arestras de um dígrafo são chamadas de arcos.

Todos os métodos, que utilizam grafos apresentados neste trabalho, são dígrafos, portanto, a partir desta sessão, qualquer menção a grafos significa que o mesmo é um dígrafo.

2.2.4 Grafo Ponderado

Um grafo $G = (V, A)$ que possui um peso $w(u, v)$ associado a cada arco é dito ponderado. Um grafo ponderado pode ter uma relação de adjacência assimétrica caso $\forall(u, v) \in A, w(u, v) \neq w(v, u)$ e simétrica caso $\forall(u, v) \in A, w(u, v) = w(v, u)$.

2.2.5 Caminho

Para um dado grafo $G = (V, A)$, um caminho $\pi_t = \langle t_0, t_1, \dots, t_n = t \rangle$ é uma sequência de nós adjacentes tal que $(t_{i-1}, t_i) \in A$ para $1 \leq i \leq n$ com início em t_0 e término em t , onde n é o comprimento do caminho, o qual é formado por $n + 1$ nós e n arcos $(t_0, t_1), (t_1, t_2), \dots, (t_{n-1}, t_n)$. Um caminho é trivial quando $\pi_t = \langle t \rangle$.

Caso todos os nós de um caminho sejam distintos, esse caminho é um caminho simples.

Um caminho $\pi_t = \pi_s \cdot \langle s, t \rangle$ indica a concatenação de um caminho π_s por um arco (s, t) com as duas instâncias de s se fundindo em uma.

Quando é pretendido indicar expressamente a origem de um caminho, a notação $\pi_{s \rightsquigarrow t} = \langle t_0 = s, t_1, \dots, t_n = t \rangle$ pode também ser usada, em que s indica a origem e t o nó de destino.

Um nó t é dito conexo a um nó s quando há um caminho de s até t no grafo $G = (V, A)$.

2.2.6 Ciclo

Dado um caminho qualquer $\langle t_0, t_1, \dots, t_n \rangle$ em um grafo com relação de adjacência simétrica, caso $t_0 = t_n$ e $n \geq 2$, esse caminho apresenta um ciclo. Um ciclo é considerado simples caso nenhum nó for repetido, exceto pelo primeiro e último nó. Um grafo é denominado acíclico caso o mesmo não contenha nenhum ciclo simples.

2.2.7 Grafo Conexo

Um grafo G com relação de adjacência simétrica é dito conexo caso exista um caminho simples entre cada par possível de nós em G , caso contrário, o grafo é desconexo. Caso um grafo G com relação de adjacência simétrica contenha subgrafos conexos, o conjunto desses subgrafos é denominado componentes conexos.

2.2.8 Corte no Grafo

Dado um grafo $G = (V, A)$ com relação de adjacência simétrica, um corte $(V', V - V')$ de G é uma partição de V . Um arco $(u, v) \in A$ cruza um corte $(V', V - V')$ caso um de seus nós pertença a V' e o outro a $V - V'$.

2.2.9 Árvore e Floresta

Um grafo G é considerado uma árvore caso o mesmo tenha uma relação de adjacência simétrica, seja conexo e acíclico.

Uma floresta é um conjunto de árvores. Um grafo com relação de adjacência simétrica e acíclico, sendo ou não conexo, também é uma floresta.

2.2.10 Imagem como Grafo

Uma imagem $\hat{I} = (D_I, \vec{I})$ pode ser representada por um grafo $G = (V, A)$, na qual os nós em V correspondem aos pixels em D_I , e seus arcos em A correspondem a uma relação binária entre os pixels, dadas por uma relação de adjacência \mathcal{A} tal que $(u, v) \in A$ se $v \in \mathcal{A}(u)$.

Essa relação de adjacência \mathcal{A} depende das posições relativas dos pixels, e opcionalmente, de outras propriedades locais da imagem. Um exemplo de relação de adjacência é a circular, na qual os pixels adjacentes a um pixel u são definidos por $\forall v \in V, v \in \mathcal{A}(u)$ se $d(u, v) \leq \rho$ e $u \neq v$, em que d é a distância euclidiana e ρ uma constante.

Na Figura 2.1 são mostrados exemplos da relação de adjacência circular, além de outras relações como:

- Retangular: $\forall v \in V, v \in \mathcal{A}(u)$ se $|x_v - x_u| \leq \frac{a}{2}$ e $|y_v - y_u| \leq \frac{b}{2}$, em que a e b são os comprimentos dos 2 lados do retângulo com centro em (x_u, y_u) ;
- Conjuntos: $\forall v \in V, v \in \mathcal{A}(u)$ se $v - u \in \{(-1, -1), (1, -1)\}$;
- Posição relativa e propriedades locais: $\forall v \in V, v \in \mathcal{A}(u)$ se $d(u, v) \leq \rho_i$ e $d(\vec{I}(u), \vec{I}(v)) \leq \rho_a$, em que $d()$ é o operador de distância Euclidiana e ρ_i e ρ_a são os raios nos espaços de imagem e de atributos, respectivamente.

Outro caso de posição relativa e propriedades locais seria $\forall v \in V, v \in \mathcal{A}(u)$ se $d(u, v) \leq \rho_i$ e v estiver entre os k -vizinhos mais próximos de u no espaço de atributos, em que k é menor que o número de pixels no raio ρ_i .

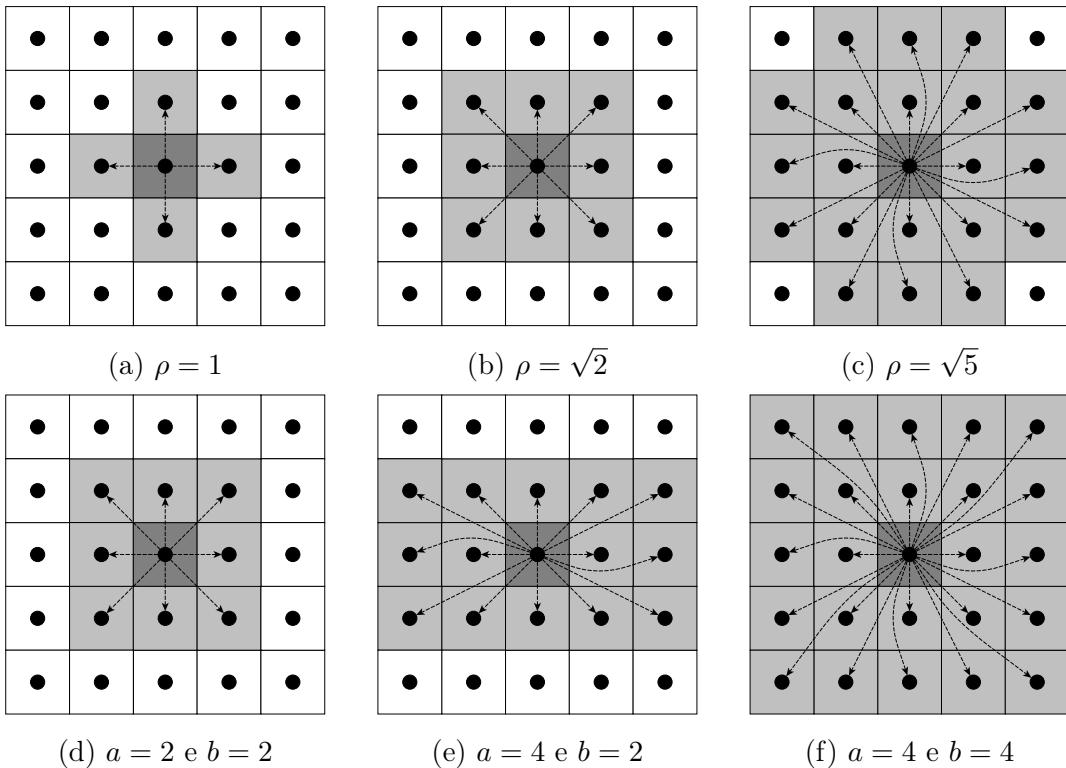


Figura 2.1: (a-c) Exemplos de adjacência circular. (d-f) Exemplos de adjacência retangular.

2.3 Avaliação de Métodos de Segmentação

Um método pode ser avaliado de diversas formas dependendo de sua aplicação e contexto. Formas comuns de analisar um método são a partir de sua complexidade computacional, tempo de execução, acurácia, entre outros. No caso da acurácia, o objetivo é descobrir o grau de proximidade entre um resultado obtido e o valor correto (gabarito).

O gabarito da segmentação de um objeto em uma imagem corresponde a uma imagem de rótulos, na qual os pixels brancos representam o objeto de interesse corretamente segmentado e os pixels pretos o resto ou fundo. Geralmente essa segmentação é obtida manualmente. Uma medida de acurácia normalmente possui um intervalo entre $[-1, 1]$ ou $[0, 1]$, em que -1 ou 0 respectivamente representam a pior acurácia possível e 1 a melhor acurácia.

Diversas medidas de acurácia podem ser encontradas na literatura, porém não há uma medida universal que seja superior às outras em todos os contextos possíveis (Stehman, 1997). Para este trabalho, o coeficiente *Dice* (Dice, 1945), também conhecido como *F-score* ou *F-measure* (Labatut e Cherifi, 2012; Rauber *et al.*, 2013; Rijsbergen, 1979), coeficiente de similaridade *Sørensen* (Sørensen, 1948) e índice de acurácia média de *Helldén* (Helldén, 1980), é usado como medida de acurácia na avaliação dos métodos de segmentação de imagens.

O coeficiente de *Dice* é representado pela seguinte relação:

$$D = \frac{2|X \cap Y|}{|X| + |Y|} \quad (2.1)$$

ou

$$D = \frac{2|X \cap Y|}{|X \cap Y| + |X \cup Y|}, \quad (2.2)$$

em que X representa o conjunto de pixels rotulados como objeto segmentado pelo método analisado e Y representa os pixels rotulados como objeto do gabarito.

Ao analisar uma segmentação X de uma imagem com seu gabarito Y , caso $X \neq Y$, logo a segmentação Y conterá pixels rotulados incorretamente, ou seja, apresentará erros de segmentação. Essa rotulação incorreta pode ser denominada como Falsos Positivos (*False Positive* FP) ou Falsos Negativos (*False Negative* FN), representando pixels positivos na segmentação, porém, negativos no gabarito e pixels negativos na segmentação, porém, positivos no gabarito, respectivamente.

Por outro lado, para pixels corretamente rotulados a denominação é Verdadeiros Positivos (*True Positive* TP) para pixels do objeto corretamente segmentados e Verdadeiros Negativos (*True Negative* TN) para pixels do fundo corretamente segmentados.

A Tabela 2.1 apresenta uma matriz de confusão como uma forma de visualização desses conjuntos, na qual, para um pixel positivo no gabarito e na segmentação, significa um TP, se o mesmo for negativo na segmentação, têm-se um FN, já se o pixel no gabarito e na segmentação for negativo, têm-se um TN, e FP, caso o mesmo seja positivo na segmentação.

		Imagen Segmentada	
		Objeto	Fundo
Gabarito	Objeto	TP Corretamente segmentado como objeto	FN Incorretamente segmentado como fundo
	Fundo	FP Incorretamente segmentado como objeto	TN Corretamente segmentado como fundo

Tabela 2.1: Matriz de confusão

O coeficiente de *Dice* pode ser reescrito baseado na matriz de confusão da seguinte forma:

$$D = \frac{2|TP|}{2|TP| + |FN| + |FP|}. \quad (2.3)$$

A Figura 2.2 apresenta um exemplo de gabarito (Figura 2.2 (b)) para uma imagem de entrada (Figura 2.2 (a)), a Figura 2.2 (c) representa uma segmentação obtida por algum método qualquer, e a Figura 2.2 (d) demonstra os conjuntos de pixels corretamente e incorretamente rotulados ao comparar a segmentação ao gabarito. Usando a Figura 2.2 como exemplo, pode-se calcular o seu coeficiente *Dice* para obter sua acurácia da seguinte forma:

$$\begin{aligned} D &= \frac{2|X \cap Y|}{|X| + |Y|} \\ D &= \frac{2(24)}{31 + 30} \\ D &= 0.786885 \end{aligned} \quad (2.4)$$

Além do *Dice*, foi usado também a medida de Compacidade *CO* (Schick *et al.*, 2012). A compacidade de um superpixel é severamente impactada pela regularidade de sua borda e pelas suas mudanças de orientação. Sua medida pode ser descrita pela seguinte equação:

$$CO = \sum_{i=1}^k Q_{P_i} \frac{|P_i|}{N}, \quad (2.5)$$

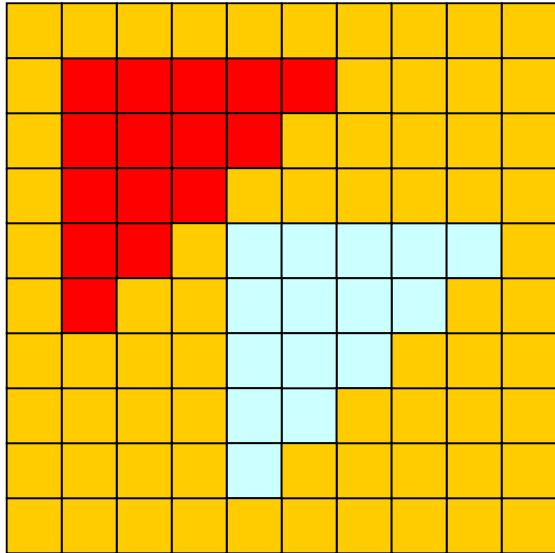
em que $|P_i|$ denota a área do superpixel P_i , N é o número total de pixels da imagem, k é o número total de superpixels e Q_{P_i} denota o quociente isoperimétrico para o superpixel P_i definido como:

$$Q_{P_i} = \frac{|P_i|}{|\mathcal{C}|} = \frac{4\pi|P_i|}{Peri(P_i)^2}, \quad (2.6)$$

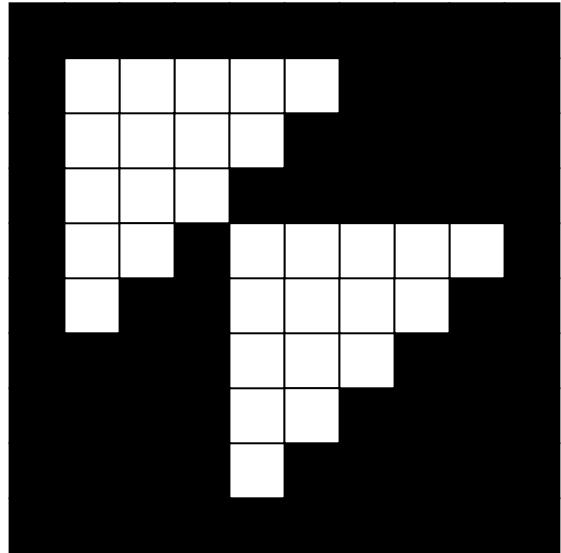
em que $|\mathcal{C}|$ é a área de um círculo \mathcal{C} ¹ de mesmo perímetro que P_i e $Peri(P_i)$ ² denota o perímetro de P_i .

¹O raio r do círculo \mathcal{C} é obtido por $r = \frac{Peri(P_i)}{2\pi}$.

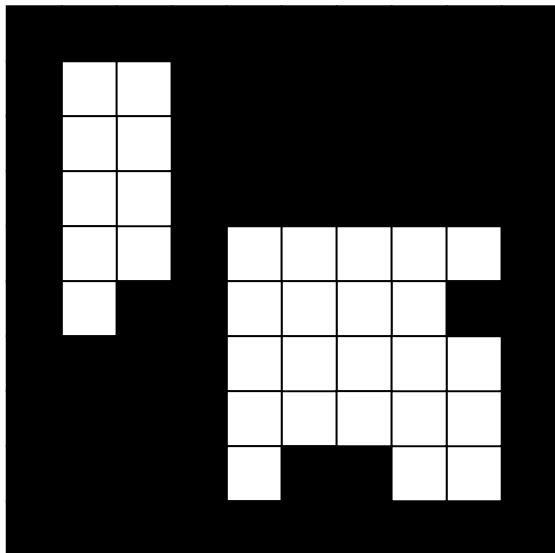
²Para o cálculo do perímetro de P_i , $Peri(P_i)$, o método sugerido por Kiryati e Székely (1993) para melhor aproximar o tamanho do contorno no espaço discreto foi usado.



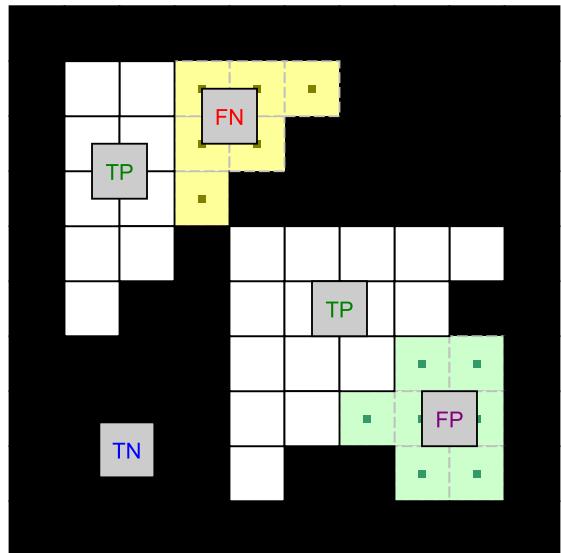
(a) Imagem de entrada



(b) Gabarito



(c) Segmentação



(d) Classificação

Figura 2.2: (a-b) Imagem de entrada e seu gabarito. (c) Um exemplo de segmentação por um algoritmo qualquer. (d) A classificação dos pixels da segmentação entre pixels do objeto e fundo corretamente segmentados, TP e TN respectivamente, e entre pixels do objeto e fundo incorretamente segmentados, FN e FP respectivamente.

Capítulo 3

Conceitos Específicos

Nesta seção são apresentados os métodos e noções mais específicas usadas na pesquisa.

3.1 Função de Conexidade

Uma função de conexidade é um mapeamento que atribui a cada caminho π_t em um grafo $G = (V, A)$ um custo $f(\pi_t)$ com base em propriedades da imagem ao longo desse caminho, sendo elas tanto locais (intensidade, posição, etc), quanto globais (textura, geometria, etc).

Um caminho $\pi_t = \langle t_0, t_1, \dots, t_n \rangle$ é ótimo se $f(\pi_t) \leq f(\tau_t)$ para qualquer outro caminho τ_t em π_G , em que π_G denota o conjunto de todos os caminhos em G . Caso todos os caminhos $\pi_{t_i} = \langle t_0, t_1, \dots, t_i \rangle$, em que $i = 0, 1, \dots, n$, sejam ótimos, então π_t é denominado caminho ótimo completo. Ao tomar para cada pixel $t \in V$ um caminho ótimo com pixel terminal t , obtém-se o custo do caminho ótimo $C(t)$, que é definido como:

$$C(t) = \min_{\forall \pi_t \text{ em } \pi_G} \{f(\pi_t)\} \quad (3.1)$$

O custo de um caminho trivial $\pi_t = \langle t \rangle$ é normalmente obtido por um valor de custo inicial $H(t)$, enquanto que as funções de conexidade para caminhos não triviais seguem uma regra de extensão de caminho. Por exemplo:

$$\begin{aligned} f_{\max}(\langle t \rangle) &= H(t), \\ f_{\max}(\pi_s \cdot \langle s, t \rangle) &= \max\{f_{\max}(\pi_s), w(s, t)\}, \end{aligned} \quad (3.2)$$

$$\begin{aligned} f_{sum}(\langle t \rangle) &= H(t), \\ f_{sum}(\pi_s \cdot \langle s, t \rangle) &= f_{sum}(\pi_s) + w(s, t), \end{aligned} \quad (3.3)$$

$$\begin{aligned} f_{euc}(\langle t \rangle) &= H(t), \\ f_{euc}(\pi_{r \rightsquigarrow s} \cdot \langle s, t \rangle) &= \|t - r\|^2, \end{aligned} \quad (3.4)$$

em que $w(s, t) \geq 0$ é um peso fixo do arco e r o primeiro pixel do caminho $\pi_{r \rightsquigarrow s}$.

Funções como f_{\max} , f_{sum} e f_{euc} atribuem o custo inicial $H(t)$ como:

$$H(t) = \begin{cases} 0 & \text{se } t \in S \\ +\infty & \text{caso contrário,} \end{cases} \quad (3.5)$$

em que $S \subseteq V$ é o conjunto de pixels sementes.

3.1.1 Função Monotonicamente Incremental

Uma função monótona é uma função entre 2 conjuntos ordenados que preserva ou inverte a sua relação de ordem (relação binária como maior, menor, etc). Uma função é denominada Monotonicamente Incremental (*Monotonically Increasing MI*) quando ela obedece as seguintes equações:

$$\begin{aligned} f_m(\langle t \rangle) &= H(t), \\ f_m(\pi_s \cdot \langle s, t \rangle) &= f_m(\pi_s) \odot (s, t), \end{aligned} \quad (3.6)$$

em que \odot é uma operação binária entre o custo de um caminho e um arco que satisfaz as condições

$$(M1) \quad f_m(\pi_s) \geq f_m(\tau_s) \Rightarrow f_m(\pi_s) \odot (s, t) \geq f_m(\tau_s) \odot (s, t),$$

$$(M2) \quad f_m(\pi_s) \odot (s, t) \geq f_m(\pi_s),$$

em que π_s e τ_s são caminhos no grafo $G = (V, A)$ e qualquer aresta $(s, t) \in A$. Uma característica essencial desse modelo de função é que \odot depende apenas do valor de custo do caminho π_s e não de qualquer outra propriedade desse caminho.

As funções f_{max} e f_{sum} são exemplos de funções MI, enquanto que f_{euc} não é uma função MI.

3.2 Mapa de Predecessores

Um mapa de predecessores é uma função P que atribui a cada nó t em V algum outro nó adjacente em V , ou um marcador nulo distinto $nil \notin V$, neste último caso t é dito ser uma raiz do mapa.

3.3 Floresta de Espalhamento

Uma Floresta de Espalhamento (*Spanning Forest SF*) é um mapa de predecessores que não contém ciclos, ou seja, aquele que leva cada nó à nil em um número finito de iterações. Para qualquer nó $t \in V$, uma SF P define um caminho π_t^P recursivamente como $\langle t \rangle$ se $P(t) = nil$, e $\pi_s^P \cdot \langle s, t \rangle$ se $P(t) = s \neq nil$. A Figura 3.1 apresenta um exemplo de SF contendo 3 árvores.

3.4 IFT

A IFT pode ser interpretada como uma generalização do algoritmo de Dijkstra (Cormen *et al.*, 2001) permitindo o uso de funções de conexidade mais gerais. Ela permite a redução de problemas de processamento de imagens baseados em conexidade ao cálculo de uma floresta de caminhos ótimos no grafo derivado da imagem.

Sendo a IFT um arcabouço facilmente modificável pela troca de funções de conexidade, isso possibilita que vários operadores de imagem sejam derivados de seu algoritmo geral, favorecendo implementações baseadas em hardware (Cappabianco *et al.*, 2007), facilitando a compreensão da relação entre esses operadores e permitindo possíveis extensões dos mesmos.

Além disso, a IFT é extremamente eficiente e simples, permitindo a implementação em tempo linear da maioria dos operadores de imagem disponíveis. Outras otimizações também são possíveis dependendo da aplicação e fáceis de implementar, como paralelismo (Bergo e Falcão, 2007) e cálculo diferencial (Falcão e Bergo, 2004). Ademais, os operadores de imagem são reduzidos à escolha de poucos parâmetros na IFT e a um processamento local de sua saída.

A IFT toma um grafo $G = (V, A)$ de uma imagem na qual os nós são os pixels da imagem e os arcos são os pares de pixels $(s, t) \in A$, e uma função de conexidade f ; e atribui um caminho ótimo π_t para cada nó $t \in V$ tal que uma floresta de caminhos ótimos P é obtida, ou seja, uma SF na qual

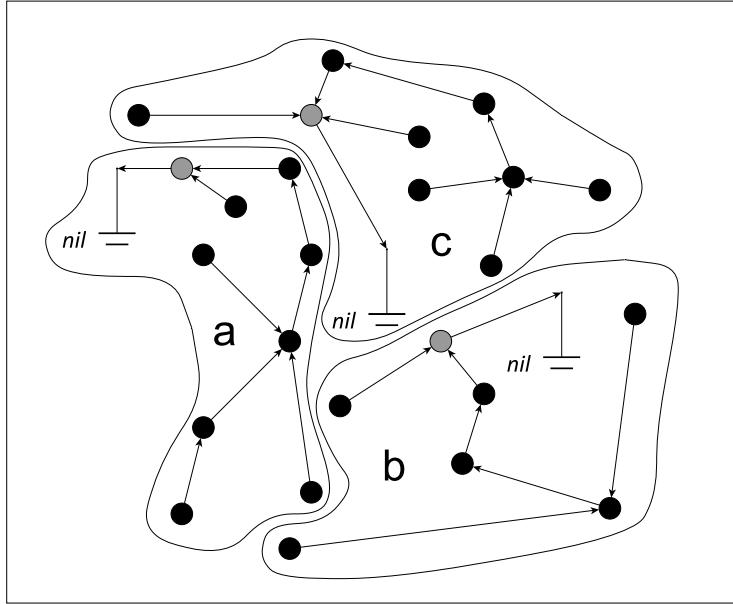


Figura 3.1: Uma SF com 3 árvores a , b e c .

todos os caminhos são ótimos. No entanto, f deve ser monotonicamente incremental (Falcão *et al.*, 2004), de outro modo, os caminhos podem não ser ótimos.

Para gerar uma SF de caminhos ótimos, a IFT aplica uma competição entre todas as suas raízes; cada raiz na IFT é considerada um mínimo local de f , com o objetivo de decidir qual raiz irá conquistar cada nó $t \in V$. Uma raiz só sucede na conquista de t caso ela seja a raiz mais fortemente conexa a ele. O resultado final é uma partição ótima da imagem que consequentemente representa uma segmentação de G .

As raízes utilizadas pela IFT podem ser obtidas de duas formas:

- **Supervisionada:** O usuário define as regiões de interesse do objeto a ser segmentado.

Nesse cenário normalmente existirão 2 grupos de sementes, o primeiro grupo S_o representando o objeto conterá nós posicionados dentro do mesmo e o outro grupo S_f representando o fundo conterá nós do fundo da imagem, tal que $S_o \cap S_f = \emptyset$.

A competição entre S_o e S_f pelos seus nós mais fortemente conexos resultará em duas SFs de caminhos ótimos, enraizada nas sementes de objeto e fundo respectivamente, gerando a segmentação final da imagem R definida por:

$$R(t) = \begin{cases} 1 & \text{se } \mathcal{R}(\pi_t^P) \in S_o \\ 0 & \text{caso contrário,} \end{cases} \quad (3.7)$$

em que π_t^P é um caminho ótimo com término em t obtido de P e $\mathcal{R}(\pi_t^P)$ é a sua raiz;

- **Não supervisionada:** Alguma regra é utilizada para definir quais serão as raízes utilizadas no algoritmo, por exemplo, pegando como raízes os mínimos regionais de uma imagem de entrada (como na transformada de *Watershed* via IFT) ou amostrando raízes uniformemente espalhadas pela imagem, tal como realizado neste trabalho.

Nesse caso, $S = \{S_1 \cup S_2 \cup \dots \cup S_n\}$, em que n pode variar entre 1 e $|V|$ e S_1, S_2, \dots, S_n . A segmentação final é definida por:

$$R(t) = x \text{ se } \mathcal{R}(\pi_t^P) \in S_x, \quad (3.8)$$

em que x é um valor entre 1 e n e π_t^P é um caminho ótimo com término em t obtido de P .

3.4.1 Ambiguidade

Seja $s_1 \in S_1$ e $s_2 \in S_2$ duas sementes com rótulos distintos que alcançam t com caminhos ótimos completos, $\pi_{s_1 \leadsto t}$ e $\pi_{s_2 \leadsto t}$ respectivamente. O nó t é chamado de nó de empate (Figura 3.3), já que a segmentação final difere dependendo da escolha desses caminhos. Nesse caso, pode-se obter duas possíveis florestas de caminhos ótimos gerando diferentes partições do conjunto V .

Assim, uma zona de empate é um conjunto maximal de nós de empate, o qual forma uma subárvore em alguma floresta de caminhos ótimos (Miranda e Falcão, 2009).

Políticas de Desempate

A estratégia adotada pela IFT e apresentada no Algoritmo 1 para resolver ambiguidades de florestas ótimas consiste em sempre escolher o primeiro caminho com o menor custo que alcance um nó qualquer t , ignorando qualquer outro caminho que alcance t com o mesmo custo posteriormente (linha 12).

Outra ambiguidade tratada pela IFT acontece quando há dois ou mais caminhos de custo ótimo que alcançam nós distintos, ou seja, seja $\pi_t = \pi_s \cdot \langle s, t \rangle$ e $\pi_b = \pi_a \cdot \langle a, b \rangle$, tal que $C(\pi_t) = C(\pi_b)$, sua ordem de processamento dependerá da política de desempate usada pela fila de prioridade Q .

No Algoritmo 1 a fila de prioridade Q é implementada com uma política de desempate Primeiro a Entrar, Primeiro a Sair (*First-In-First-Out* FIFO) (Figuras 3.2a e 3.2c), ou seja, o nó escolhido sempre será o primeiro da fila. Q também pode ser implementada com uma política Último a Entrar, Primeiro a Sair (*Last-In-First-Out* LIFO) (Figuras 3.2b e 3.2d). Nesse caso, o Algoritmo 1 necessitará de mudanças para sempre selecionar o último caminho ótimo encontrado ao invés do primeiro, trocando a comparação da estritamente menor para \leq na linha 12, preservando assim sua consistência no caso de empates.

3.4.2 Algoritmo

Pelo fato da IFT ser uma generalização do Dijkstra, logo o seu procedimento é similar ao mesmo, com mudanças para o suporte de fontes múltiplas e funções de conexidade mais gerais.

O primeiro laço do Algoritmo 1 serve para inicializar os mapas P , C , R e $T \forall s \in V$ (linhas 2-6), sendo que $C(s)$ conterá o valor do caminho trivial de s , $\mathcal{R}(s)$ conterá a si mesmo como raiz, $T(s)$ conterá *false* já que o nó s ainda não foi analisado e $P(s)$ será nulo já que nesta etapa s ainda não contém um predecessor. Na linha 6, todos os nós com valores em C diferentes de $+\infty$, ou seja, as raízes, são inseridas na fila Q .

No segundo laço (linha 7), são calculados os caminhos ótimos em P . A cada iteração, um nó $s \in Q$ mínimo é removido de Q na linha 8, sendo π_s^P um caminho ótimo e na linha 9 o valor de s em T é atualizado para *true* já que s está sendo verificado agora. As linhas 10-15 calculam e propagam caminhos melhores para a adjacência de s seguindo uma ordem não-decrescente de valores de custo com a seguinte regra de expansão: Se $f(\pi_s^P \cdot \langle s, t \rangle) < f(\pi_t^P)$ (linha 12), então π_t^P é trocado por $\pi_s^P \cdot \langle s, t \rangle$ (linha 14), isto é, caso o caminho $\pi_s^P \cdot \langle s, t \rangle$ tenha um custo menor que o caminho atual de π_t^P , $C(t)$, $P(t)$, $\mathcal{R}(t)$ e $R(t)$ são atualizados com o novo caminho.

3.4.3 Funcionamento

A IFT obtém o seu particionamento por meio da competição de suas raízes a partir do grafo. A IFT utiliza-se de um mapa de predecessores para definir a floresta calculada. A Figura 3.4

Algoritmo 1 IFT

Entrada:

Grafo $G = (V, A)$ da imagem $\hat{I} = (D_I, \vec{I})$
 Relação de adjacência \mathcal{A}
 Função de conexidade f
 Conjunto de sementes $S \subseteq V$

Auxiliares:

Fila Q de prioridade
 Variável $custo$
 Mapa T de nós já verificados

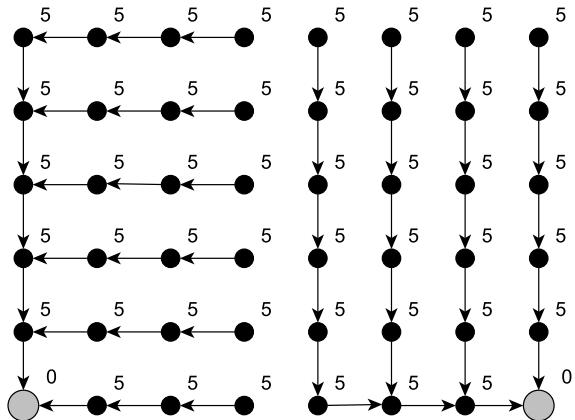
Saída:

Mapa P de predecessores
 Mapa C de custos mínimos
 Mapa \mathcal{R} de raízes
 Mapa R de rótulos

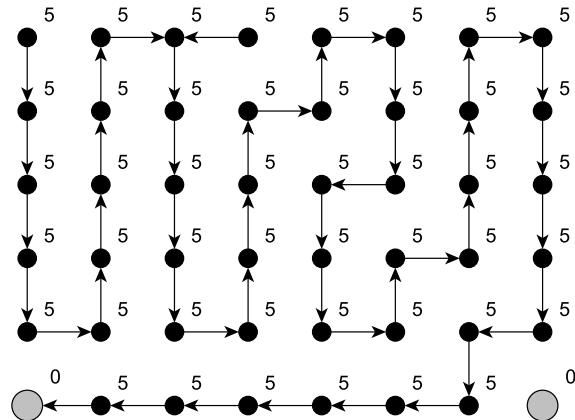
```

1: procedimento IFT
2:   para todo  $s \in V$  faça
3:      $P(s) \leftarrow nil$ ,  $C(s) \leftarrow f(\langle s \rangle)$ ,  $\mathcal{R}(s) \leftarrow s$  e  $T(s) \leftarrow false$ 
4:     se  $C(s) \neq +\infty$  então
5:       Insira  $s$  em  $Q$ 
6:        $R(s) \leftarrow x$  para  $s \in S_x$ 
7:   enquanto  $Q \neq \emptyset$  faça
8:     Remova  $s$  de  $Q$  cujo valor  $C(s)$  é mínimo
9:      $T(s) \leftarrow true$ 
10:    para todo  $t \in \mathcal{A}(s)$ , tal que  $T(t) = false$  faça
11:       $custo \leftarrow f(\pi_s^P \cdot \langle s, t \rangle)$ 
12:      se  $custo < C(t)$  então
13:        se  $C(t) \neq +\infty$  então Remova  $t$  de  $Q$ 
14:         $P(t) \leftarrow s$ ,  $C(t) \leftarrow custo$ ,  $\mathcal{R}(t) \leftarrow \mathcal{R}(s)$  e  $R(t) \leftarrow R(s)$ 
15:        Insira  $t$  em  $Q$ 

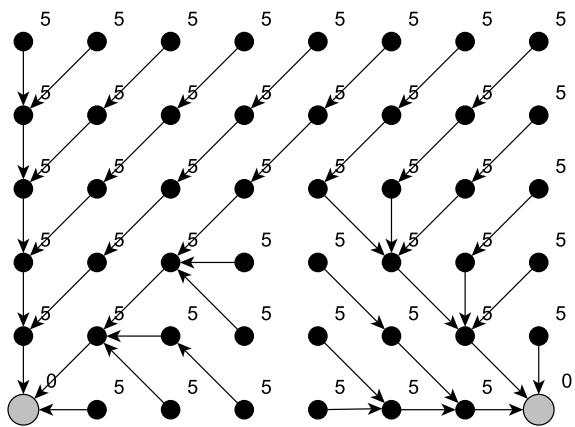
```



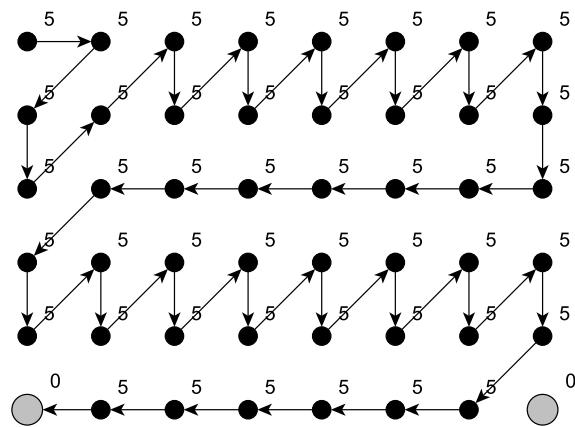
(a) Política FIFO com adjacência de 4 vizinhos



(b) Política LIFO com adjacência de 4 vizinhos



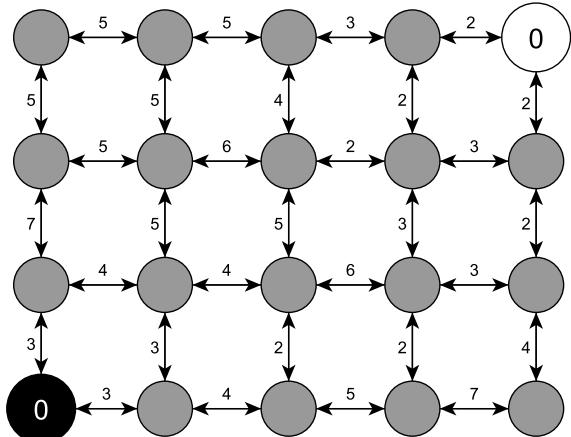
(c) Política FIFO com adjacência de 8 vizinhos



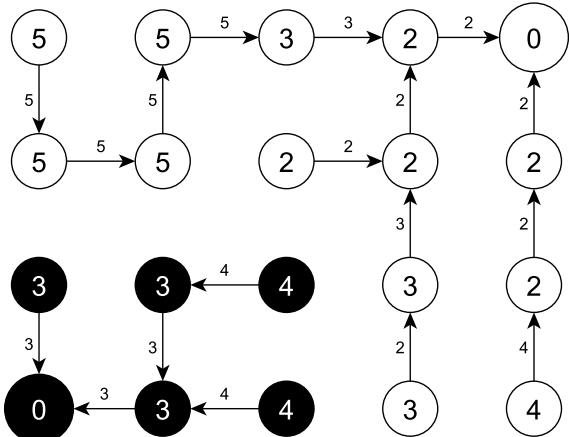
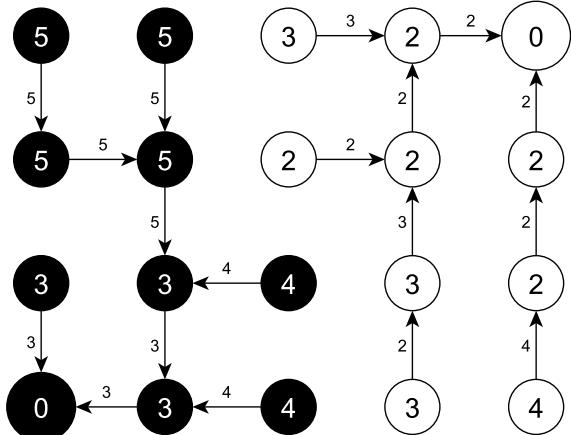
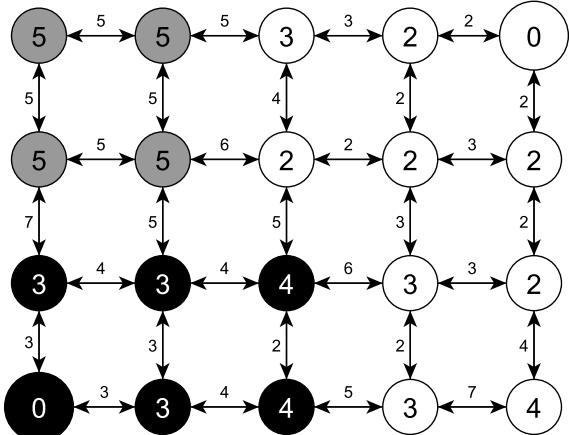
(d) Política LIFO com adjacência de 8 vizinhos

Figura 3.2: Exemplos de políticas de desempate.

exemplifica a ordem de processamento e o papel do mapa de predecessores durante a execução da IFT.

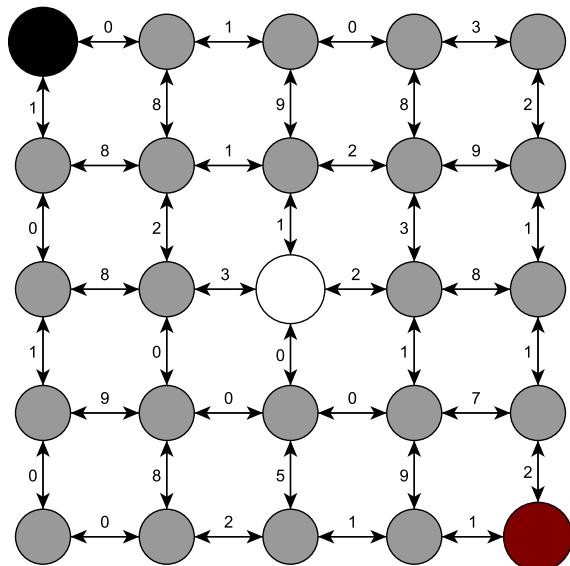


(a) Imagem de entrada

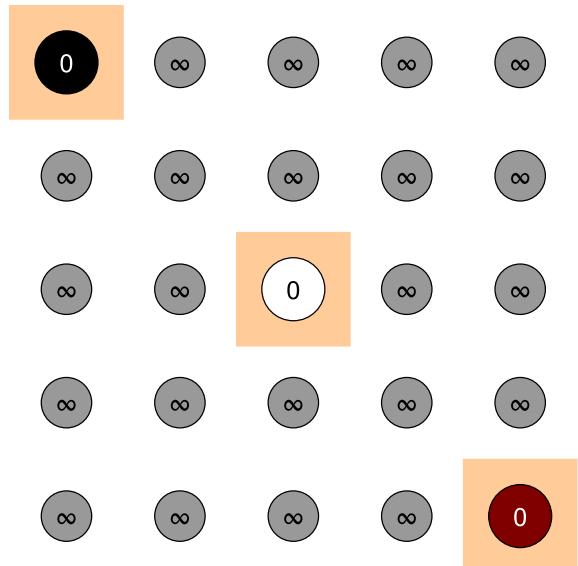
(b) SF_1 (c) SF_2 

(d) Zona de empate

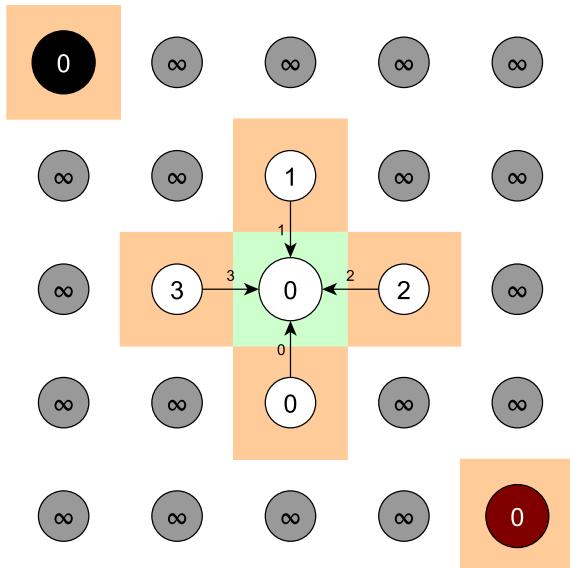
Figura 3.3: (a) Um grafo com duas raízes iniciais com valor 0 e os custos de cada aresta. (b-c) Duas possíveis SF de caminhos ótimos através da função f_{max} (Equação 3.2). (d) Zona de empate no grafo (cor cinza).



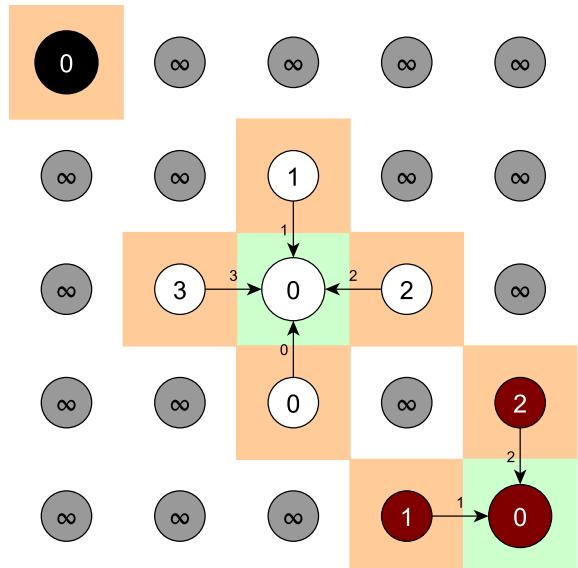
(a) Grafo inicial



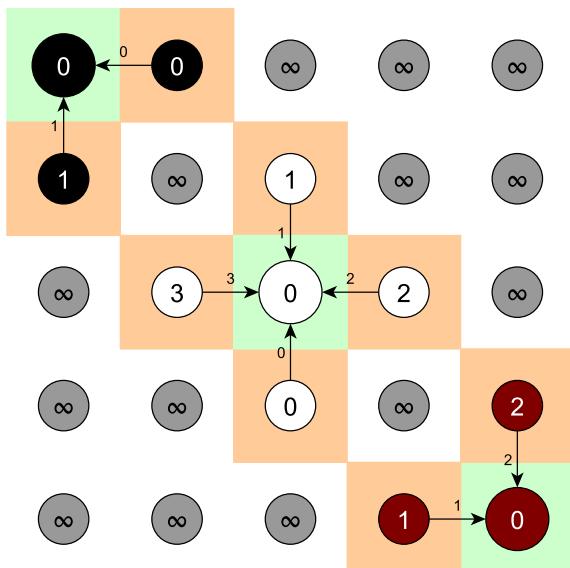
(b) Inicialização do mapa C



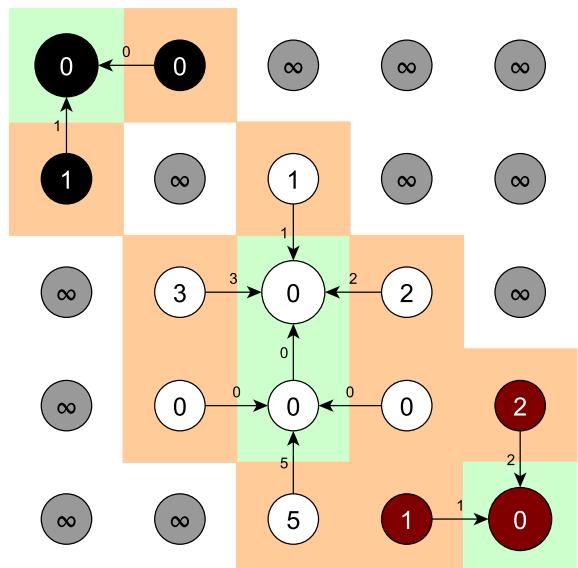
(c) Iteração 1



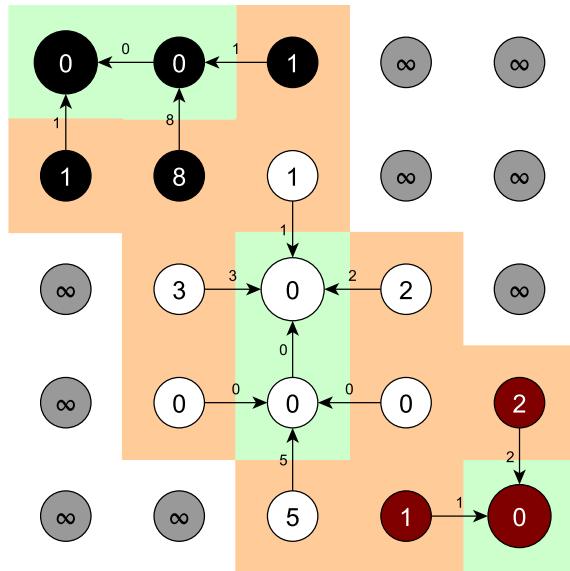
(d) Iteração 2



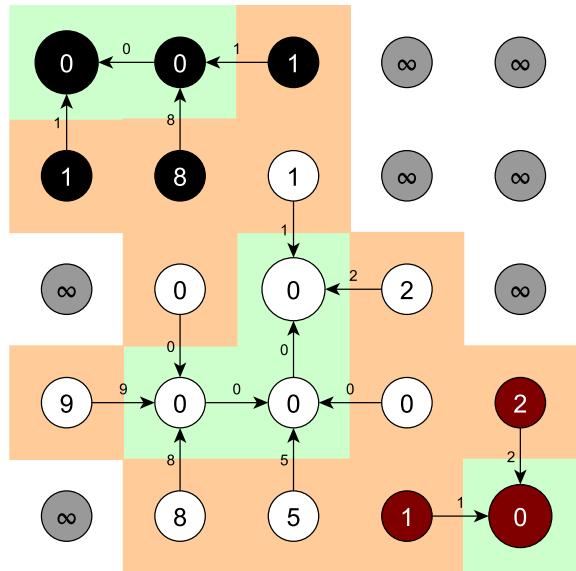
(e) Iteração 3



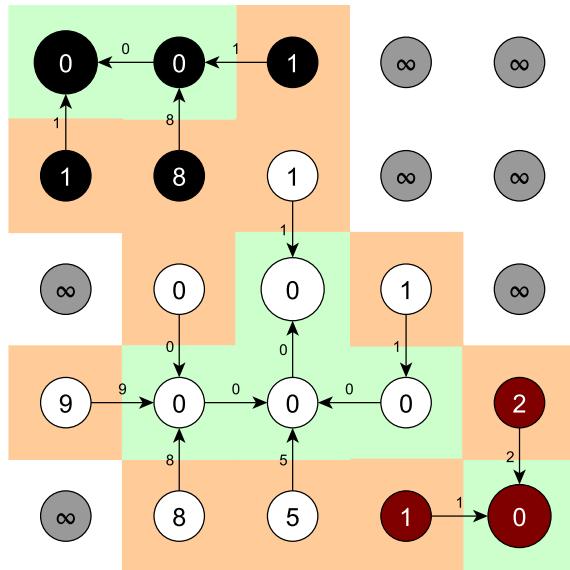
(f) Iteração 4



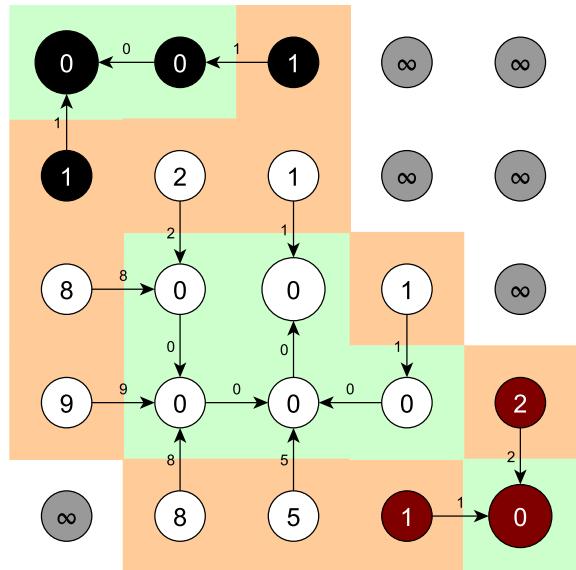
(g) Iteração 5



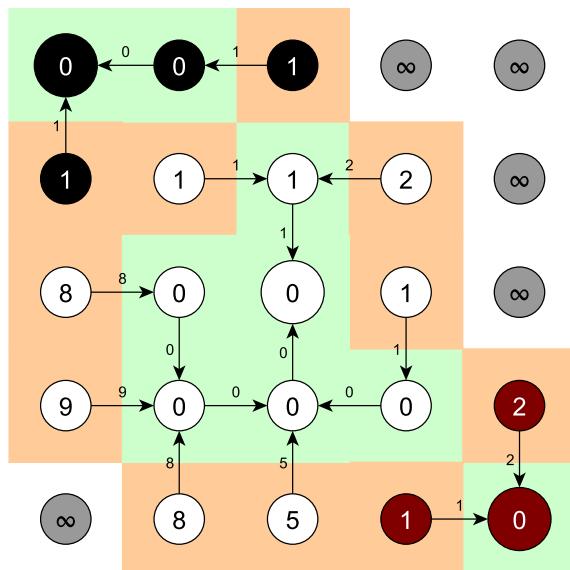
(h) Iteração 6



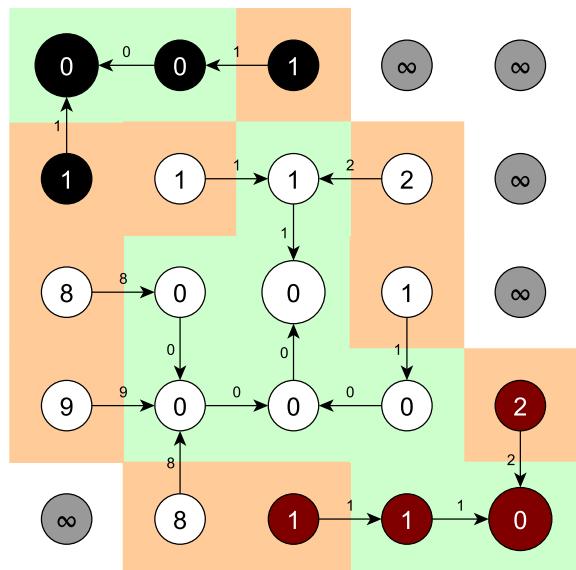
(i) Iteração 7



(j) Iteração 8



(k) Iteração 9



(l) Iteração 10

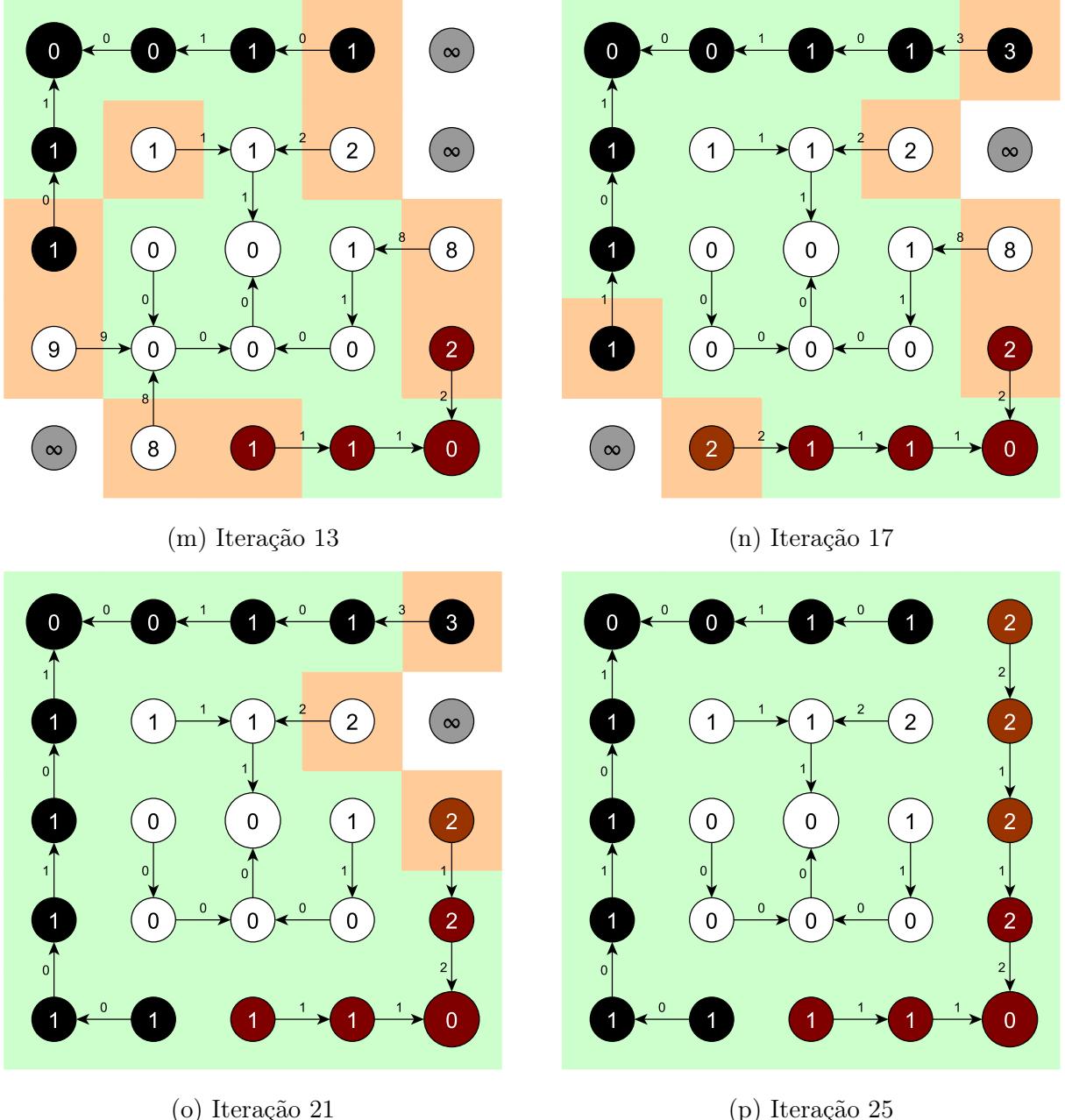


Figura 3.4: Exemplo de execução da IFT com função de conexidade f_{max} em um grafo. Os nós com cor de fundo laranja correspondem aos nós presentes na fila de prioridade Q a cada iteração. Cor de fundo verde é usada para indicar nós que foram removidos de Q em definitivo. Os valores dentro dos nós representam os custos dos melhores caminhos encontrados.

3.4.4 Exemplos de Segmentação

A Figura 3.5 apresenta um exemplo de segmentação supervisionada de uma imagem usando a IFT com a função f_{max} , as sementes do objeto S_o e do fundo S_f selecionadas pelo usuário são demonstradas pelas cores amarelo e roxo respectivamente. Pixels pertencentes à $S_o \cup S_f$ obtêm $H(t) = 0$, enquanto os restantes obtêm $H(t) = +\infty$. S_o e S_f competem entre si gerando duas SF de caminhos ótimos.

Um exemplo de segmentação não supervisionada é demonstrado na Figura 3.6¹. Nesse caso,

¹Imagem extraída da base de imagens Dorini *et al.* (2007).

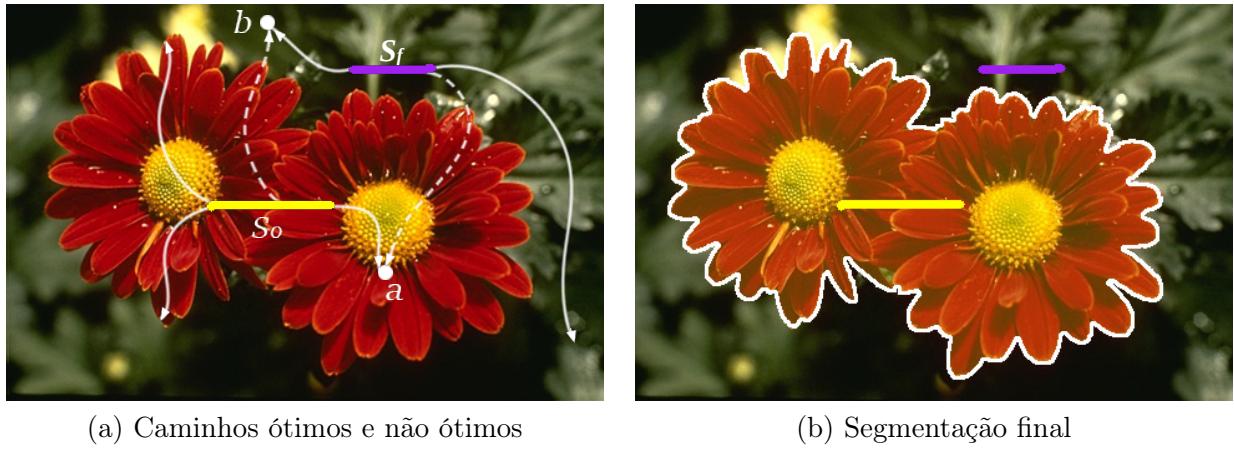


Figura 3.5: Exemplo de segmentação supervisionada, onde S_o e S_f representam as sementes do objeto e do fundo respectivamente. (a) Exemplos de caminhos ótimos ($\langle S_o, \dots, a \rangle$) e caminhos não ótimos ($\langle S_o, \dots, b \rangle$ e $\langle S_f, \dots, a \rangle$). (b) Segmentação final onde a borda branca representa os limites entre as SFs de S_o e S_f .

as sementes são selecionadas automaticamente através da extração dos mínimos locais da imagem. Cada semente competirá com as outras e obterá sua própria árvore de caminhos ótimos.

3.5 DIFT

Vários cenários exigem o recálculo de um método de clusterização em uma mesma imagem usando sementes diferentes. Esse processo pode se tornar custoso dependendo do número de iterações, complexidade do método de clusterização e o tamanho da imagem.

Falcão e Bergo (2004) desenvolveram um método chamado Transformada Imagem-Floresta Diferencial (*Differential Image Foresting Transform* DIFT) para esse tipo de cenário que alivia o tempo de computação de IFTs sequenciais calculando-as de forma diferencial. Ou seja, sendo $G = (V, A)$ o grafo da imagem $\hat{I} = (D_I, \vec{I})$ usado na DIFT, somente os nós em V que necessitam de mudanças no caminho ótimo serão recalculados.

Para usar a DIFT é necessário que tanto o grafo quanto a função de conexidade usados na IFT sejam idênticos durante cada iteração $t = \langle 1, 2, \dots, n \rangle$. A cada iteração novas sementes podem ser adicionadas no conjunto S_a e outras sementes antigas removidas no conjunto S_r . Quando uma semente é removida, consequentemente sua árvore de caminhos ótimos também é removida, seus nós se tornam disponíveis novamente para uma nova disputa e os nós de fronteira \mathcal{F} das árvores vizinhas são adicionados na fila Q para expansão em conjunto com as novas sementes em S_a .

3.5.1 Algoritmo

A DIFT, assim como a IFT, é essencialmente o algoritmo de Dijkstra com modificações para suporte a múltiplos rótulos e adição e remoção de árvores de uma forma diferencial.

Inicialmente a DIFT recebe os valores de P , C , \mathcal{R} e R da primeira iteração da IFT e $S_r \leftarrow \emptyset$.

O Algoritmo 2 começa chamando a função auxiliar *DIFT-TreeRemoval* demonstrada no Algoritmo 3 que serve para reiniciar todos os nós pertencentes às árvores contidas em S_r , ou seja, todo nó $t \in V$ tal que $\exists r \in S_r$, $R(t) = R(r)$, tem seu custo $C(t)$ e predecessor $P(t)$ reinicializados para $+\infty$ e *nil* respectivamente. Além disso, o Algoritmo 3 também adiciona em \mathcal{F} todos os nós de árvores não removidas que sejam adjacentes aos nós removidos anteriormente, isto é, todo nó $t \in V$ tal que $s \in \mathcal{A}(t)$, em que a árvore de s foi removida porém a de t não, é adicionado à \mathcal{F} . Já que os nós pertencentes à \mathcal{F} não foram reinicializados, eles serão responsáveis na disputa da IFT pelos nós removidos.

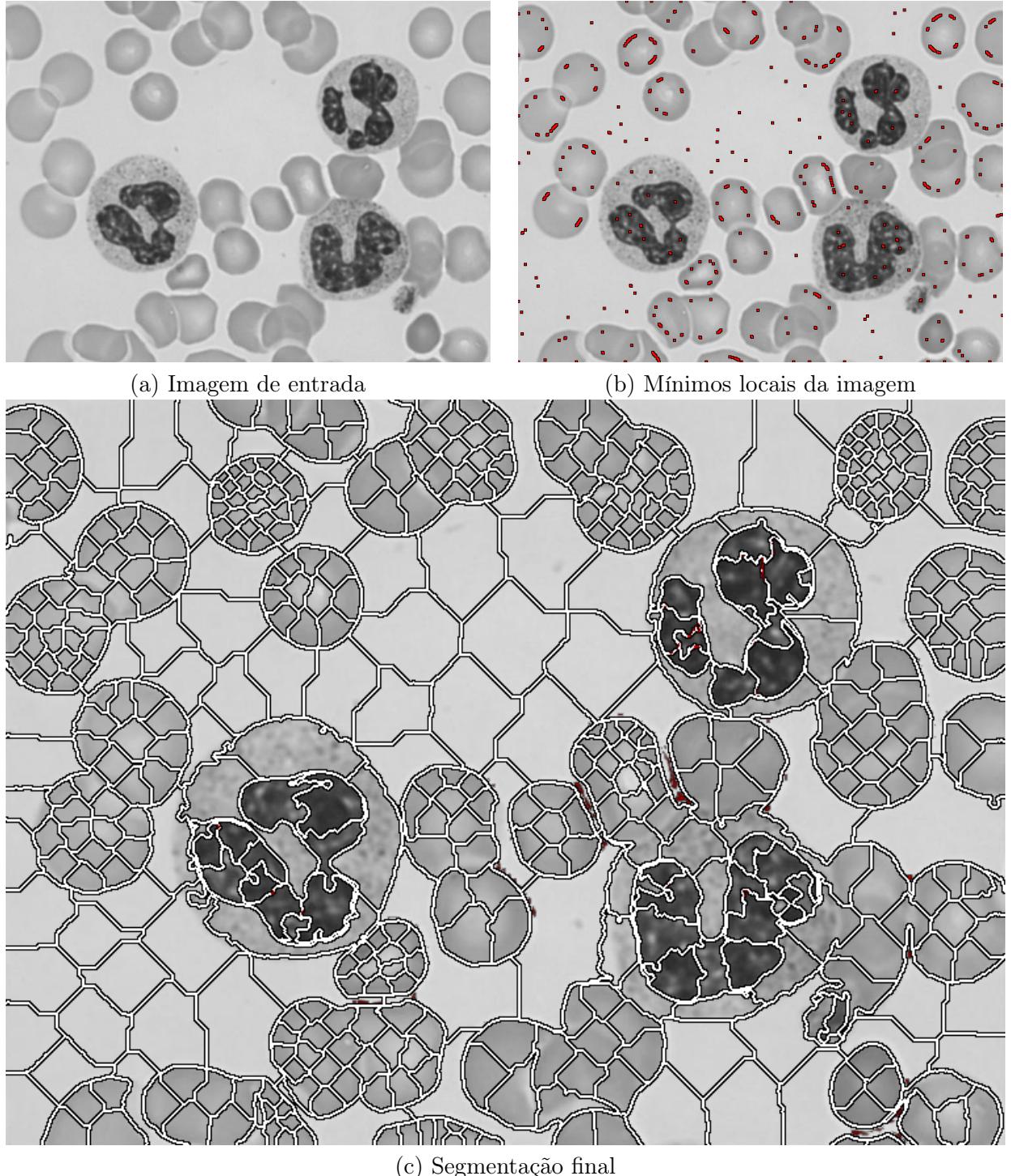


Figura 3.6: (a-b) Imagem de entrada e as sementes selecionadas de forma não supervisionada. (c) A segmentação final onde cada semente gerou sua própria SF de caminhos ótimos.

Após a chamada da função *DIFT-TreeRemoval*, são removidos de \mathcal{F} todas as raízes S_a que porventura nele estejam contidas (linha 4). Essas raízes contidas em S_a são readicionadas à \mathcal{F} nas linhas 5-8 caso seu custo trivial seja menor que o seu custo já contido em C , além disso, caso positivo, os valores de C , \mathcal{R} e P são reinicializados. Nesse momento \mathcal{F} conterá tanto os nós de S_a quanto os retornados pela função *DIFT-TreeRemoval* obtidas na linha 3. Já nas linhas 9-10, todos os nós pertencentes a \mathcal{F} são adicionados à Q . Nas linhas 11-12, o mapa T é preenchido com o valor *false*. O laço iniciado na linha 13, assim como no algoritmo original da IFT, serve para o cálculo

dos caminhos ótimos em P .

Algoritmo 2 DIFT

Entrada:

Grafo $G = (V, A)$ da imagem $\hat{I} = (D_I, \vec{I})$
 Mapa P de predecessores
 Mapa C de custos mínimos
 Mapa \mathcal{R} de raízes
 Mapa R de rótulos
 Relação de adjacência \mathcal{A}
 Função de conexidade f
 Conjunto S_a de sementes a serem adicionadas
 Conjunto S_r de sementes a serem removidas

Auxiliares:

Fila Q de prioridade
 Variável $custo$
 Conjunto \mathcal{F} com nós de fronteira
 Mapa T de nós já verificados

Saída:

Mapa P de predecessores
 Mapa C de custos mínimos
 Mapa \mathcal{R} de raízes
 Mapa R de rótulos

```

1: procedimento DIFT
2:    $Q \leftarrow \emptyset$ 
3:    $(C, P, \mathcal{F}) \leftarrow \text{DIFT-TreeRemoval}(C, R, P, \mathcal{A}, S_r)$ 
4:    $\mathcal{F} \leftarrow \mathcal{F} \setminus S_a$ 
5:   enquanto  $S_a \neq \emptyset$  faça
6:     Remova  $t$  de  $S_a$ 
7:     se  $f(\langle t \rangle) < C(t)$  então
8:        $C(t) \leftarrow f(\langle t \rangle)$ ,  $\mathcal{R}(t) \leftarrow t$ ,  $R(t) \leftarrow x$  para  $t \in S_{a_x}$ ,  $P(t) \leftarrow \text{nil}$  e  $\mathcal{F} \leftarrow \mathcal{F} \cup \{t\}$ 
9:   enquanto  $\mathcal{F} \neq \emptyset$  faça
10:    Remova  $t$  de  $\mathcal{F}$  e Insira  $t$  em  $Q$ 
11:    para todo  $s \in V$  faça
12:       $T(s) \leftarrow \text{false}$ 
13:    enquanto  $Q \neq \emptyset$  faça
14:      Remova  $s$  de  $Q$  cujo valor  $C(s)$  é mínimo
15:       $T(s) \leftarrow \text{true}$ 
16:      para todo  $t \in \mathcal{A}(s)$ , tal que  $T(t) = \text{false}$  faça
17:         $custo \leftarrow f(\pi_s^P \cdot \langle s, t \rangle)$ 
18:        se  $custo < C(t)$  ou  $P(t) = s$  então
19:          se  $C(t) \neq +\infty$  então Remova  $t$  de  $Q$ 
20:           $P(t) \leftarrow s$ ,  $C(t) \leftarrow custo$ ,  $\mathcal{R}(t) \leftarrow \mathcal{R}(s)$  e  $R(t) \leftarrow R(s)$ 
21:          Insira  $t$  em  $Q$ 
```

Algoritmo 3 DIFT-TreeRemoval

Entrada:

Mapa C de custos mínimos
 Mapa R de rótulos
 Mapa P de predecessores
 Relação de adjacência \mathcal{A}
 Conjunto S_r de sementes a serem removidas

Auxiliares:

Fila T de prioridade
 Conjunto R_r de rótulos de árvores marcadas para remoção

Saída:

Mapa C de custos mínimos
 Mapa P de predecessores
 Conjunto \mathcal{F} com nós de fronteira

```

1: procedimento DIFT-TREEREMOVAL
2:    $R_r \leftarrow \emptyset$  e  $\mathcal{F} \leftarrow \emptyset$ 
3:   enquanto  $S_r \neq \emptyset$  faça
4:     Remova  $t$  de  $S_r$ 
5:      $r \leftarrow R(t)$  e  $R_r \leftarrow R_r \cup \{r\}$ 
6:     se  $C(r) \neq +\infty$  então
7:        $C(r) \leftarrow +\infty$  e  $P(r) \leftarrow \text{nil}$ 
8:       Insira  $r$  em  $T$ 
9:   enquanto  $T \neq \emptyset$  faça
10:    Remova  $s$  de  $T$ 
11:    para todo  $t \in \mathcal{A}(s)$  faça
12:      se  $P(t) = s$  então
13:         $C(t) \leftarrow +\infty$  e  $P(t) \leftarrow \text{nil}$ 
14:        Insira  $t$  em  $T$ 
15:      else
16:        se  $R(t) \notin R_r$  então
17:           $\mathcal{F} \leftarrow \mathcal{F} \cup \{t\}$ 

```

3.5.2 Teste de Predecessor

Vale a pena salientar que o laço da linha 13 do Algoritmo 2 é essencialmente idêntico ao do Algoritmo 1 da IFT, com a única diferença na linha 18 da adição de um teste de predecessor $P(t) = s$. Esse teste garante que quando uma nova semente é inclusa em S_a , todos os nós $t \in V$ acessíveis por ela por algum caminho ótimo $\pi_k = \pi_s \cdot \langle s, t \rangle$ com custo $f(\pi_k) \leq C(t)$ serão reavaliados. Porém, quando $f(\pi_k) = C(t)$, t só será reavaliado se $P(t) = s$.

A Figura 3.7 ilustra o problema resolvido pelo teste de predecessor. Seja $\langle a, \dots, s, t, \dots, u \rangle$ um caminho ótimo de a a u em uma floresta de caminhos ótimos com 2 sementes, a e b , e c uma nova semente adicionada numa iteração subsequente, tal que há um caminho $\langle c, \dots, s \rangle$ com $f(\langle c, \dots, s \rangle) < C(s)$ e $f(\langle c, \dots, s, t \rangle) = C(t)$. Sem o teste de predecessor, t não será inserido em Q quando for visitado por s e consequentemente os nós de $\langle t, \dots, u \rangle$ não serão inseridos também. Nesse caso, mesmo se $\langle c, \dots, s, t, \dots, u \rangle$ seja um caminho ótimo de c para u , \mathcal{R} apontará para a antiga semente a para todos os nós do caminho $\langle t, \dots, u \rangle$.

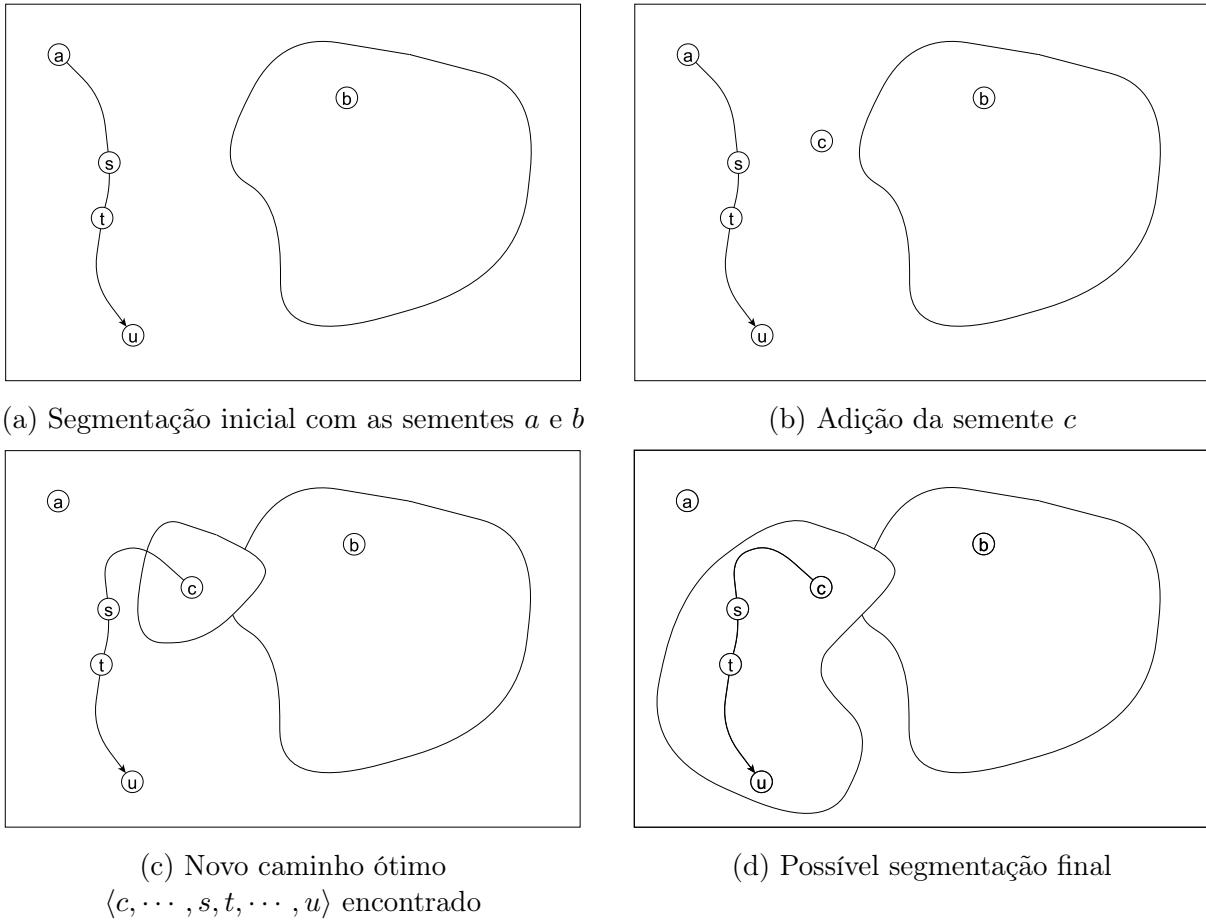


Figura 3.7: (a) Exemplo de uma SF de caminhos ótimos com duas árvores com raízes em a e b , onde $\langle a, \dots, s, t, \dots, u \rangle$ é um caminho ótimo. Em (b), a semente c é adicionada. Já em (c), o caminho $\langle c, \dots, s \rangle$ é encontrado como um caminho ótimo mais barato que $\langle a, \dots, s \rangle$, porém o custo atual de t ($C(t)$) é igual ao custo de $f(\langle c, \dots, s, t \rangle)$. O teste de predecessor garante nesse caso que todos os pixels com caminhos ótimos passando por s , como é o caso de $\langle t, \dots, u \rangle$, serão reavaliados e possivelmente reatribuídos para a semente c . (d) Uma possível segmentação final da DIFT com a adição da semente c .

3.6 SLIC

O SLIC adapta agrupamentos k -means para eficientemente gerar superpixels. Superpixels SLIC correspondem a agrupamentos no espaço de características $labxy$. SLIC tem dois parâmetros, o número k de superpixels desejados com tamanho aproximadamente igual, e um parâmetro m para oferecer controle sobre sua compacidade. A sua complexidade é linear no número de pixels N , e independente do número de superpixels k .

Para imagens em cores, o algoritmo SLIC tem as seguintes etapas:

- Em primeiro lugar, a imagem de entrada é convertida para o espaço de cor CIELAB.
- Em seguida, um total de k'^2 centros de agrupamentos $c_i = [l_i \ a_i \ b_i \ x_i \ y_i]^T$ iniciais são amostrados em uma malha regular espaçada em $\mathcal{S} = \sqrt{N/k}$ pixels separados.
- Opcionalmente, os centros podem ser movidos para a posição mais baixa de gradiente numa vizinhança 3×3 , para evitar a inicialização em um pixel com muito ruído.

²SLIC não garante o número exato k de superpixels desejados. Só k' centros iniciais são realmente utilizados, em que k' é um valor aproximado de k ($k' \approx k$), de acordo com seu código fonte.

- A seguir, no passo de atribuição, cada pixel é associado com o centro do agrupamento mais próximo de acordo com uma medida de distância D , mas considerando apenas os centros cuja região de $2S \times 2S$ pixels sobrepõe a sua localização.
- Depois disso, um passo de atualização ajusta os centros dos grupos a um vector $[l \ a \ b \ x \ y]^T$ da média de todos os pixels que pertencem ao agrupamento.
- Os passos de atribuição e de atualização são então repetidos para um total de 10 iterações.
- No final, alguns pixels disjuntos que não pertencem ao mesmo componente conexo que o seu centro de agrupamento podem permanecer. Portanto, uma etapa de pós-processamento para forçar a conexidade é aplicado através da atribuição de um rótulo distinto para cada componente conexo³.

A medida de distância, D , é:

$$d_c = \sqrt{(l_i - l_j)^2 + (a_i - a_j)^2 + (b_i - b_j)^2}, \quad (3.9)$$

$$d_s = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}, \quad (3.10)$$

$$D = \sqrt{d_c^2 + \left(\frac{d_s}{S}\right)^2 m^2}, \quad (3.11)$$

em que m dá a importância relativa entre dissimilaridade de cor (d_c) e distância espacial (d_s). Quando m é grande, os superpixels resultantes são mais compactos, em contrapartida, quando m é pequeno, os superpixels têm uma melhor aderência para as bordas dos objetos presentes na imagem, mas com tamanho e forma irregulares.

³Além disso, de acordo com o seu código fonte, se um determinado componente é muito pequeno, ele é mesclado com um componente adjacente encontrado anteriormente. Então, o número k' inicial de superpixels muda no final quando segmentos são adicionados ou removidos.

Capítulo 4

IFT-SLIC

Semelhante ao SLIC, o IFT-SLIC começa com a mesma seleção k' de centros de agrupamentos iniciais $c_i = [l_i \ a_i \ b_i \ x_i \ y_i]^T$, que são amostrados em uma malha regular espaçada em $\mathcal{S} = \sqrt{N/k}$ pixels de distância.

A principal diferença ao SLIC reside no passo de atribuição. Em vez de usar uma abordagem de agrupamento k -means adaptativo, considera-se o cálculo de uma IFT com a função de conexidade não monotonicamente incremental f_D , a qual é baseada na função de custo de caminho $f_{\sum |\Delta I|}$ de Mansilla *et al.* (2013a) que usa a soma do valor absoluto de intensidades relativas. Essas funções são justificadas pelos resultados teóricos e experimentais apresentados por Mansilla *et al.* (2013a)¹.

Os centros dos agrupamentos iniciais $c_i = [l_i \ a_i \ b_i \ x_i \ y_i]^T$ definem um conjunto de sementes S , tal que para cada pixel $r \in S$ na coordenada (x_r, y_r) , têm-se um centro de agrupamento correspondente $c_j = [l_j \ a_j \ b_j \ x_j \ y_j]^T$ e $(x_j, y_j) = (x_r, y_r)$.

É importante notar que a função de custo de caminho f_D desempenha o mesmo papel que a medida de distância D no SLIC.

$$\begin{aligned} f_D(\pi_t = \langle t \rangle) &= \begin{cases} 0 & \text{se } t \in S \\ +\infty & \text{caso contrário,} \end{cases} \\ f_D(\pi_{r \rightsquigarrow s} \cdot \langle s, t \rangle) &= f_D(\pi_s) + \left(\|\vec{I}(t) - \vec{I}_r\| \cdot \alpha \right)^\beta + \|s, t\|, \end{aligned} \quad (4.1)$$

em que $\vec{I}(t)$ é o vetor de cor no pixel t , ou seja, $\vec{I}(t) = [l_t \ a_t \ b_t]^T$, e \vec{I}_r é o vetor de cor do centro do agrupamento da semente r (ou seja, $\vec{I}_r = [l_r \ a_r \ b_r]^T$ em que $c_j = [l_j \ a_j \ b_j \ x_j \ y_j]^T$ e r está na coordenada (x_j, y_j)).

No final do passo de atribuição, cada agrupamento/superpixel será representado pela sua respectiva árvore na SF, isto é, o mapa de predecessores P calculado pela IFT.

Depois disso, uma etapa de atualização (*RecomputarSementes*, linha 18) ajusta os centros dos agrupamentos. Diferentemente do SLIC, que considera o vetor $[l \ a \ b \ x \ y]^T$ da média de todos os pixels pertencentes ao agrupamento, o IFT-SLIC toma como (x, y) as coordenadas do pixel do agrupamento mais próximo da posição média. A ideia é evitar a seleção de uma posição de atualização que se encontre fora do seu agrupamento.

Os passos de atribuição e de atualização são então repetidos num total de 10 iterações para igualar ao mesmo número de iterações recomendados por Achanta *et al.* (2010), porém, como os gráficos 5.33, 5.34 e 5.35 demonstram, é possível diminuir esse número sem perdas significativas de eficácia.

O IFT-SLIC não requer um passo de pós-processamento, já que a conexidade é garantida por construção. Sua complexidade é $N \log(N)$ independente do número de superpixels k devido a sua fila de prioridade *heap* Q . O IFT-SLIC pode ter sua complexidade reduzida à linear em relação ao

¹Por exemplo, as funções f_D e $f_{\sum |\Delta I|}$ são mais adaptáveis para lidar com problemas de falta de homogeneidade, que são comuns em imagens de RM de 3T (Mansilla *et al.*, 2013a).

número de pixels N com o uso de outras filas de prioridade especializadas.

4.1 Algoritmo

Algoritmo 4 IFT-SLIC

Entrada:

Grafo $G = (V, A)$ da imagem $\hat{I} = (D_I, \vec{I})$

Relação de adjacência \mathcal{A}

Função de conexidade f

Número máximo de iterações $MaxIters \geq 1$

Conjunto de sementes $S \subseteq V$

Auxiliares:

Fila Q de prioridade

Variável *custo*

Mapa T de nós já verificados

Saída:

Mapa P de predecessores

Mapa C de custos mínimos

Mapa \mathcal{R} de raízes

Mapa R de rótulos

```

1: procedimento IFT-SLIC
2:    $iter \leftarrow 0$ 
3:   enquanto  $iter < MaxIters$  faça
4:     para todo  $s \in V$  faça
5:        $P(s) \leftarrow \text{nil}$ ,  $C(s) \leftarrow f(\langle s \rangle)$ ,  $\mathcal{R}(s) \leftarrow s$  e  $T(s) \leftarrow \text{false}$ 
6:       se  $C(s) \neq +\infty$  então
7:         Insira  $s$  em  $Q$ 
8:          $R(s) \leftarrow x$  para  $s \in S_x$ 
9:     enquanto  $Q \neq \emptyset$  faça
10:      Remova  $s$  de  $Q$  cujo valor  $C(s)$  é mínimo
11:       $T(s) \leftarrow \text{true}$ 
12:      para todo  $t \in \mathcal{A}(s)$ , tal que  $T(t) = \text{false}$  faça
13:         $custo \leftarrow f(\pi_s^P \cdot \langle s, t \rangle)$ 
14:        se  $custo < C(t)$  então
15:          se  $C(t) \neq +\infty$  então Remova  $t$  de  $Q$ 
16:           $P(t) \leftarrow s$ ,  $C(t) \leftarrow custo$ ,  $\mathcal{R}(t) \leftarrow \mathcal{R}(s)$  e  $R(t) \leftarrow R(s)$ 
17:          Insira  $t$  em  $Q$ 
18:       $S \leftarrow RecomputarSementes(S, G, \mathcal{R})$ 
19:       $iter \leftarrow iter + 1$ 

```

4.2 Propriedades Teóricas

Nesta seção, será discutido sobre algumas propriedades teóricas do IFT-SLIC. Sendo $\mathcal{P}_1^i, \mathcal{P}_2^i, \dots, \mathcal{P}_k^i$ a partição de uma imagem $\bigcup_{j=1}^k \mathcal{P}_j^i = D_I$ em k superpixels obtidos na iteração i pelas sementes $s_1^i, s_2^i, \dots, s_k^i$ e as seguintes definições:

Definição 4.2.1 (*Caminhos restritos*). Seja B um subconjunto de nós do conjunto de nós de G . Um caminho $\pi_{s \rightsquigarrow t} = \langle t_1 = s, t_2, \dots, t_n = t \rangle$ indica um caminho que é restrito dentro do subgrafo

induzido por B ($t_i \in B$, $i = 1, \dots, n$ e $(t_i, t_{i+1}) \in A$, $i = 1, \dots, n - 1$).

Definição 4.2.2 (*Caminhos restritos ótimos*). Um caminho $\pi_{s \rightsquigarrow t}^B$ é restrito e ótimo em B se $f_D(\pi_{s \rightsquigarrow t}^B) \leq f_D(\tau_{x \rightsquigarrow t}^B)$ pra qualquer outro caminho restrito $\tau_{x \rightsquigarrow t}^B$ em B com o mesmo nó t de destino. A notação $\pi_{s \rightsquigarrow t}^*$ será usada para indicar explicitamente um caminho restrito ótimo.

Têm-se a seguinte proposição:

Proposição 4.2.1 (*árvores de caminhos restritos ótimos pelo ISF*). A árvore de espalhamento para cada partição \mathcal{P}_j^i , $j = 1, \dots, k$, computada pelo ISF com a função f_D é uma árvore de caminhos restritos ótimos em \mathcal{P}_j^i , ou seja, os caminhos $\pi_{s_j \rightsquigarrow t}^{\mathcal{P}_j^i}$ computados pela função de conexidade não monotonicamente incremental f_D são caminhos restritos ótimos dentro de suas partições \mathcal{P}_j^i .

A Proposição 4.2.1 pode ser provada percebendo que cada partição \mathcal{P}_j^i tem uma única semente s_j^i , consequentemente tornando a função f_D em uma função monotonicamente incremental no subgrafo induzido por \mathcal{P}_j^i para essa semente (Falcão et al., 2004).

Seja o conjunto de pixels de borda entre superpixels vizinhos para cada partição \mathcal{P}_j^i definido como $\mathcal{B}(\mathcal{P}_j^i) = \{t \in \mathcal{P}_j^i \mid \exists s \in \mathcal{A}(t) \text{ tal que } s \notin \mathcal{P}_j^i\}$. Define-se também a seguinte proposição:

Proposição 4.2.2 (*Proteção da borda*). Para cada pixel $t \in \mathcal{B}(\mathcal{P}_j^i)$, $j = 1, \dots, k$ se $s \in \mathcal{A}(t)$ é um pixel tal que $s \in \mathcal{P}_l^i$ e $l \neq j$, têm-se que $f_D(\pi_{s_j^i \rightsquigarrow t}^{\mathcal{P}_j^i}) \leq f_D(\pi_{s_l^i \rightsquigarrow s}^{\mathcal{P}_l^i} \cdot \langle s, t \rangle)$.

Basicamente, essa proposição mostra que cada superpixel \mathcal{P}_j^i é cercado por pixels de borda $\mathcal{B}(\mathcal{P}_j^i)$ que são iguais ou mais fortemente conectados a sua semente s_j^i do que pelos seus superpixels vizinhos através de qualquer extensão dos seus respectivos caminhos restritos ótimos.

A proposição 4.2.2 procede pela propagação ordenada do mapa de predecessores e pelo fato que f_D é uma função não decrescente. Resultando em dois casos dependendo de qual pixel (t ou s) é removido primeiramente de Q na linha 10. Se t é removido antes que s , então têm-se que $f_D(\pi_{s_j^i \rightsquigarrow t}^{\mathcal{P}_j^i}) \leq f_D(\pi_{s_l^i \rightsquigarrow s}^{\mathcal{P}_l^i})$, o que implica que $f_D(\pi_{s_j^i \rightsquigarrow t}^{\mathcal{P}_j^i}) \leq f_D(\pi_{s_l^i \rightsquigarrow s}^{\mathcal{P}_l^i} \cdot \langle s, t \rangle)$ já que f_D é uma função não decrescente. Caso contrário, se s é removido antes que t de Q , têm-se que $f_D(\pi_{s_l^i \rightsquigarrow s}^{\mathcal{P}_l^i} \cdot \langle s, t \rangle)$ é avaliada na linha 13 já que $T(t) = \text{false}$. Ou seja, $f_D(\pi_{s_j^i \rightsquigarrow t}^{\mathcal{P}_j^i})$ não pode ser pior que $f_D(\pi_{s_l^i \rightsquigarrow s}^{\mathcal{P}_l^i} \cdot \langle s, t \rangle)$, senão t seria conquistado pelo caminho $\pi_{s_l^i \rightsquigarrow s}^{\mathcal{P}_l^i} \cdot \langle s, t \rangle$. Portanto, ambos os casos resultam em $f_D(\pi_{s_j^i \rightsquigarrow t}^{\mathcal{P}_j^i}) \leq f_D(\pi_{s_l^i \rightsquigarrow s}^{\mathcal{P}_l^i} \cdot \langle s, t \rangle)$.

Agora é apresentado o primeiro teorema:

Teorema 1. *Os superpixels gerados pelo arcabouço ISF com a função de conexidade f_D são garantidamente conexos.*

Demonstração. No final de qualquer iteração do laço principal (linha 3), a partição da imagem computada pelo mapa de rótulos R resulta em um conjunto de superpixels computados. As partições \mathcal{P}_j^i , $j = 1, \dots, k$, são gradualmente computadas no laço da linha 9 pela remoção sucessiva de pixels de Q (linhas 10-11), tal que a qualquer instante $\mathcal{P}_j^i = \{t \in \hat{I} \mid R(t) = j\}$ e $T(t) = \text{true}$.

A geração de partições conectadas \mathcal{P}_j^i pode ser provada por indução matemática. No caso básico, têm-se inicialmente cada partição \mathcal{P}_j^i composta exclusivamente pela sua semente correspondente s_j^i , que é obviamente conectada. É importante notar que as sementes são iniciadas com o menor custo possível (linha 5) e portanto são os primeiros pixels a sair da fila de prioridade Q .

A condição $T(t) = \text{false}$ na linha 12 garante que qualquer pixel $t \in \mathcal{P}_j^i$ não pode ser posteriormente removido de \mathcal{P}_j^i e adicionado em outro superpixel, já que a mudança de rótulo em $R(t)$ só pode apenas acontecer na linha 16. Logo, no passo de indução, é necessário apenas provar que a conexidade de \mathcal{P}_j^i , $j = 1, \dots, k$ é mantida quando um novo nó s é adicionado à \mathcal{P}_j^i depois de ser removido de Q nas linhas 10-11.

De acordo com a linha 16, o predecessor $P(s)$ do nó s tem o mesmo rótulo que s , ou seja, $R(P(s)) = R(s)$. Portanto, o nó s é necessariamente conectado a um superpixel \mathcal{P}_j^i em que $j = R(s)$. O uso de 4 vizinhos garantem superpixels conectados não apenas na topologia do grafo como no domínio da imagem. Essa adjacência simétrica leva a um dígrafo fortemente conexo garantindo que todos os pixels são atribuídos a algum superpixel. Logo, com essa escolha de conectividade e adjacência, garante-se a geração de uma partição da imagem em superpixels conectados. ■

Teorema 2. *Se para uma próxima iteração $i + 1$, as novas sementes s_j^{i+1} são selecionadas de tal forma que $\sum_{t \in \mathcal{P}_j^i} f(\pi_{s_j^{i+1}}^* \xrightarrow{\mathcal{P}_j^i} t) < \sum_{t \in \mathcal{P}_j^i} f(\pi_{s_j^i}^* \xrightarrow{\mathcal{P}_j^i} t)$, em que $j = 1, \dots, k$ e f é uma função monotonicamente incremental, logo os superpixels gerados pelo arcabouço ISF garantidamente convergerão.*

Demonstração. Para cada iteração i , considere a funcional F_i :

$$F_i = \sum_{j=1}^k \sum_{t \in \mathcal{P}_j^i} f(\pi_{s_j^i}^* \xrightarrow{\mathcal{P}_j^i} t) = \sum_{t \in D_i} C_i(t), \quad (4.2)$$

em que $C_i(t)$ denota o mapa C de conexidade computado pela IFT em sua iteração i .

Para as novas sementes, temos:

$$F_i = \sum_{j=1}^k \sum_{t \in \mathcal{P}_j^i} f(\pi_{s_j^i}^* \xrightarrow{\mathcal{P}_j^i} t) > \sum_{j=1}^k \sum_{t \in \mathcal{P}_j^{i+1}} f(\pi_{s_j^{i+1}}^* \xrightarrow{\mathcal{P}_j^{i+1}} t) \quad (4.3)$$

Os superpixels \mathcal{P}_j^{i+1} computados na próxima iteração são normalmente diferentes dos superpixels \mathcal{P}_j^i da iteração passada, mas $\mathcal{P}_j^{i+1} \cap \mathcal{P}_j^i \neq \emptyset$, por causa da imposição das sementes e $s_j^{i+1} \in \mathcal{P}_j^i$. Para qualquer pixel $p \in \mathcal{P}_j^i$ tal que $p \notin \mathcal{P}_j^{i+1}$, se f é uma função monotonicamente incremental, pode-se concluir que p foi conquistado, na iteração $i + 1$, por um caminho ótimo $\pi_{s_l^{i+1}}^* \xrightarrow{\mathcal{P}_l^{i+1}} p$, tal que $l \neq j$ e $f(\pi_{s_l^{i+1}}^* \xrightarrow{\mathcal{P}_l^{i+1}} p) \leq f(\pi_{s_j^{i+1}}^* \xrightarrow{\mathcal{P}_j^{i+1}} p)$. O que nos dá:

$$\sum_{j=1}^k \sum_{t \in \mathcal{P}_j^i} f(\pi_{s_j^{i+1}}^* \xrightarrow{\mathcal{P}_j^{i+1}} t) \geq \sum_{j=1}^k \sum_{t \in \mathcal{P}_j^{i+1}} f(\pi_{s_j^{i+1}}^* \xrightarrow{\mathcal{P}_j^{i+1}} t) = F_{i+1} \quad (4.4)$$

Combinando as equações 4.4 e 4.3, obtém-se:

$$F_i = \sum_{j=1}^k \sum_{t \in \mathcal{P}_j^i} f(\pi_{s_j^i}^* \xrightarrow{\mathcal{P}_j^i} t) > \sum_{j=1}^k \sum_{t \in \mathcal{P}_j^{i+1}} f(\pi_{s_j^{i+1}}^* \xrightarrow{\mathcal{P}_j^{i+1}} t) = F_{i+1} \quad (4.5)$$

Já que a cada passo iterativo o valor de F_i ($F_{i+1} < F_i$) é necessariamente diminuído e F_i é limitado inferiormente por 0 (o custo de caminhos triviais pelas sementes), obtém-se a prova da convergência. Conforme i cresce, F_i irá converger para o mínimo local. ■

Caso a próxima iteração $i + 1$ e a melhor semente leve a $\sum_{t \in \mathcal{P}_{j,i}} f(\pi_{s_j^{i+1}}^* \xrightarrow{\mathcal{P}_j^i} t) = \sum_{t \in \mathcal{P}_j^i} f(\pi_{s_j^i}^* \xrightarrow{\mathcal{P}_j^i} t)$, então deve-se selecionar a mesma semente (ex., $s_j^{i+1} = s_j^i$) para estabilizar os resultados.

4.2.1 Convergência com Funções não Monotonicamente Incrementais

Agora é discutida a convergência para o caso da função não monotonicamente incremental. Na etapa de atualização, para cada superpixel \mathcal{P}_j^i , um pixel centralizado $s_j^{i+1} \in \mathcal{P}_j^i$ é selecionado. Sendo f_D uma função aditiva, uma posição central irá geralmente diminuir $\sum_{t \in \mathcal{P}_j^i} f(\pi_{s_j^{i+1}}^* \xrightarrow{\mathcal{P}_j^i} t)$ devido à redução do tamanho dos caminhos computados.

O problema com o uso da função não monotonicamente incremental f_D é que não se pode garantir a validade da equação 4.4, ou seja, para um pixel $p \in \mathcal{P}_j^i$, tal que $p \notin \mathcal{P}_j^{i+1}$, p pode ser conquistado por um caminho $\pi_{s_l^{i+1}}^* \xrightarrow{\mathcal{P}_l^{i+1}} p$, tal que $l \neq j$ e $f(\pi_{s_l^{i+1}}^* \xrightarrow{\mathcal{P}_l^{i+1}} p) > f(\pi_{s_j^{i+1}}^* \xrightarrow{\mathcal{P}_j^i} p)$. Um possível método para resolver esse problema é detectando a situação acima e adicionando novas sementes fictícias para a função f_D ; a mesma sempre convergerá a uma função monotonicamente incremental². Importante também notar que F_i diminui conforme novas sementes são adicionadas. As sementes fictícias podem ser depois promovidas a sementes reais e gerar seus próprios superpixels, ou, podem ser eliminadas após a convergência deixando suas regiões para serem conquistadas pelos seus superpixels vizinhos em uma última execução da IFT.

4.3 Detalhes de Implementação

A fim de reduzir o tempo de computação necessário para o método, foi usada a seguinte estratégia de implementação, empregando a DIFT apresentada na Seção 3.5.

Seja $c_i^t = [l_i^t \ a_i^t \ b_i^t \ x_i^t \ y_i^t]^T$ o centro i do agrupamento na iteração t . Durante os cálculos consecutivos da IFT, apenas o centro de agrupamento para c_i^{t+1} é recalculado se

$$\|[l_i^t \ a_i^t \ b_i^t] - [l_i^{t+1} \ a_i^{t+1} \ b_i^{t+1}]\| > \epsilon_c$$

ou

$$\|[x_i^t \ y_i^t] - [x_i^{t+1} \ y_i^{t+1}]\| > \epsilon_s.$$

Os centros marcados para recálculo têm suas árvores removidas ao executar o algoritmo *DIFT-TreeRemoval* (Algoritmo 3 da Seção 3.5), e suas novas posições de sementes são adicionadas ao conjunto de sementes, para computar novas árvores, que podem invadir as zonas de influência de outras raízes (Algoritmo 2 da Seção 3.5). Quando uma árvore é removida da floresta, seus pixels ficam disponíveis para uma nova disputa entre as raízes restantes.

Os Gráficos 4.1, 4.2 e 4.3 demonstram a velocidade em que os superpixels deixam de ser recomputados com valores 5 e 2 para ϵ_c e ϵ_s respectivamente.

Vale salientar que os valores de ϵ_c e ϵ_s também podem ser especificados respectivamente como um percentual da variação de brilho da imagem de entrada e da variação espacial em milímetros no caso de imagens médicas. Porém, dado que a resolução radiométrica e espacial das imagens testadas nos experimentos não sofrem grandes variações, foram usados valores absolutos tanto de brilho quanto de variação espacial em número de pixel.

²No caso extremo, se todos os pixels forem considerados sementes, então todos os caminhos triviais se tornam ótimos. Mas geralmente o problema somente se apresenta em regiões não cobertas por sementes com cores bem distintas e que estão próximas a bordas de superpixels. Marcando essas regiões o problema é resolvido.

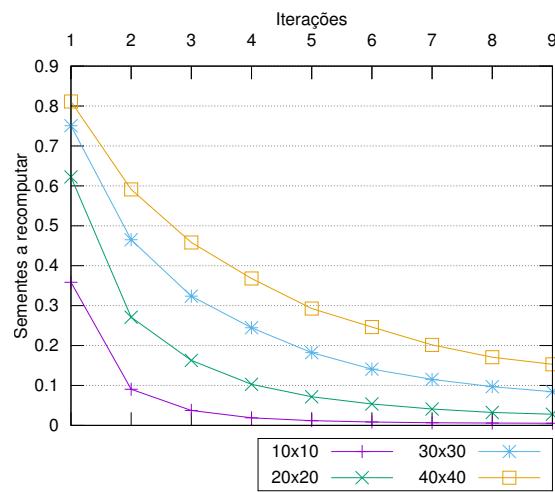


Gráfico 4.1: Exemplo da percentagem de sementes em forma decimal a serem computadas diminuindo na base de imagens Grabcut a cada iteração.

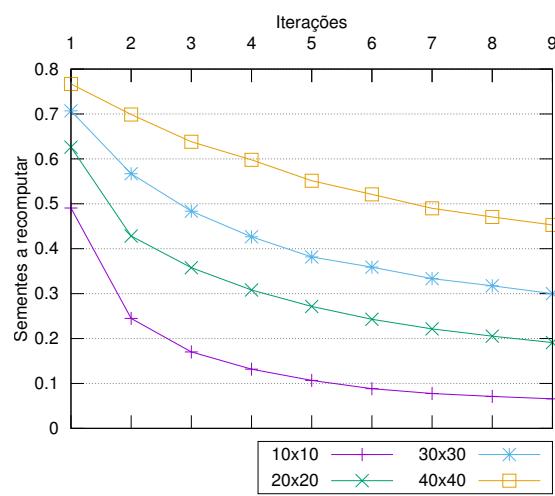


Gráfico 4.2: Exemplo da percentagem de sementes em forma decimal a serem computadas diminuindo na base de imagens Liver a cada iteração.

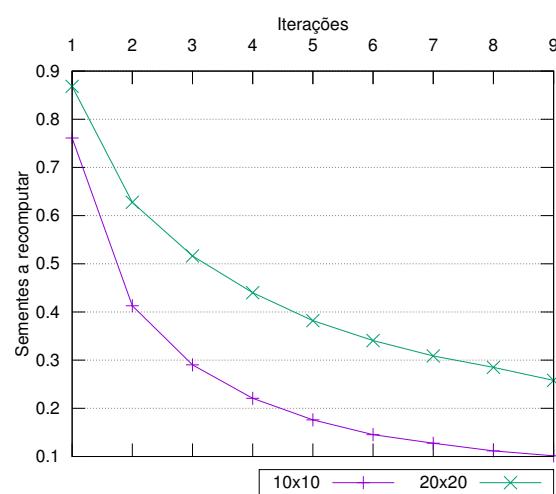


Gráfico 4.3: Exemplo da percentagem de sementes em forma decimal a serem computadas diminuindo na base de imagens Tálu a cada iteração.

Capítulo 5

Resultados

Nesta seção discute-se os resultados obtidos do IFT-SLIC em comparação direta ao SLIC¹. Como o SLIC é um dos métodos mais populares aplicados para a geração de superpixel, logo existem muitos trabalhos na literatura que demonstram sua superioridade em eficácia e eficiência com relação a outros métodos (Achanta *et al.*, 2010, 2012; Schick *et al.*, 2012), o que o torna um ótimo candidato para comparação ao IFT-SLIC. Para obter-se uma análise mais imparcial e abrangente do desempenho dos algoritmos abordados, foi executada extensiva bateria de testes com uma alta variação dos valores dos parâmetros, evitando assim, a fixação de forma arbitrária desses parâmetros e, consequentemente, anulando qualquer tendência para a má seleção dos mesmos. Além disso, o grande volume de dados obtidos permite a análise da capacidade de cada método de se adequar aos mais variados cenários, quantificar o quanto a mudança de um dado parâmetro afeta positivamente ou negativamente na qualidade da segmentação final e também na definição dos melhores valores de parâmetros para o maior número de cenários possíveis.

Vale salientar que a definição de uma certa configuração de parâmetros de um algoritmo como sua melhor configuração é subjetiva, já que sua escolha está intrinsecamente relacionada ao objetivo final da segmentação, ou seja, em um cenário no qual as informações de borda dos objetos têm prioridade, uma configuração de parâmetros que resulta em superpixels menos compactos, porém com alta aderência às bordas dos objetos será favorável, isto é exemplificado na Seção 5.6 na qual é demonstrada uma aplicação de alto nível de superpixels.

Na Tabela 5.1, é mostrada a gama de parâmetros usados por cada algoritmos nos testes.

SLIC	
p	desativado
m	[1, 2, 3, ..., 78, 79, 80]
IFT-SLIC	
p	ativado
c	ativado
α	[0.0025, 0.005, 0.0075, ..., 0.1950, 0.1975, 0.2]
β	2 e 12
ϵ_c	5
ϵ_s	2

Tabela 5.1: Tabela de parâmetros, SLIC varia o parâmetro m de 1 até 80, IFT-SLIC varia o parâmetro α de 0.0025 até 0.2 com passos de 0.0025

O SLIC, em seu artigo original (Achanta *et al.*, 2010), aplica, como passo opcional, a perturbação inicial das sementes dos superpixels para evitar sementes em regiões de alto contraste. Essa etapa é ativada pelo parâmetro p , ao qual, como mostrado na Tabela 5.1, foi testado desativado. No caso do parâmetro m , foi testado com um intervalo entre 1 e 80, ao qual engloba os valores recomendados

¹O código fonte do SLIC usado está disponível em <http://ivrg.epfl.ch/research/superpixels>

pelo autor.

Para o IFT-SLIC, além do mesmo parâmetro opcional p herdado do SLIC, o mesmo também apresenta os parâmetros c , $alfa$ e $beta$. c é um parâmetro opcional que redefine a posição das sementes dos superpixels em cada iteração caso a mesma se encontre fora de seu superpixel, já $alfa$ e $beta$ tem o mesmo papel que m de definir a relação entre compacidade e aderência às bordas dos superpixels gerados.

Em ambos os métodos, o número de superpixels k é definido pelo tamanho que cada superpixel deverá ter. Nos testes, usa-se os tamanhos 10×10 , 20×20 , 30×30 e 40×40 , em que o número de superpixels é calculado como $k = N/A$, N é o número de pixels da imagem e A o tamanho do superpixel.

5.1 Bases de Imagens

Para os testes foram utilizadas 3 bases de imagens com seus gabaritos correspondentes:

- **Grabcut:**



Figura 5.1: Imagens da base de imagens Grabcut, com seus respectivos gabaritos.

Foram utilizadas as 50 imagens naturais dessa base de imagens para testar a flexibilidade dos métodos em cenários variados;

- **Liver**

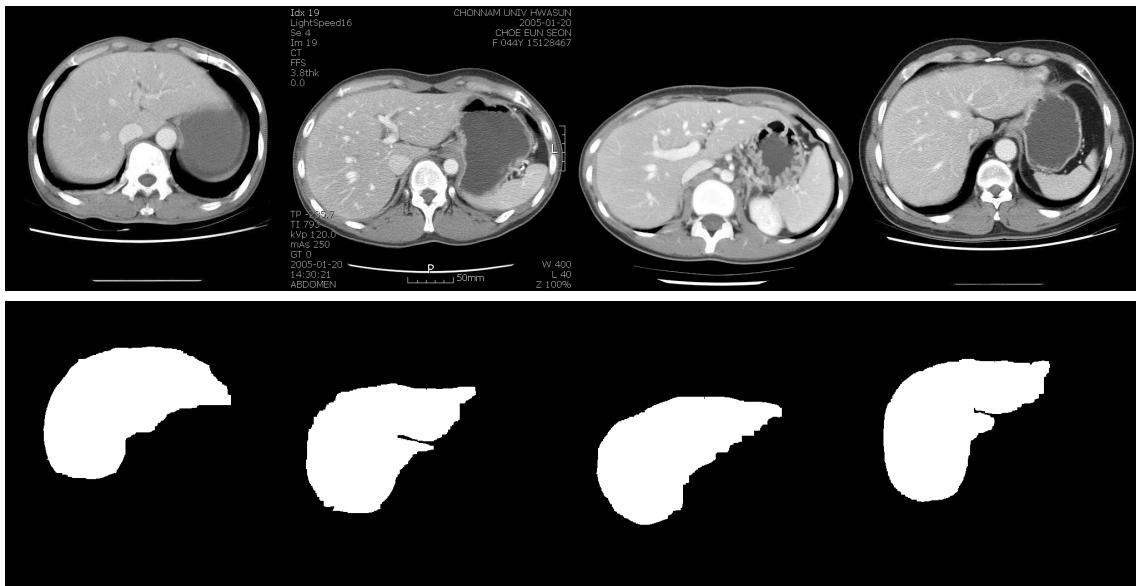


Figura 5.2: Imagens da base de imagens Liver, com seus respectivos gabinetos.

Essa base de imagens é composta por um total de 40 fatias de imagem de 10 exames de tomografia computadorizada com o objetivo de segmentação do fígado;

- **Talus**

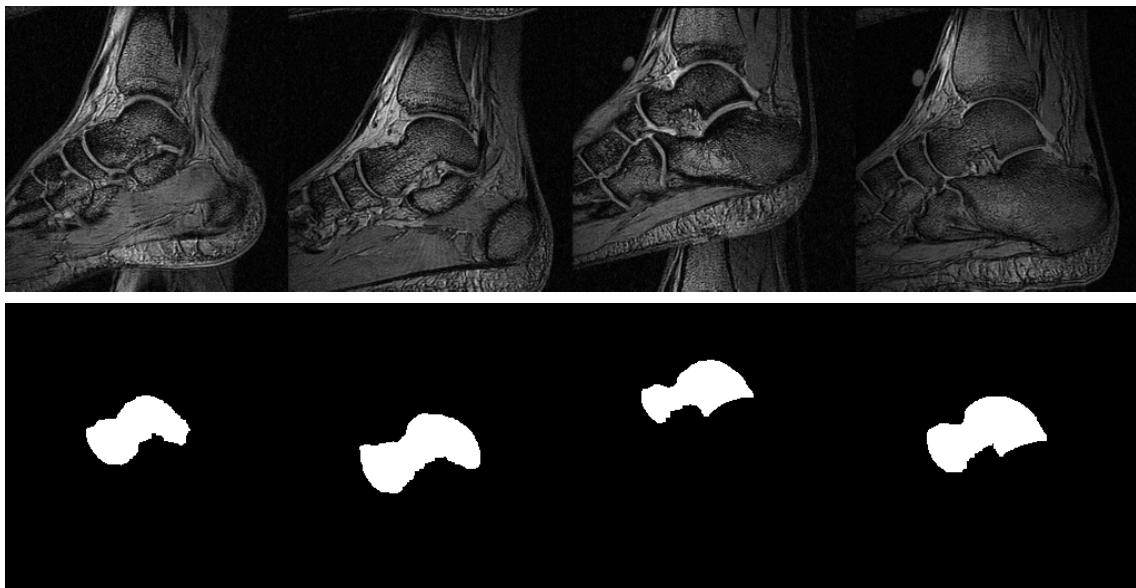


Figura 5.3: Imagens da base de imagens Talus, com seus respectivos gabinetos.

Por fim, essa base de imagens oferece 40 fatias de imagem de ressonância magnética do pé;

Para cada imagem das 3 bases de imagens, uma imagem de rótulo *R* foi gerada com cada combinação possível dos parâmetros apresentados na Tabela 5.1 de cada método.

5.2 Metodologia

Para medir a habilidade de um método em aderir as bordas dos objetos contidos em uma imagem, foi utilizada a imagem de rótulos *R* gerada por ambos IFT-SLIC e SLIC e a imagem de

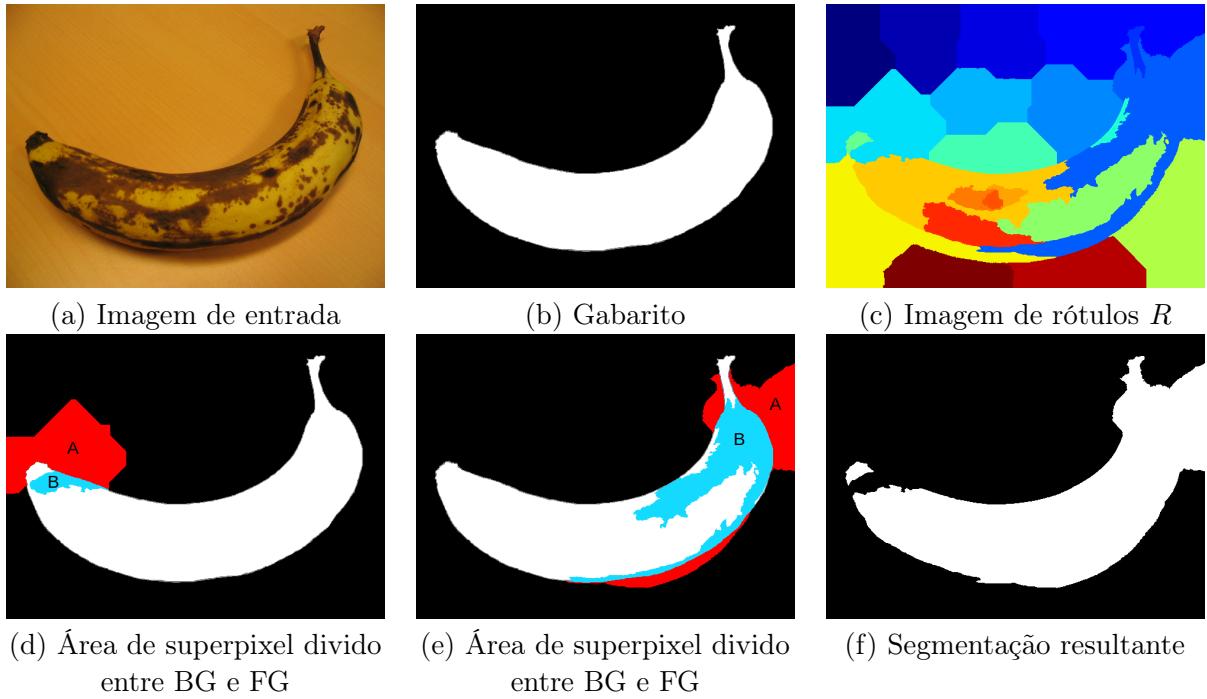


Figura 5.4: (a-b-c) Imagem de entrada, seu gabarito e imagem de rótulos. (d-e) Dividem o superpixel entre regiões A e B, na qual A representa a área do superpixel contida no fundo e B na frente. (f) A segmentação resultante.

gabarito da base de imagens. Primeiramente foram computados os superpixels usando o IFT-SLIC e SLIC. Para cada superpixel gerado foi atribuído a ele o rótulo mais frequente em seu interior do gabarito, como demonstrado na Figura 5.4. A segmentação resultante é então comparada com esse mesmo gabarito usando o coeficiente *Dice*. A Figura 5.5 ilustra esse processo passo a passo.

5.3 Análise de Parâmetros

Esta seção tem como objetivo discutir uma análise individual dos parâmetros de cada um dos métodos abordados, definindo seus efeitos, vantagens e desvantagens.

5.3.1 SLIC

Perturbação das Sementes - Parâmetro p

Como já abordado, o SLIC possui uma etapa opcional ao efetuar a distribuição de sementes, nas quais a posição delas pode ser modificada em uma vizinhança de 3×3 na direção do valor mais baixo do gradiente, com o intuito de evitar sementes em locais de muito ruído. A Figura 5.6 demonstra o efeito dessa etapa no posicionamento das sementes. O parâmetro p especifica o seu uso.

Nos Gráficos 5.7, 5.8 e 5.9 pode-se perceber que, apesar da recomendação do uso do parâmetro p pelo autor, tanto na base de imagens Liver, quanto na base de imagens Tálu, há uma leve degradação da acurácia na maioria dos pontos. Apenas na base de imagens Grabcut que o parâmetro p apresenta uma vantagem. Porém, como mostrado no Gráfico 5.10 que apresenta uma comparação média da acurácia do uso do parâmetro, no geral, o uso desse parâmetro no SLIC se torna prejudicial e portanto, é desativado.

Parâmetro m

Para o controle da relação entre a aderência de um superpixel à borda e sua compacidade, o SLIC possui o parâmetro m . Como mostrado na Figura 5.11, caso m seja um valor alto, seus superpixels

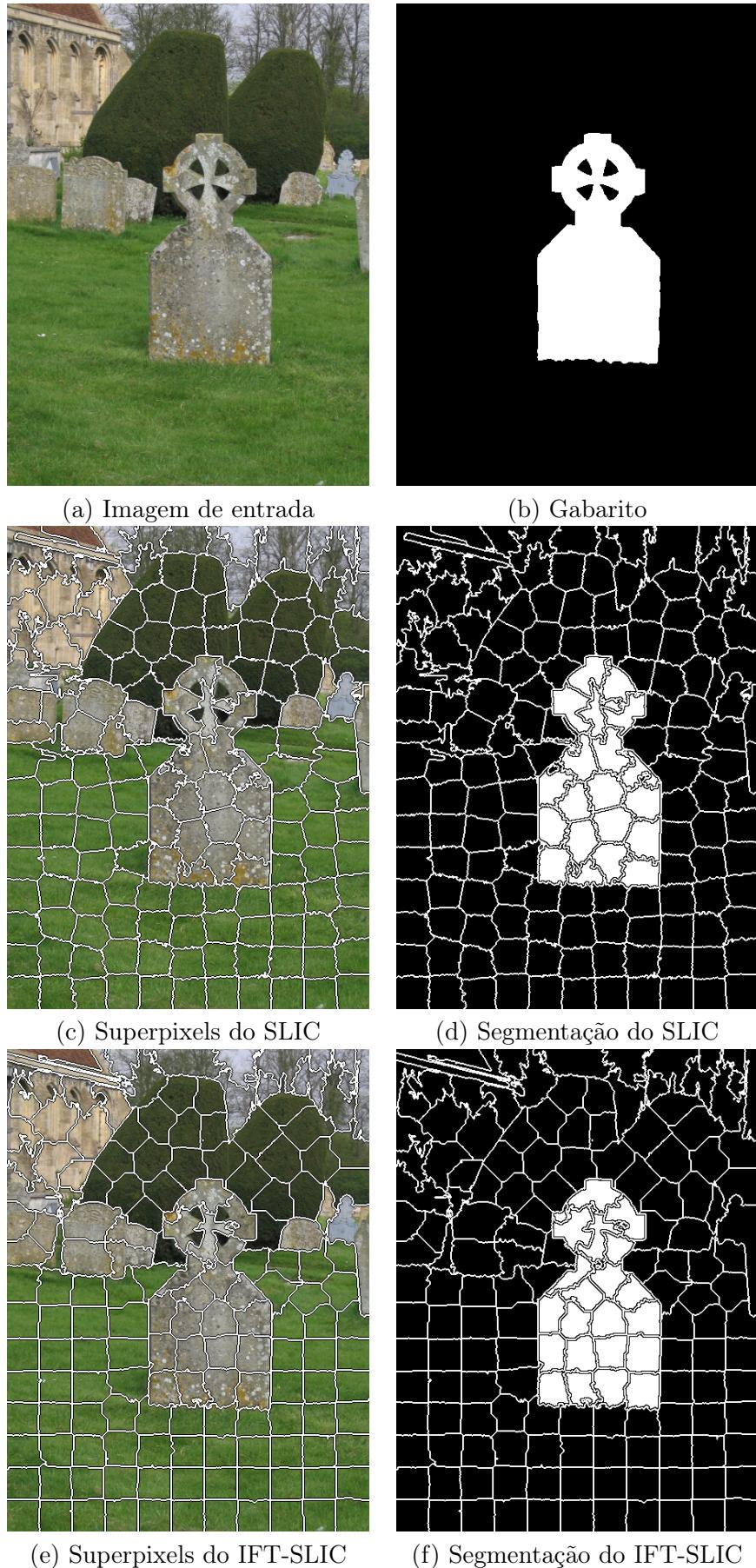


Figura 5.5: (a-b) Imagem de entrada e seu gabarito. Os superpixels de 40×40 são computados por: (c) SLIC e (e) IFT-SLIC. Foi atribuído para cada superpixel o rótulo mais frequente do gabarito ocorrendo em seu interior: (d) O resultado do SLIC tem Dice = 0.9659, e (f) A IFT-SLIC tem Dice = 0.9694.

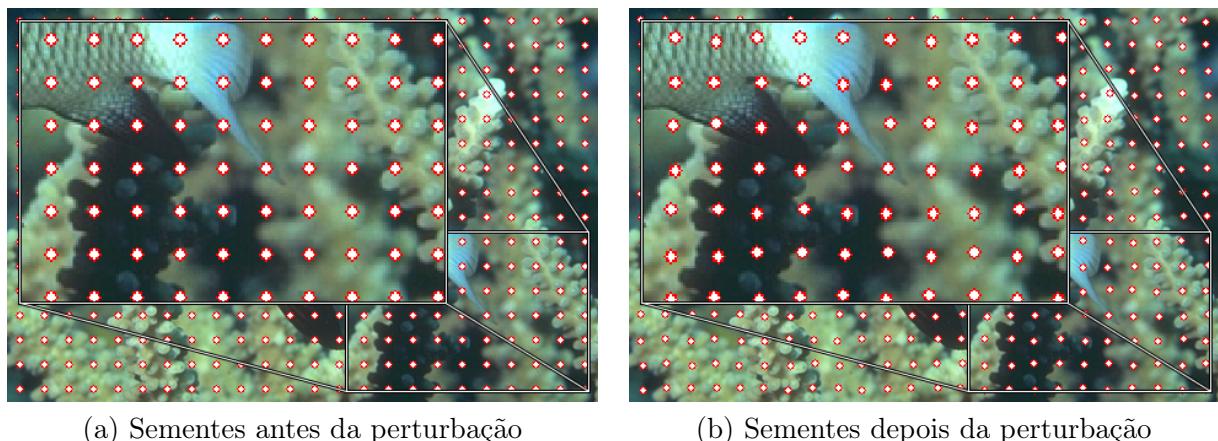


Figura 5.6: Demonstração do uso da etapa de perturbação das sementes.

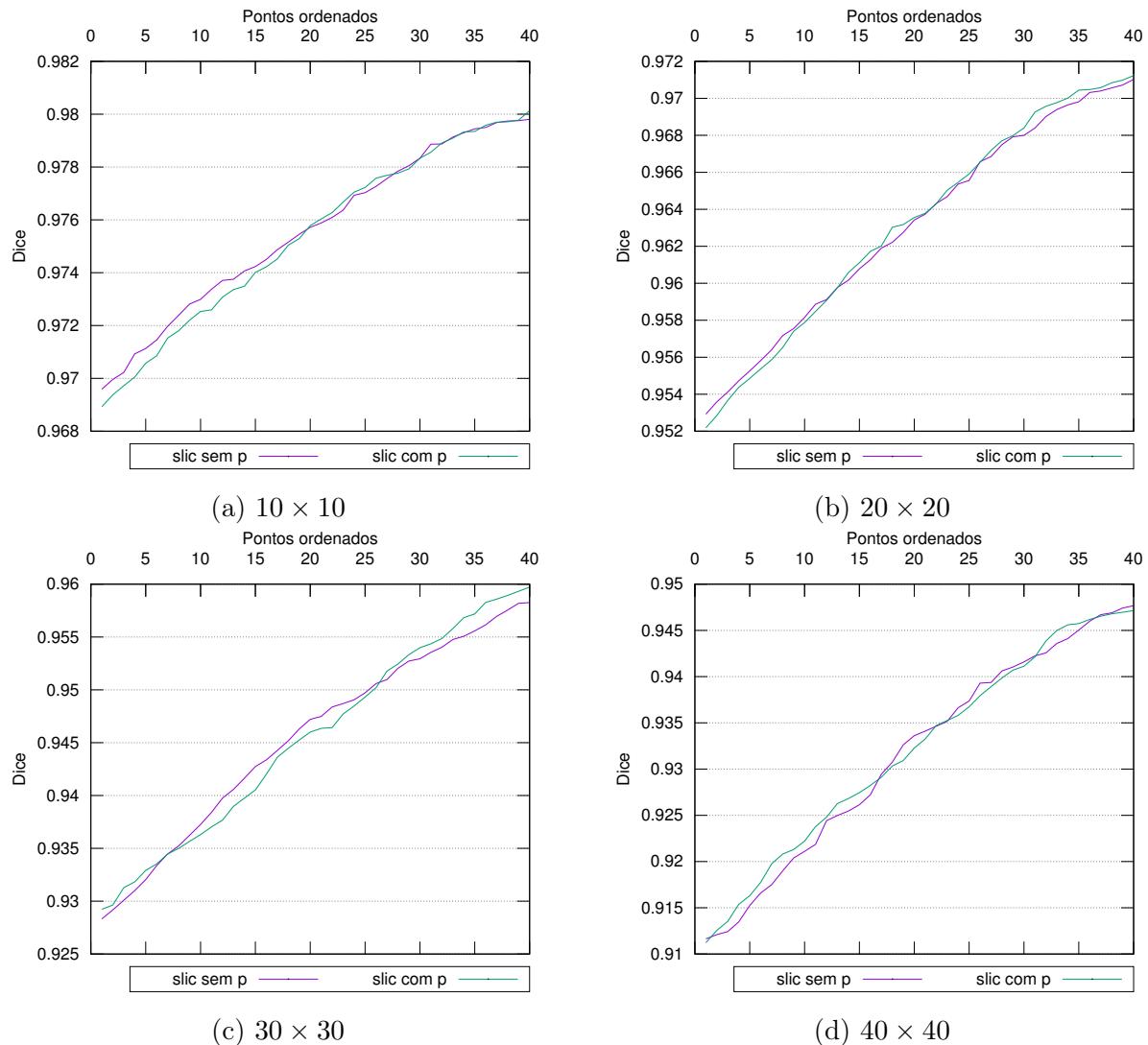


Gráfico 5.7: Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Grabcut.

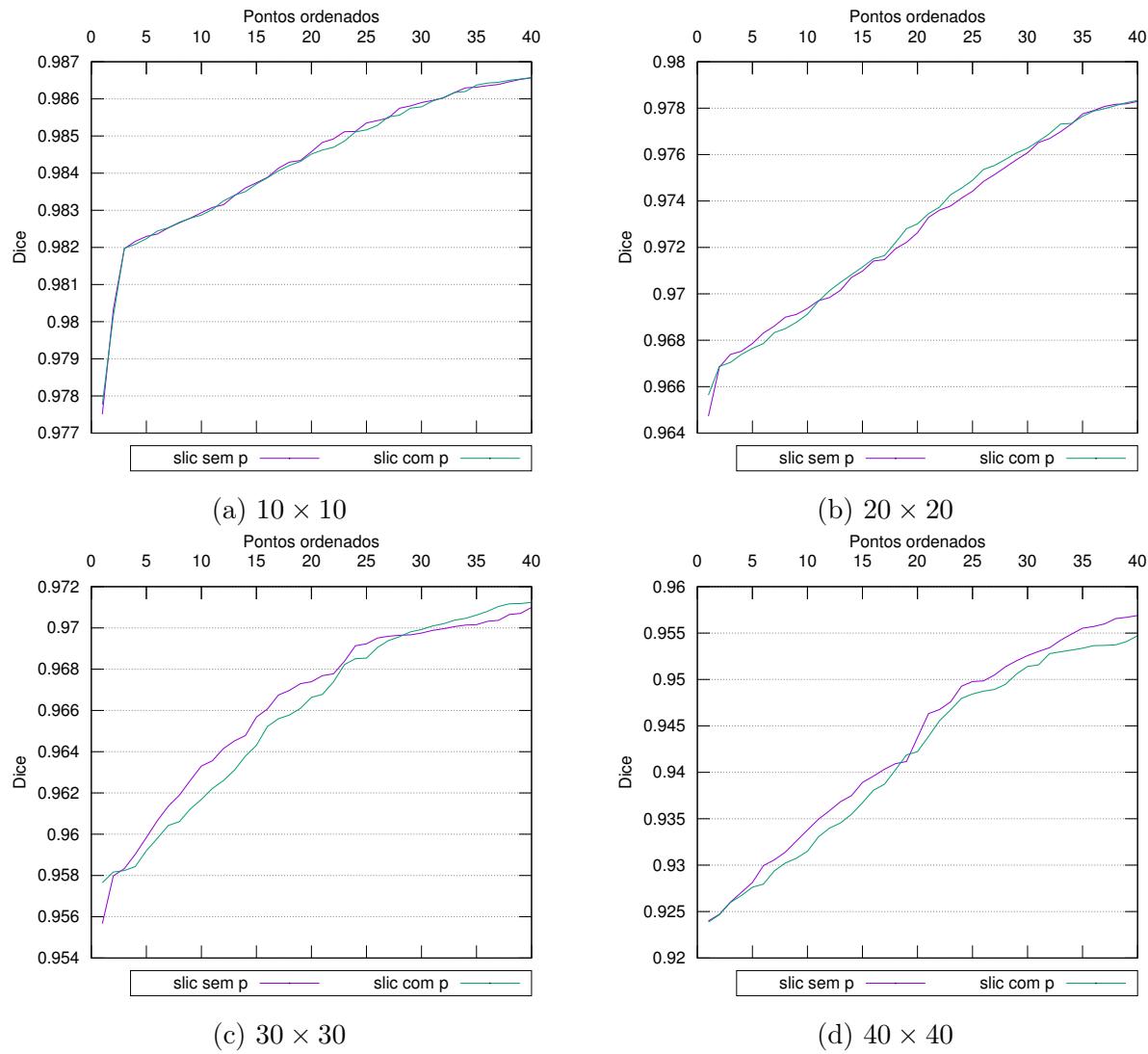


Gráfico 5.8: Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Liver.

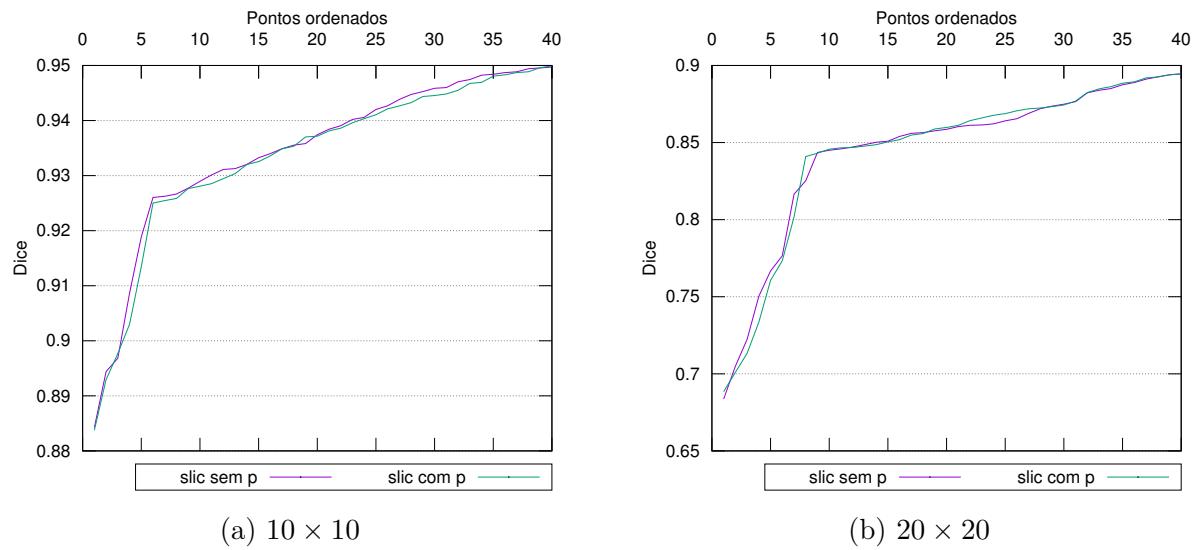


Gráfico 5.9: Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Talus.

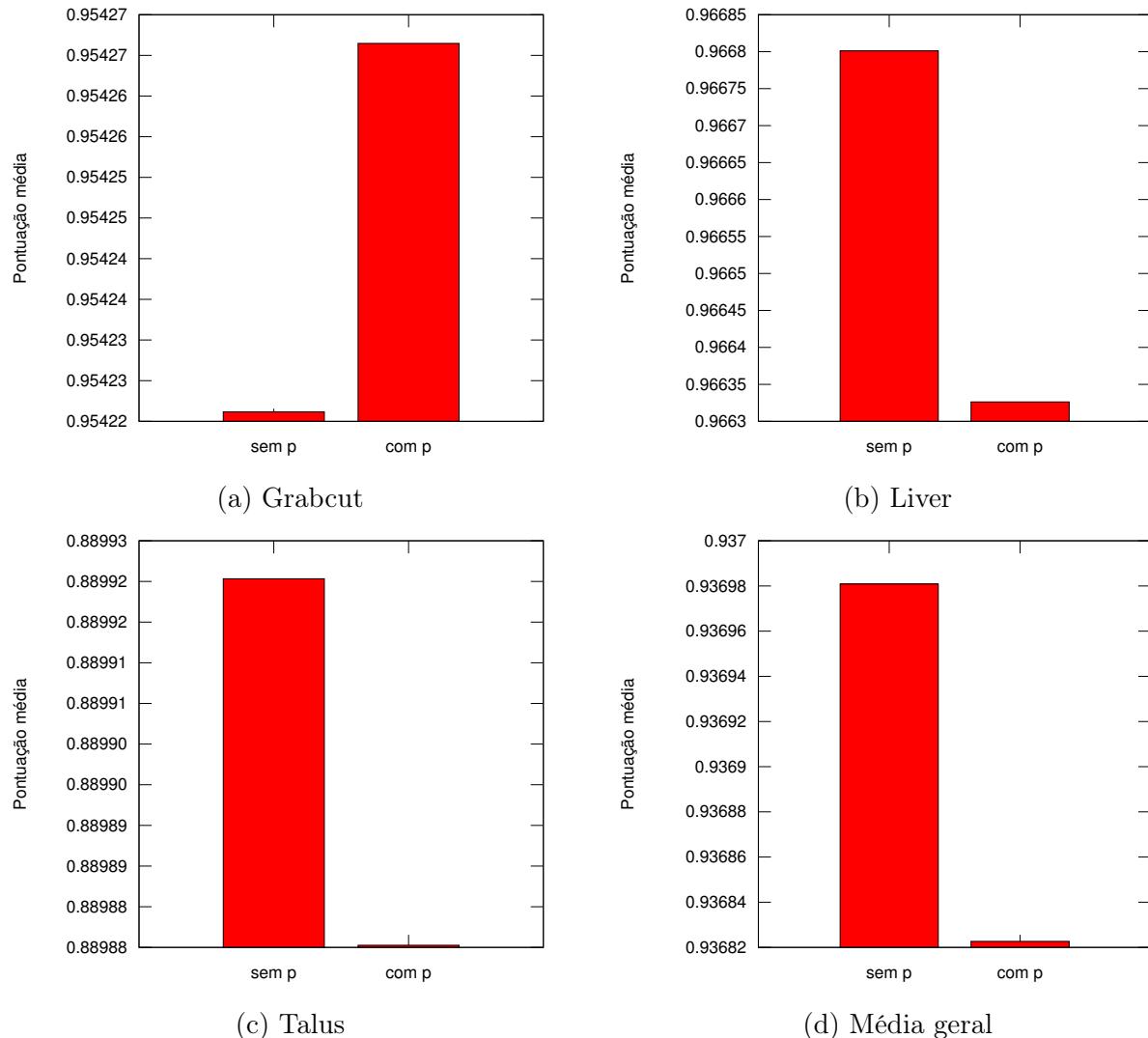


Gráfico 5.10: Comparação da média de acurácia do uso do parâmetro p ativado ou desativado no SLIC para as bases de imagens Grabcut (a), Liver (b) e Talus (c), além de sua média geral (d).

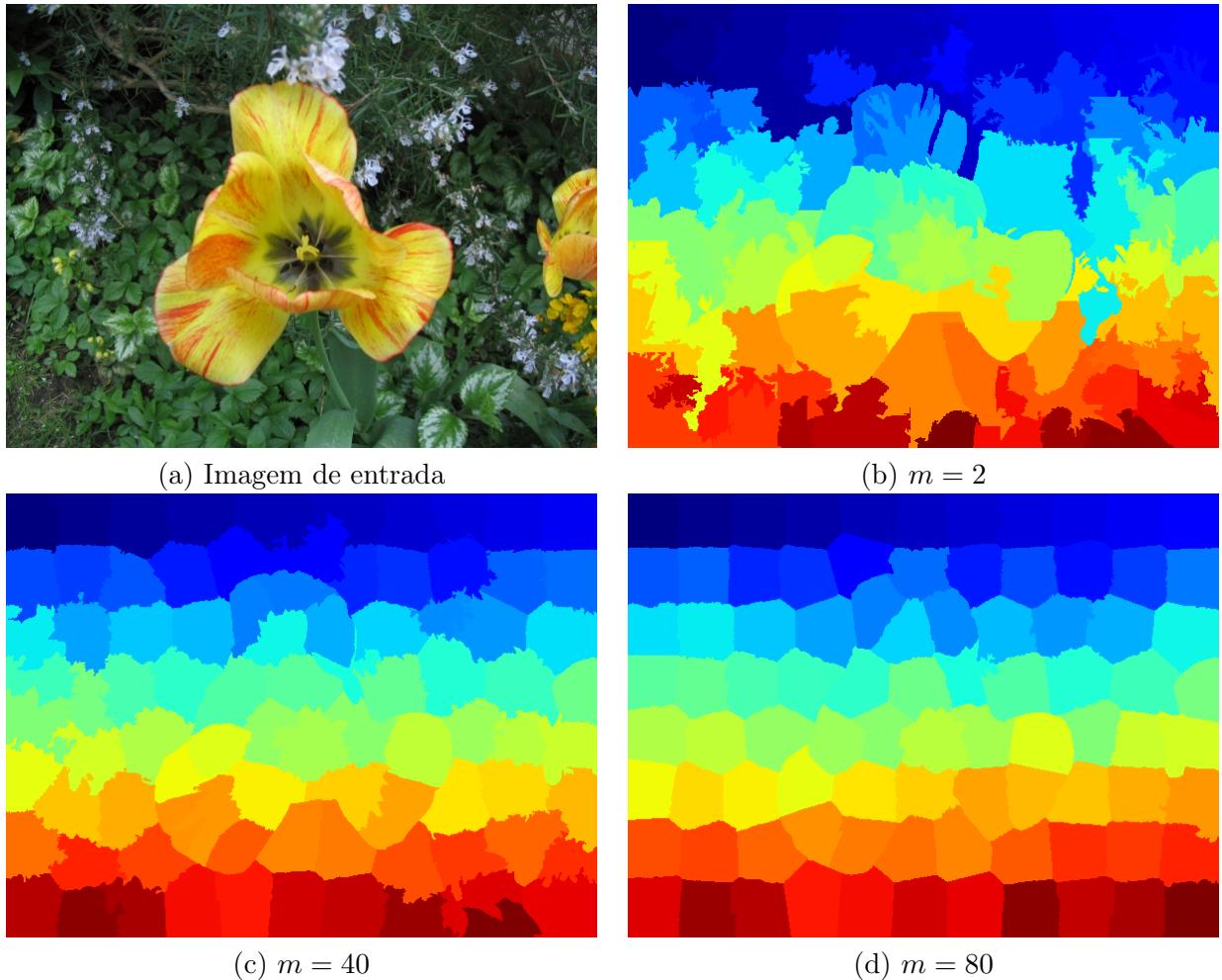


Figura 5.11: Demonstração do uso do parâmetro m .

resultantes serão mais compactos, caso contrário, os mesmos serão mais irregulares e aderentes às bordas da imagem.

Para os testes de m , foram utilizados os valores do intervalo entre 1 e 80.

5.3.2 IFT-SLIC

Perturbação das Sementes - Parâmetro p

A IFT-SLIC também contém o parâmetro p herdado do SLIC para o mesmo objetivo de perturbar as sementes iniciais.

No caso do IFT-SLIC os Gráficos 5.12, 5.13 e 5.14 demonstram que, diferentemente do SLIC, o uso do parâmetro p apresenta claramente seu benefício, especialmente no Gráfico 5.14 no qual há consideráveis melhorias. Para reforçar essa afirmação, foi mostrado novamente o gráfico de média de acurácia de p (Gráfico 5.15).

Centralização do Agrupamento - Parâmetro c

O IFT-SLIC permite a execução de um passo extra durante a etapa de atualização para a redefinição da posição do pixel semente de cada agrupamento para a próxima iteração. Esse passo tem como sua única finalidade garantir que o centro de um agrupamento não esteja fora do mesmo (ex. agrupamento com formato côncavo, Figura 5.16). O mesmo funciona analisando a distância de cada pixel do agrupamento em relação ao seu centro, selecionando como novo centro o pixel com a

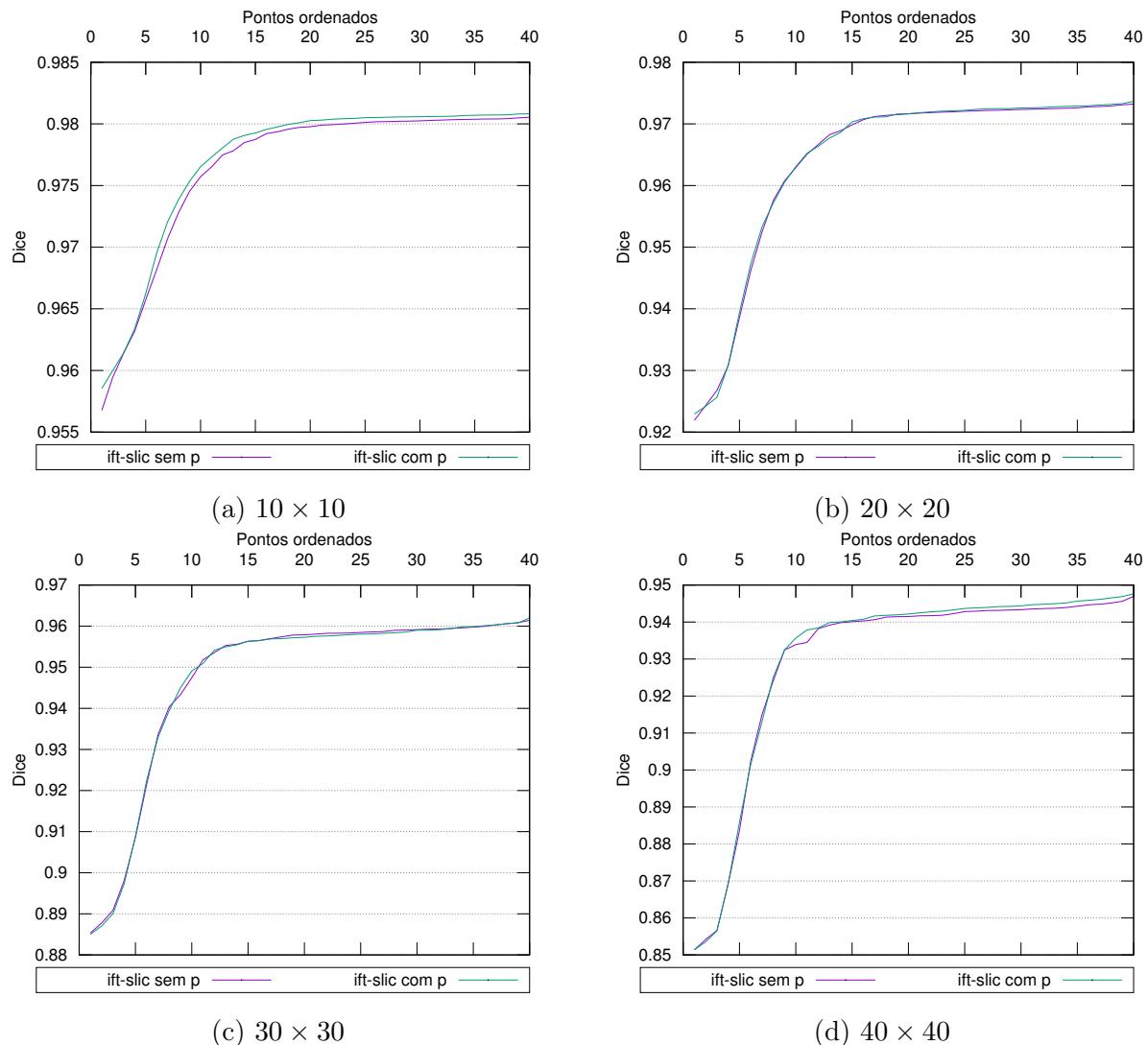


Gráfico 5.12: Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Grabcut.

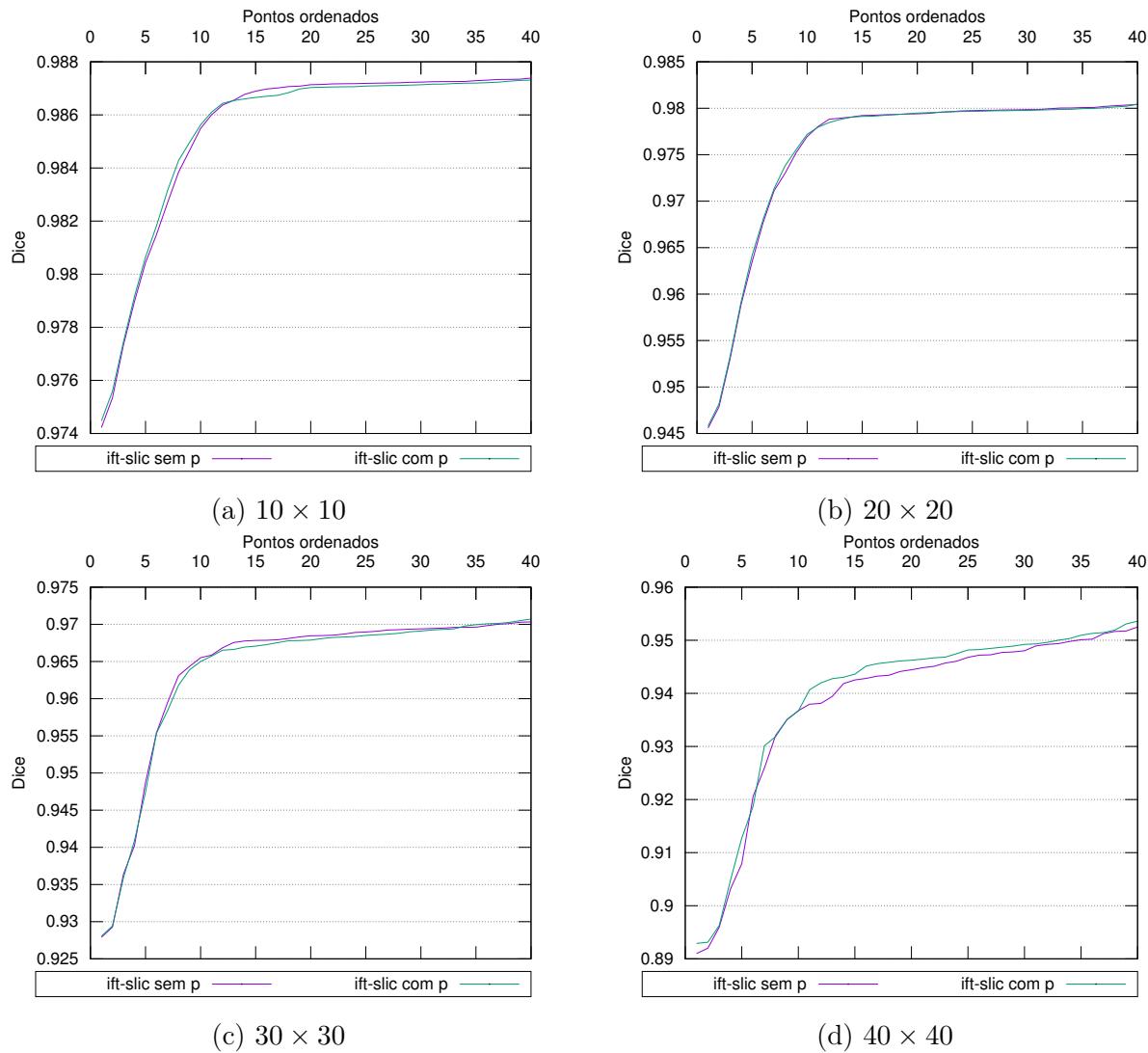


Gráfico 5.13: Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Liver.

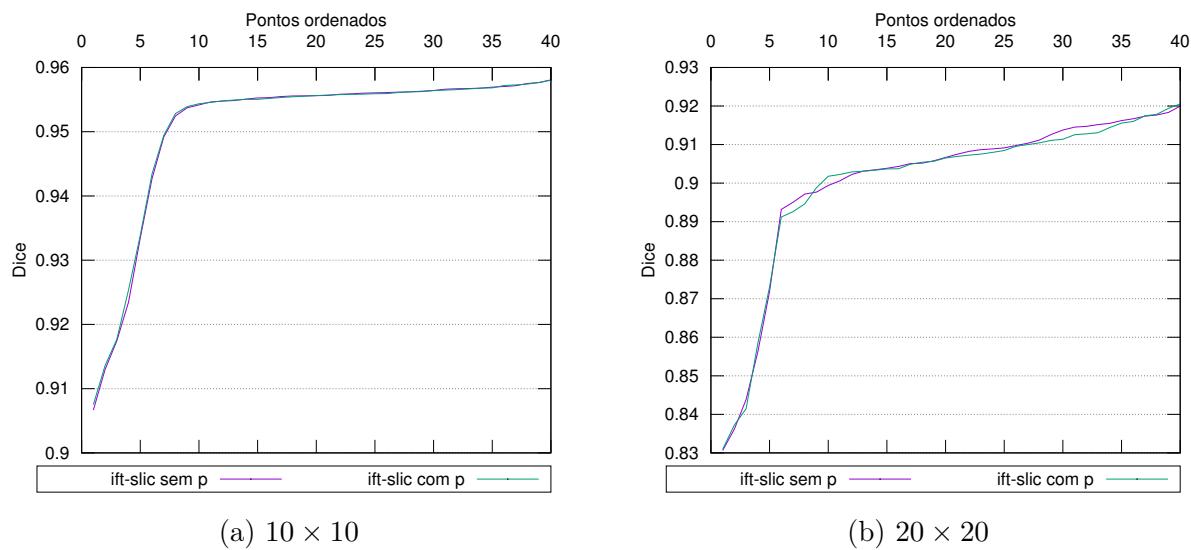


Gráfico 5.14: Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Talus.

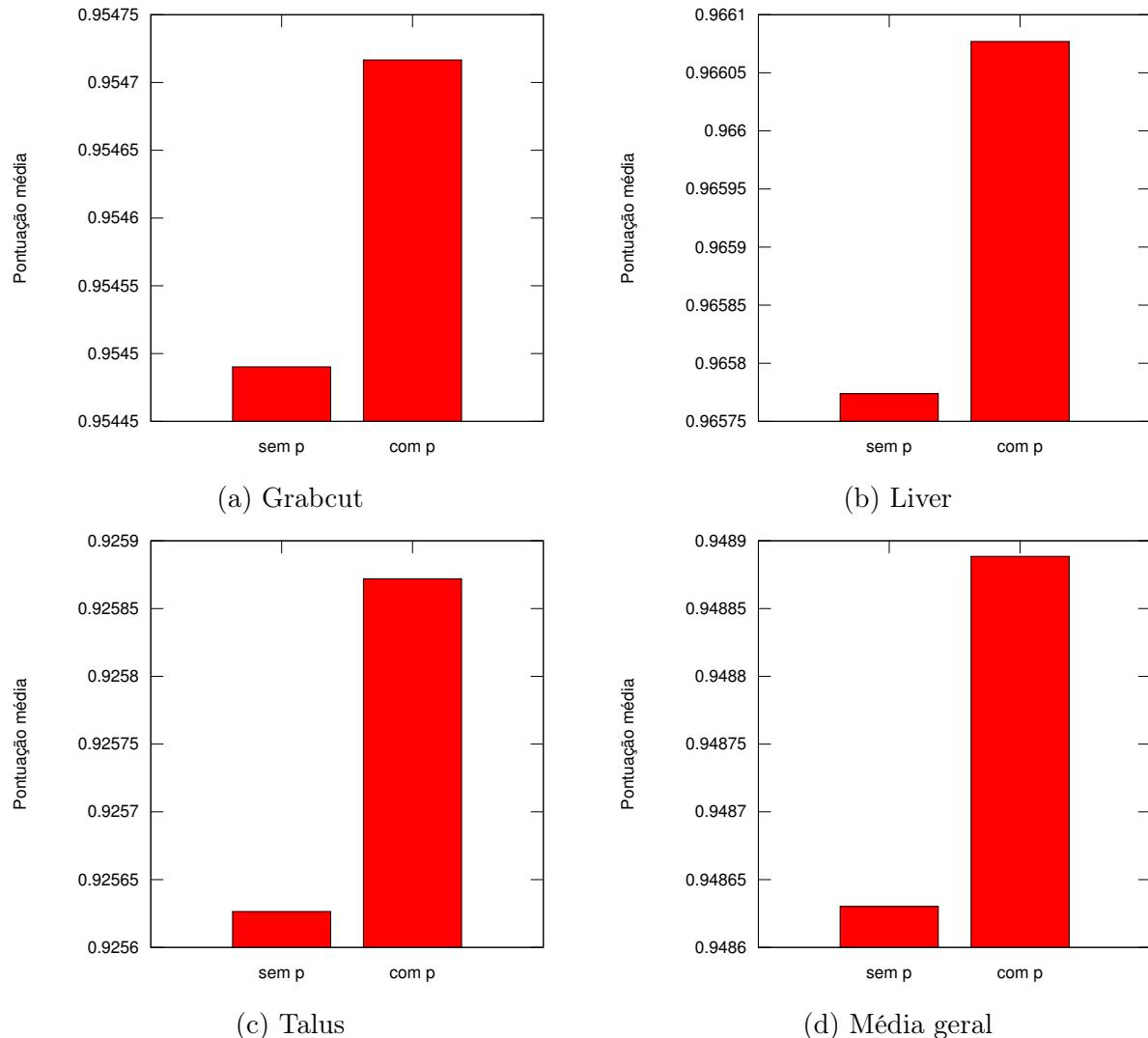


Gráfico 5.15: Comparação da média de acurácia do uso do parâmetro p ativado ou desativado para as bases de imagens Grabcut (a), Liver (b) e Talus (c), além de sua média geral (d).

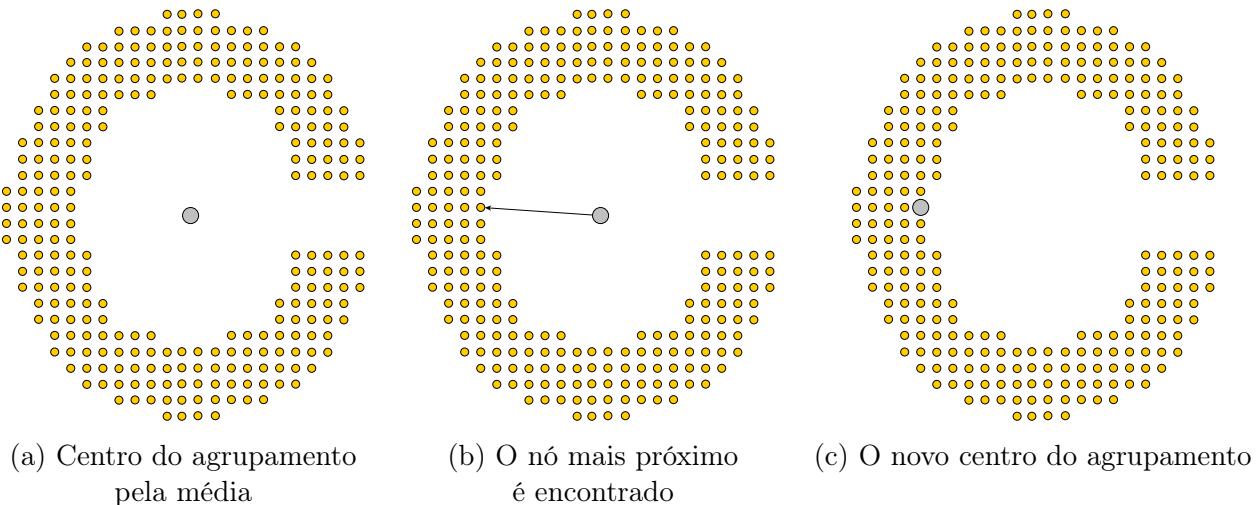


Figura 5.16: (a) Processo de seleção do centro de um agrupamento usado pelo SLIC. No exemplo, o centro acaba caindo em uma região fora do agrupamento. Em (b-c) é apresentada a solução usada pelo IFT-SLIC. Em (b) há uma varredura pelo nó pertencente ao agrupamento mais próximo do centro definido em (a). Após o nó ser encontrado, ele passa a ser o novo centro do agrupamento e portanto dentro do agrupamento como mostrado em (c).

menor distância encontrada².

Nos Gráficos 5.17, 5.18 e 5.19, é clara a vantagem de seu uso.

O parâmetro c também apresenta grande melhora conforme α cresce, pois sua acurácia também cresce com o uso de c . Isto ocorre já que quanto maior o valor de α , menos compacto o superpixel será, aumentando, assim, a chance de seu centro de agrupamento se posicionar fora do mesmo, acionando a redefinição da posição do centro deste agrupamento.

Parâmetro α e β

O controle de compacidade e acurácia no IFT-SLIC é feito através dos parâmetros α e β . Em ambos, quanto maior o valor, maior a acurácia e menor a compacidade. Seus valores devem ser definidos de acordo com o tipo de superpixel desejado. Para um superpixel mais compacto e com uma boa aderência às bordas, pode-se utilizar um β menor (ex. 3), e variar o α até a obtenção da acurácia desejada. A mesma lógica pode ser aplicada no caso em que a aderência é a prioridade: um β alto (ex. 30) gerará superpixels com alta acurácia, porém com baixa compacidade. Na Figura 5.20, foi demonstrado o efeito da mudança desses parâmetros nos superpixels gerados de uma imagem.

Para os testes, o objetivo foi a geração de superpixels com maior acurácia, porém mantendo uma compacidade razoável, portanto β foi fixado em 12, e α variado de 0.0025 até 0.2 com pulos de 0.0025.

5.4 Comparativos

Os Gráficos 5.21, 5.22 e 5.23 mostram as curvas de acurácia para as três bases de imagens para diferentes superpixels de tamanho A . Nota-se que o IFT-SLIC, a partir do valor 0.08, obtém resultados consistentes de acurácia, especialmente para superpixels de tamanho 10×10 , 20×20 e 30×30 . Essa consistência permite uma escolha mais segura do valor de α sem que o mesmo apresente deterioração em aplicações diferentes. No caso do SLIC, essa consistência já não existe, com resultados variando conforme a base de imagens usado.

²Esse passo não mudará o centro do agrupamento quando o mesmo se encontrar dentro do agrupamento, afinal sua distância já será a mínima possível.

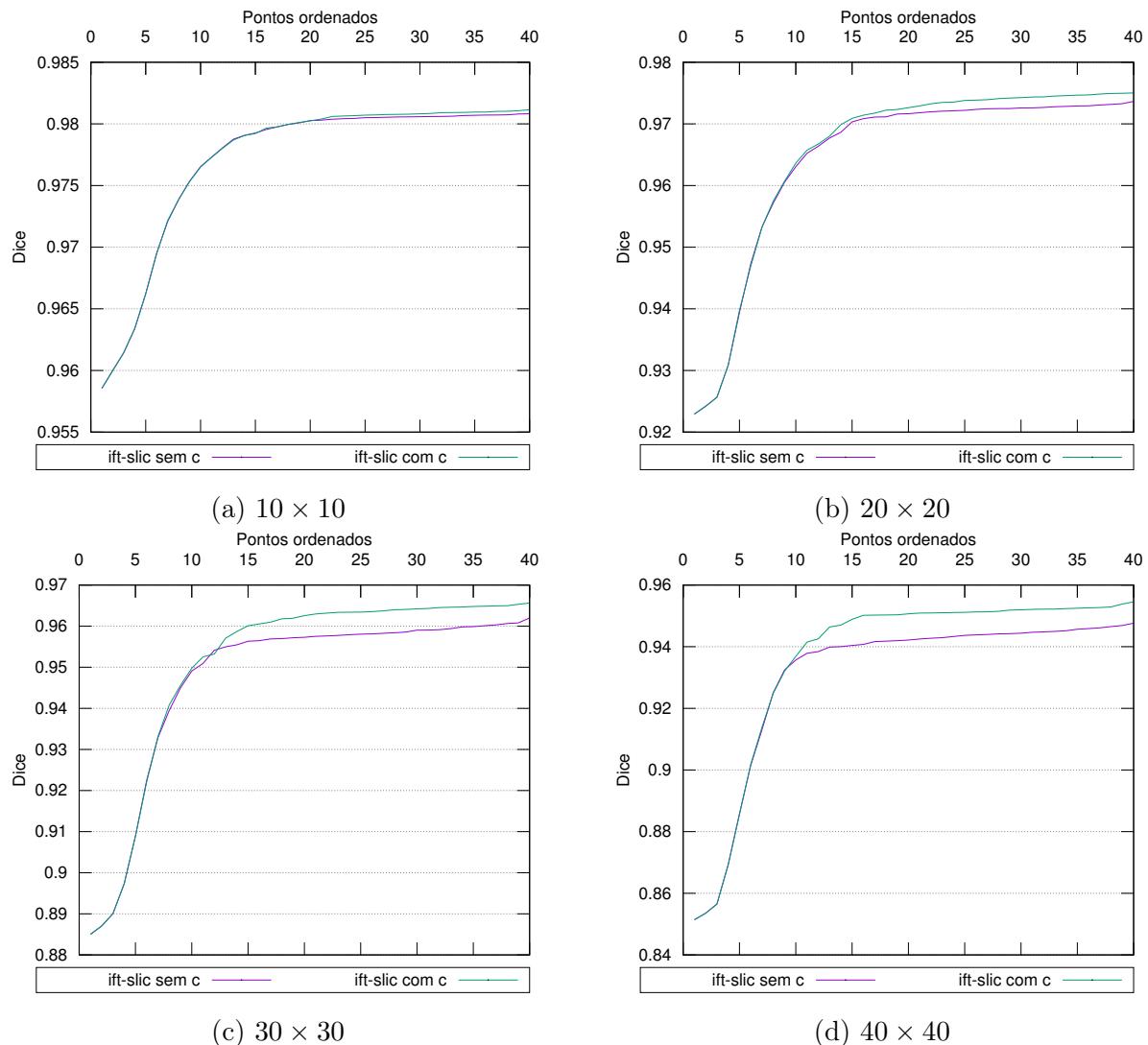


Gráfico 5.17: Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Grabcut.

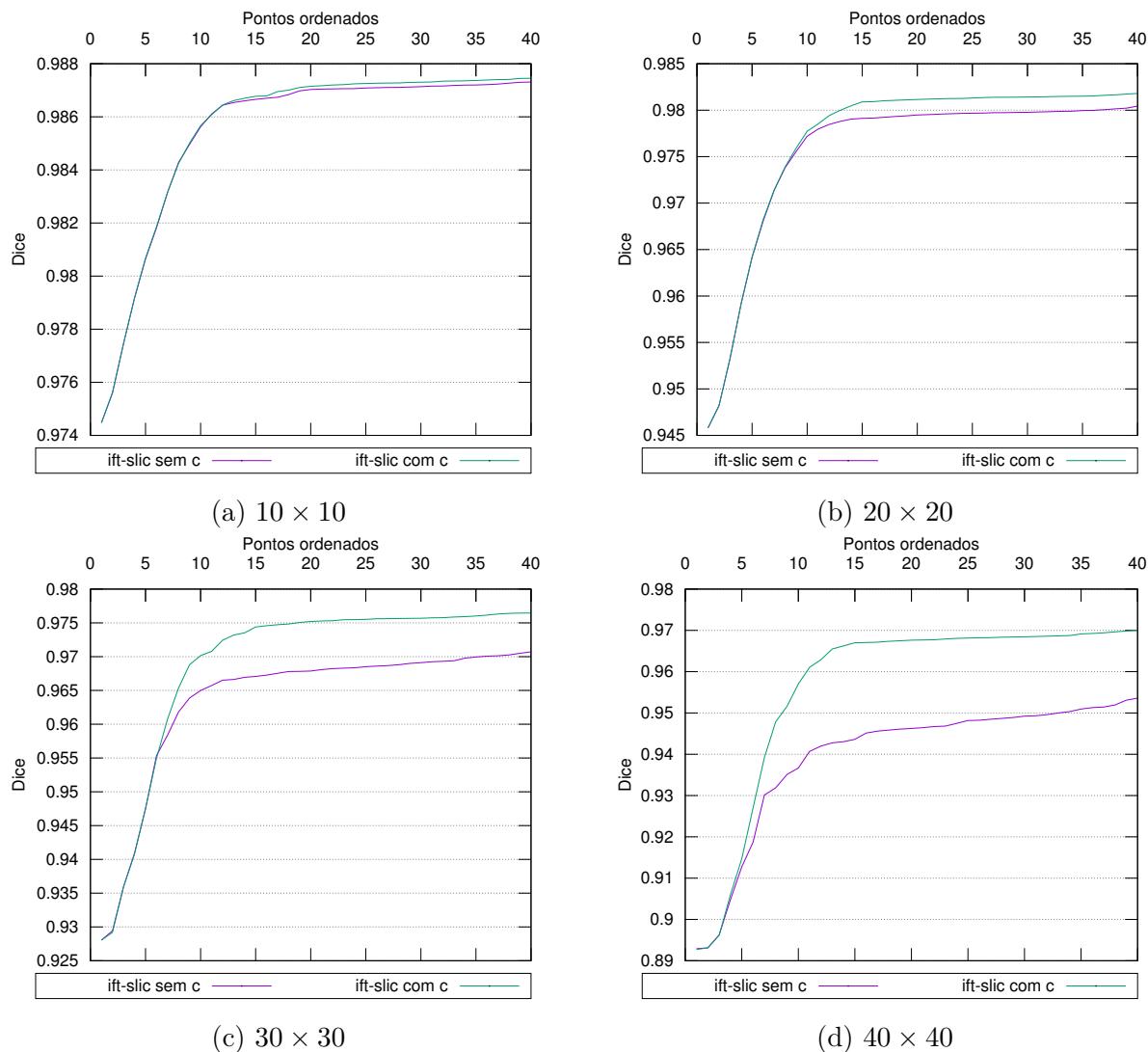


Gráfico 5.18: Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Liver.

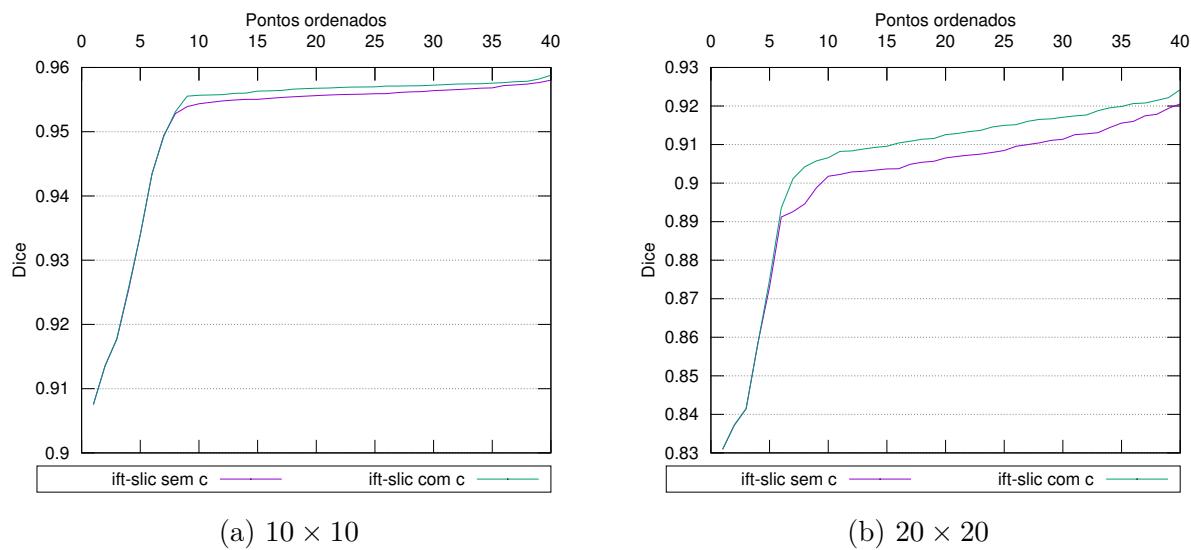


Gráfico 5.19: Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Talus.

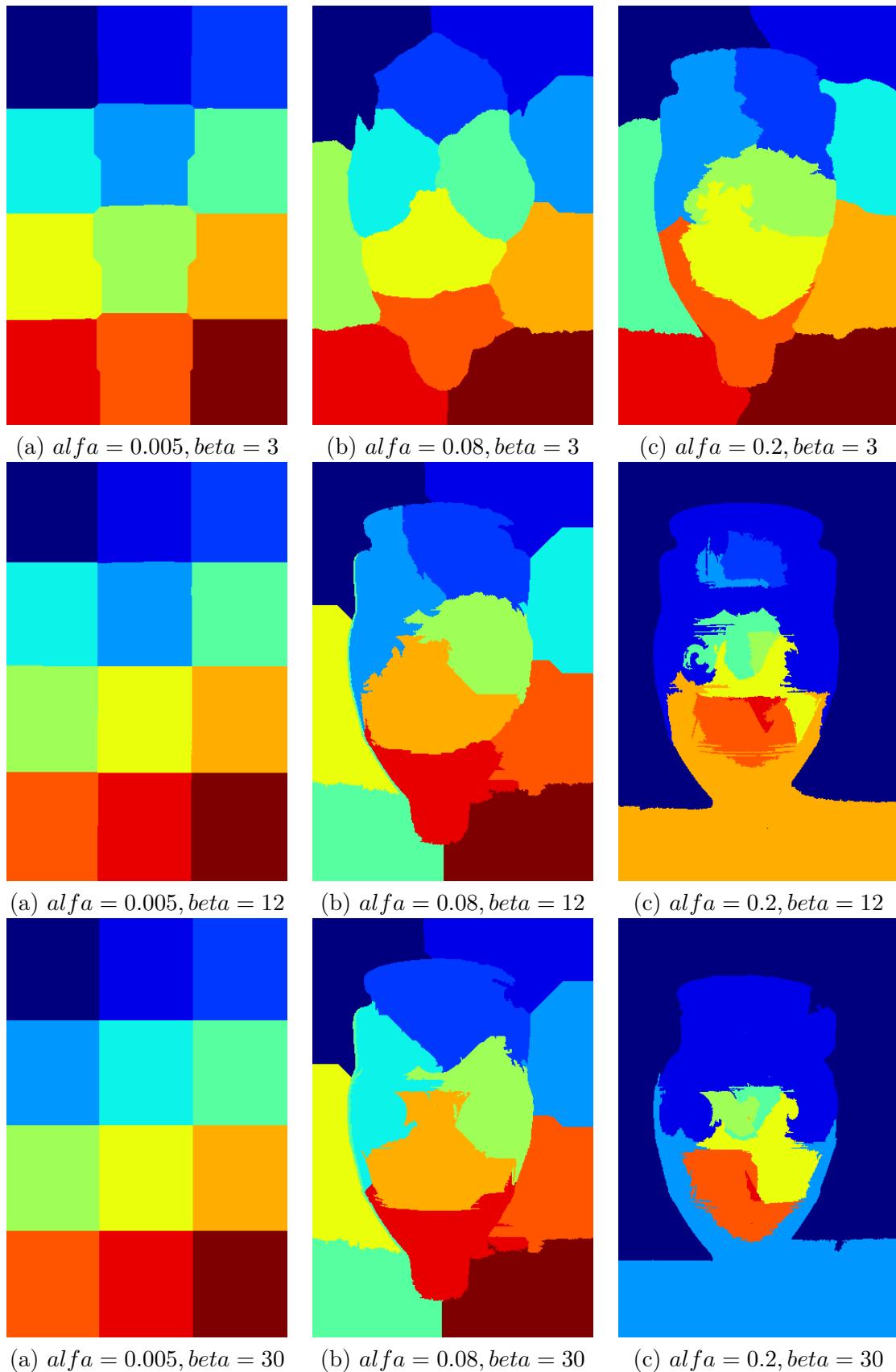


Figura 5.20: Demonstração da combinação dos parâmetros α e β .

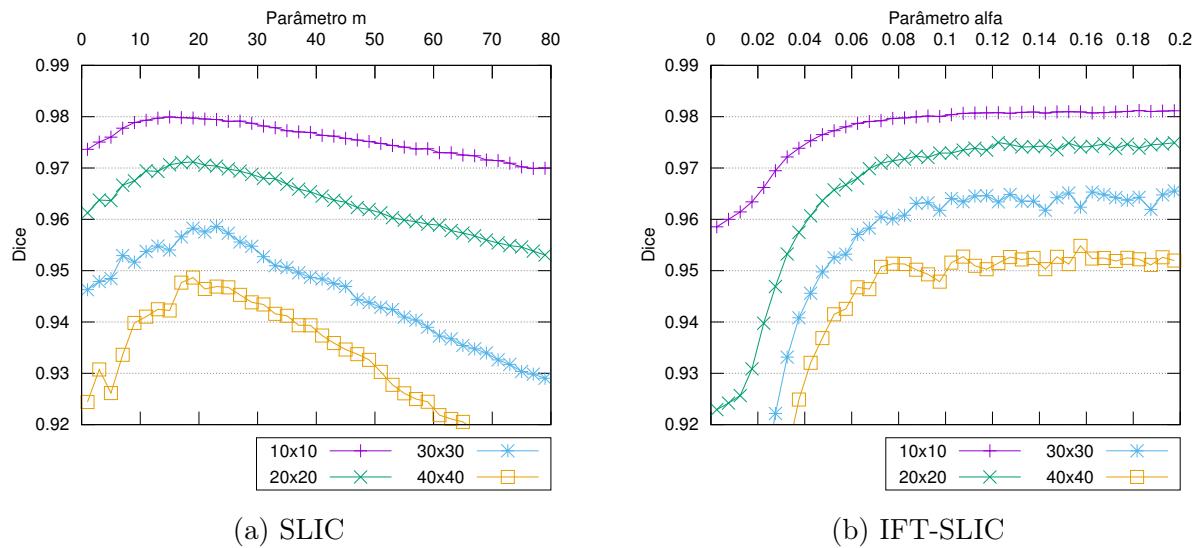


Gráfico 5.21: As curvas de acurácia média de segmentação da base de imagens Grabcut para superpixels de diferente tamanhos.

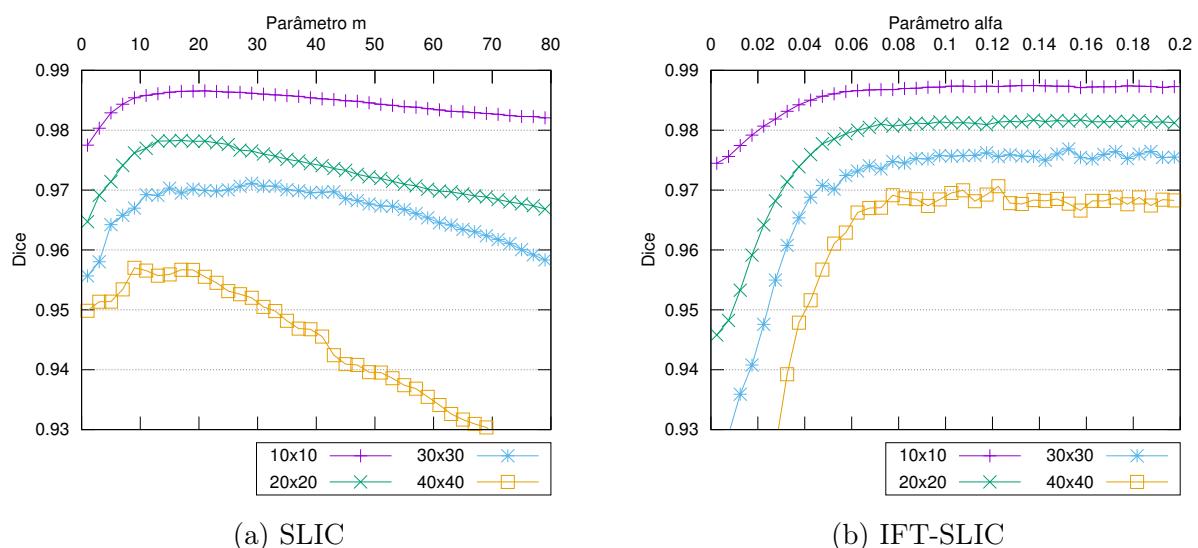


Gráfico 5.22: As curvas de acurácia média de segmentação da base de imagens Liver para superpixels de diferente tamanhos.

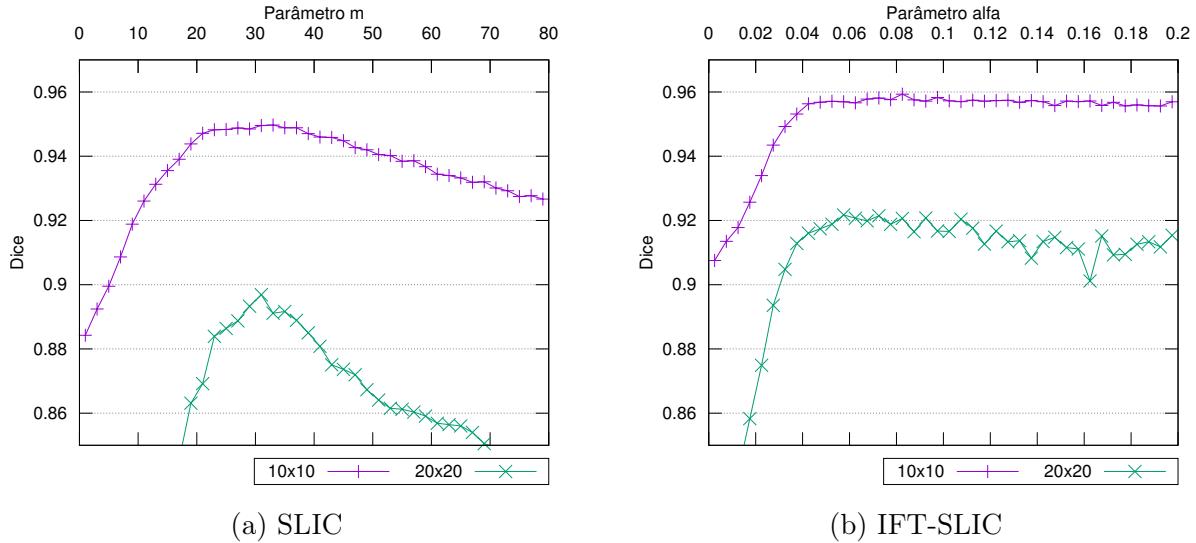


Gráfico 5.23: As curvas de acurácia média de segmentação da base de imagens Talus para superpixels de diferente tamanhos.

Comparando os resultados de ambos os métodos, fica claro que o IFT-SLIC apresenta uma melhor performance independentemente do tamanho de superpixel usado. Para melhor elucidar essa afirmação, nos Gráficos 5.24, 5.25 e 5.26 foram mostrados essas mesmas curvas, em ordem crescente de acurácia, de ambos os métodos no mesmo gráfico.

Nas Figuras 5.27, 5.28 e 5.29 foram apresentadas algumas imagens demonstrando a diferença de segmentação entre o IFT-SLIC e SLIC. Nas Figuras 5.27 e 5.28 percebe-se que ambos os algoritmos obtém uma pontuação alta com o *Dice*, porém fica claro a dificuldade do SLIC de seguir o contorno de objetos complexos ao analisarmos o topo da cruz na Figura 5.27c e os pequenos objetos na Figura 5.28c. Ambas falhas de segmentação não ocorrem ou são significantemente reduzidas com o IFT-SLIC. Na Figura 5.29 é complicado de analisar qual segmentação é superior, dada a natureza inhomogênea dessa, porém a pontuação *Dice* obtida mostra uma grande superioridade da segmentação apresentada pelo IFT-SLIC.

Além de analisar a acurácia de ambos os métodos, é importante também a análise de sua compacidade, porém, como um superpixel com alta acurácia tende a produzir uma baixa compacidade, o correto é analisar o quanto compacto o superpixel gerado pelo método é dado a sua acurácia.

Como o SLIC permite o controle entre acurácia e compacidade via o parâmetro *m*, foi mantido o seu intervalo de valores entre 1 até 80 conforme recomendação do autor. No caso do IFT-SLIC pode-se alterar tanto os valores de *alfa* quanto de *beta* permitindo uma maior flexibilidade nas escolhas dos valores. Sendo que nos testes anteriores *beta* continha um valor (*beta* = 12) otimizado para geração de superpixels com maior acurácia em detrimento de sua compacidade, nesses testes, foi usado *beta* = 2 que proporcionará superpixels mais balanceados.

Essa análise é apresentada pelos Gráficos 5.30, 5.31 e 5.32. Nas, o IFT-SLIC consegue manter uma vantagem em relação ao SLIC. Conforme o tamanho dos superpixels aumentam, essa vantagem também cresce já que o SLIC não consegue manter uma boa acurácia. A maioria dos pontos do IFT-SLIC se concentram na área superior à direita do gráfico, à qual representa o melhor balanço obtido entre compacidade e acurácia. No caso do SLIC, seus pontos tendem a se aglomerar na área superior no centro, demonstrando a dificuldade do SLIC em obter superpixels aderentes às bordas. Um caso interessante acontece na base de imagens Talus no Gráfico 5.32 no qual o SLIC apresenta todos os seus pontos com uma taxa inferior tanto em acurácia quanto na compacidade ao IFT-SLIC.

Além disso, o IFT-SLIC não apresenta uma grande mudança de pontuação conforme o tamanho dos superpixels aumentam, permitindo o seu uso com menos superpixels por imagem.

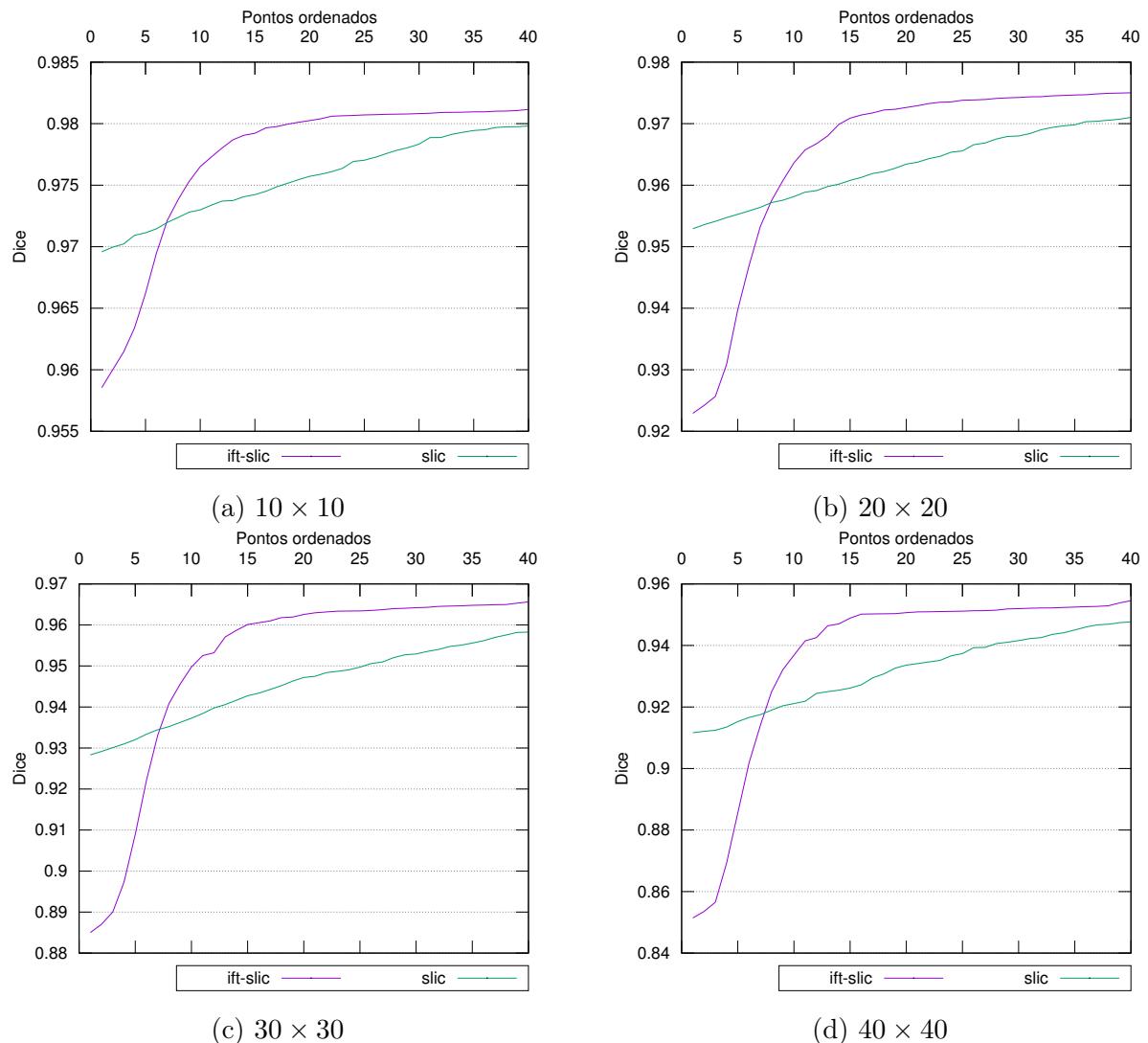


Gráfico 5.24: Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Grabcut.

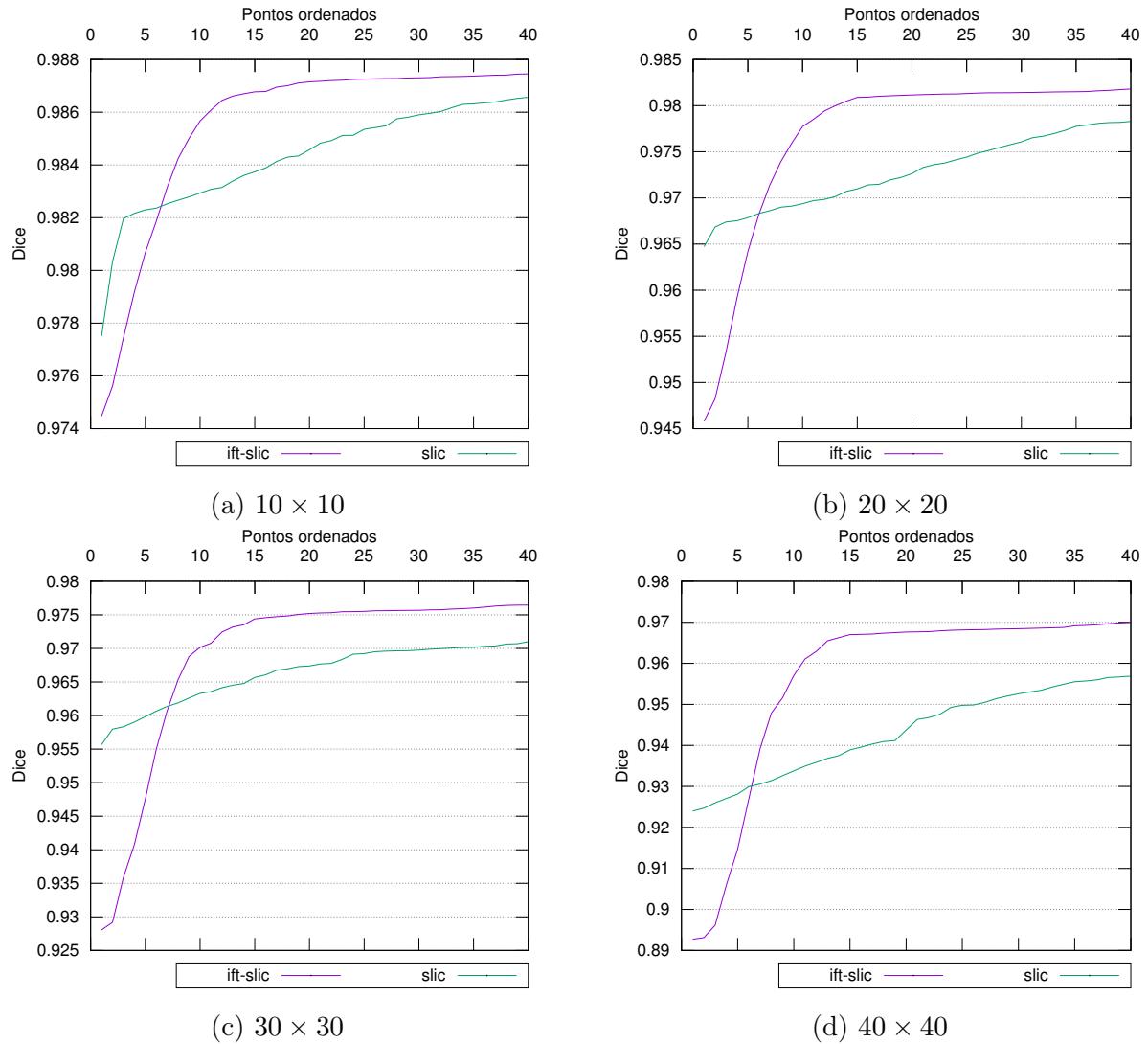


Gráfico 5.25: Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Liver.

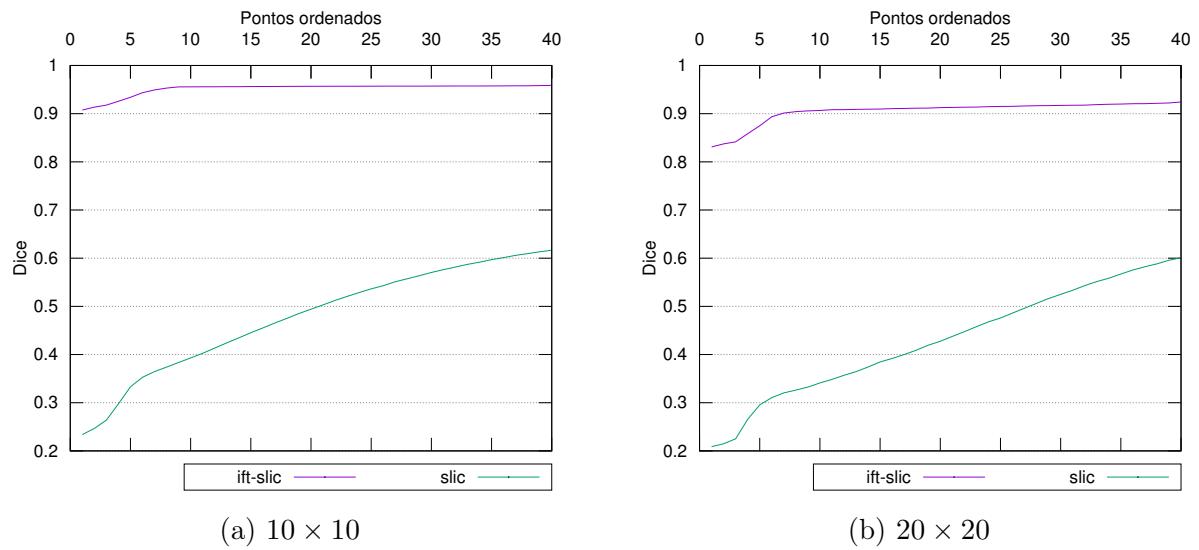
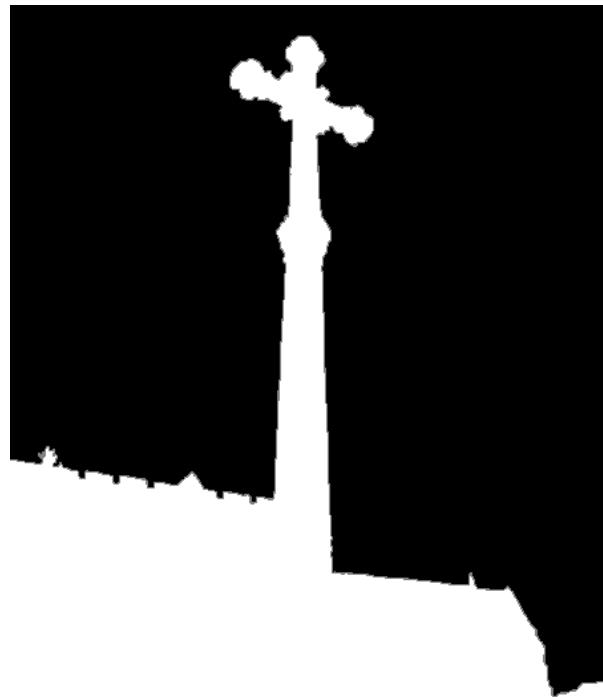


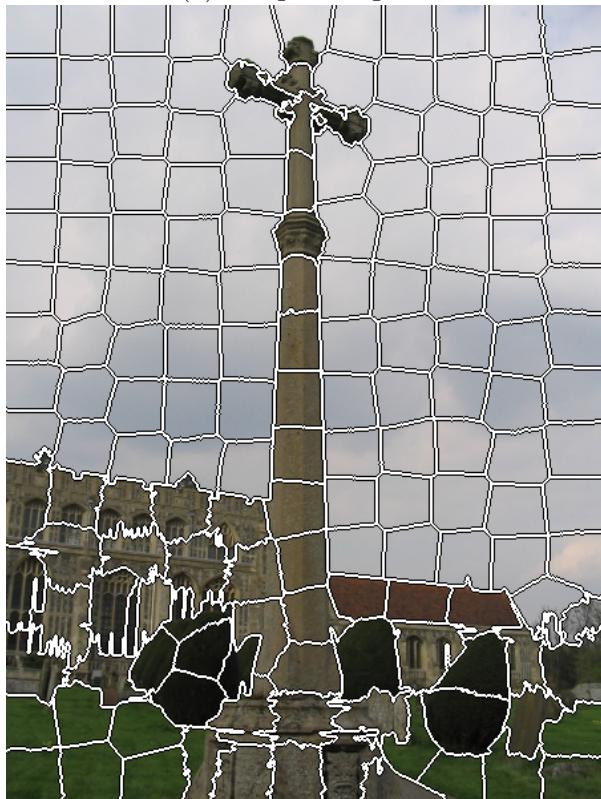
Gráfico 5.26: Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Talus.



(a) Imagem original



(b) Gabarito



(c) SLIC



(d) IFT-SLIC

Figura 5.27: Exemplos de segmentação da base de imagens Grabcut. A pontuação Dice do SLIC (c) é 0.98817 e do IFT-SLIC (d) é 0.990667.

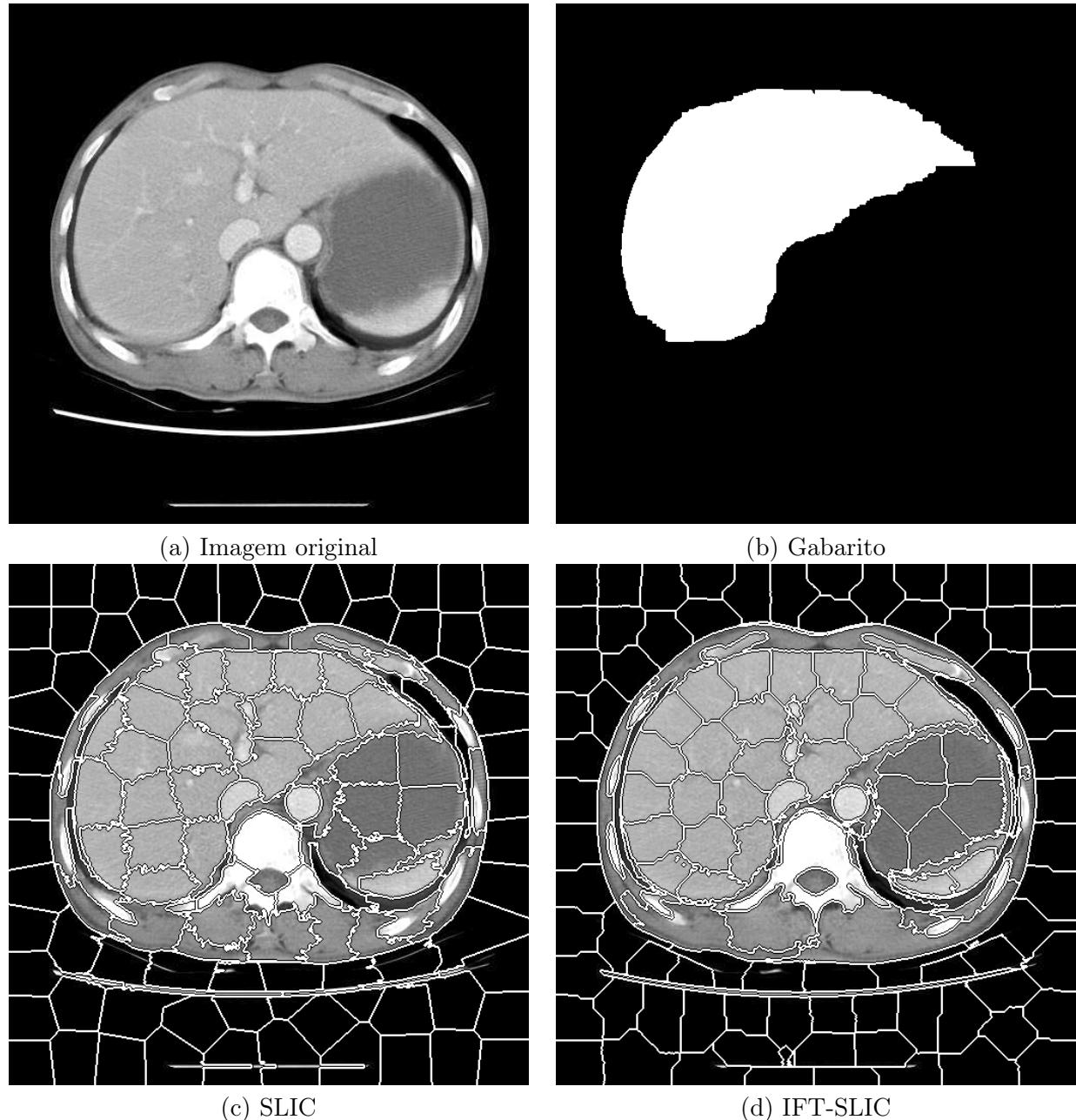


Figura 5.28: Exemplos de segmentação da base de imagens Liver. A pontuação Dice do SLIC (c) é 0.974979 e do IFT-SLIC (d) é 0.975518.

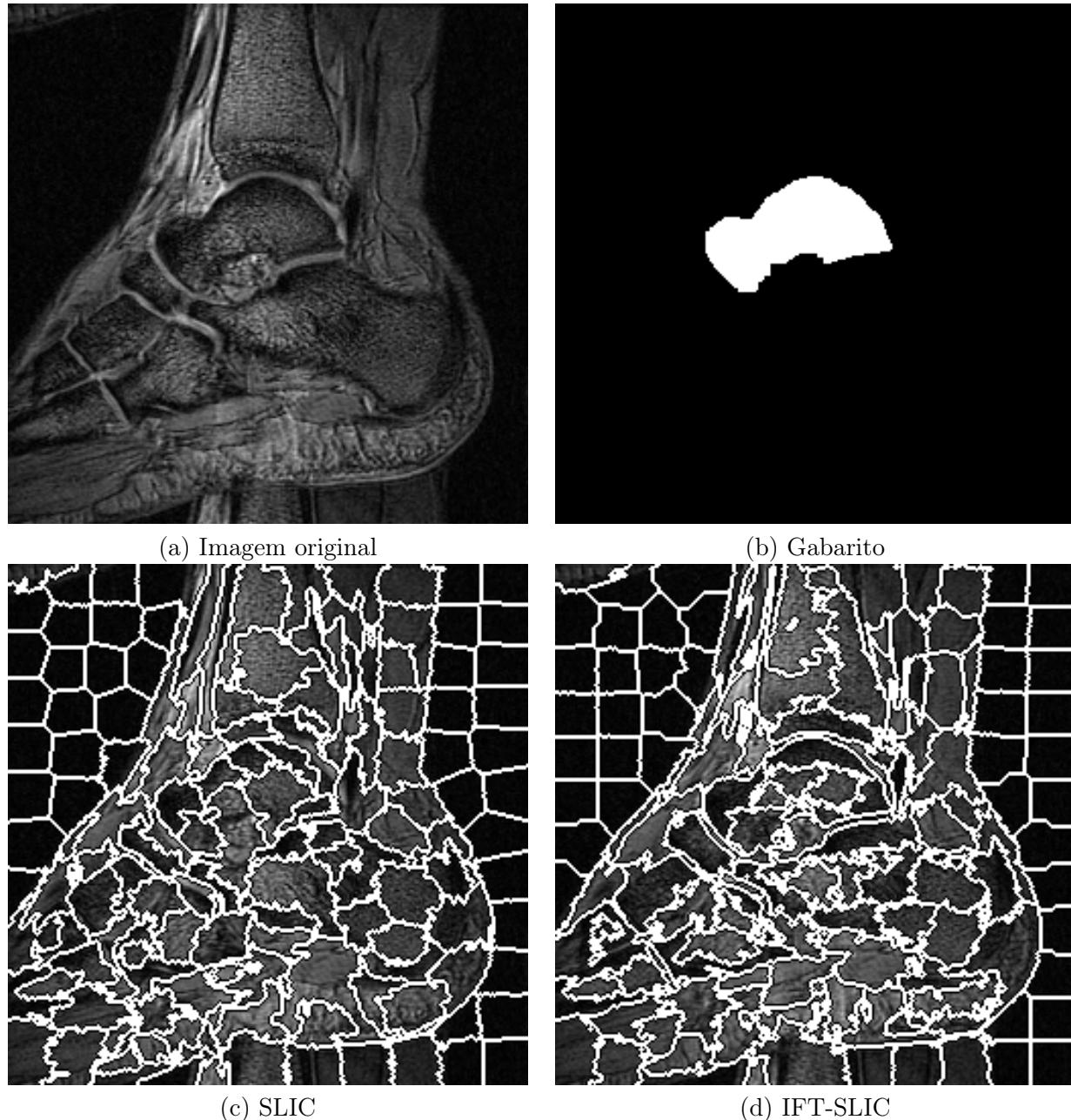


Figura 5.29: Exemplos de segmentação da base de imagens Tálus. A pontuação Dice do SLIC (c) é 0.854148 e do IFT-SLIC (d) é 0.961013.

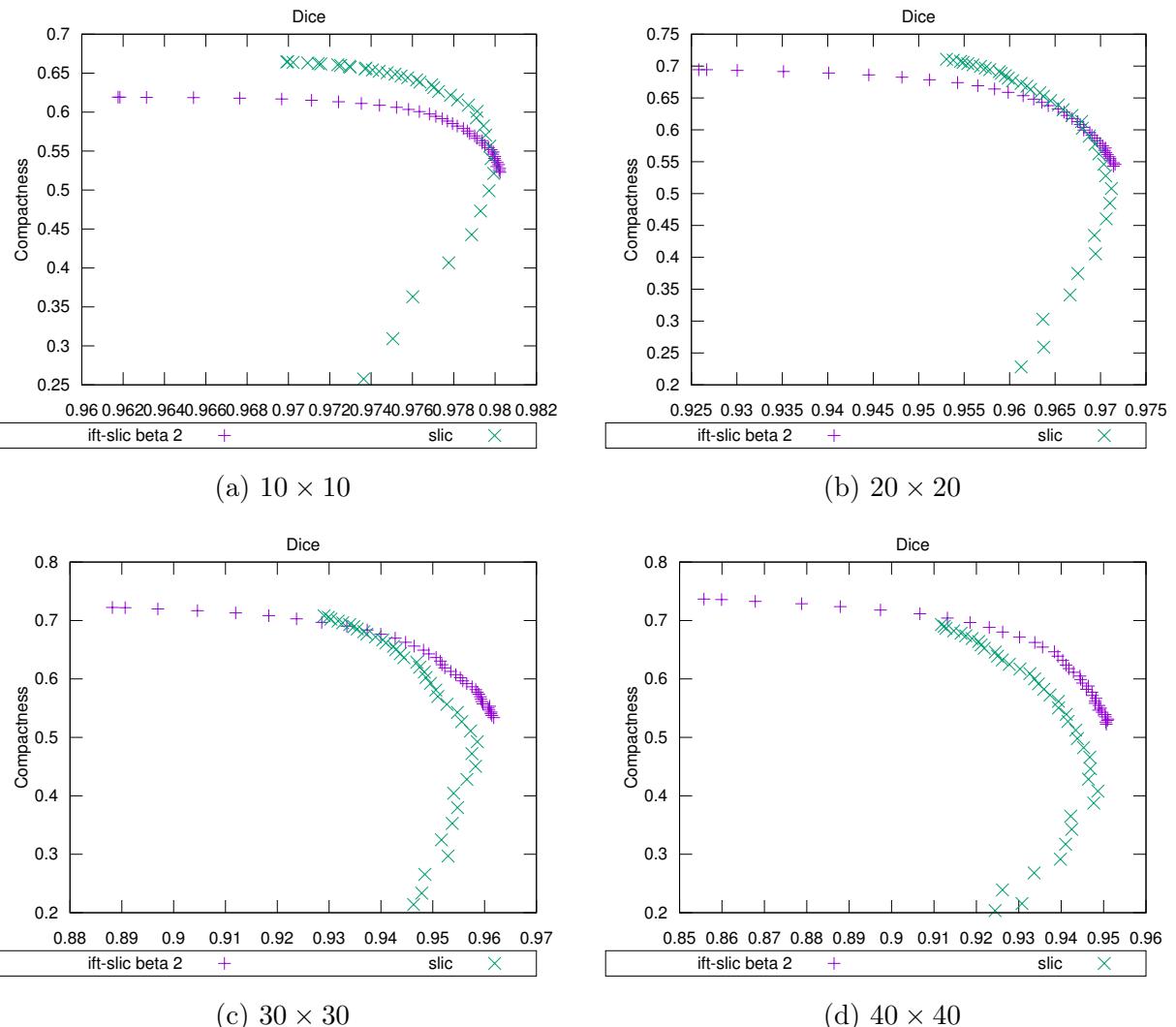


Gráfico 5.30: Curvas obtidas pela comparação entre acurácia e compacidade da variação dos parâmetros do SLIC e IFT-SLIC na base de imagens Grabcut.

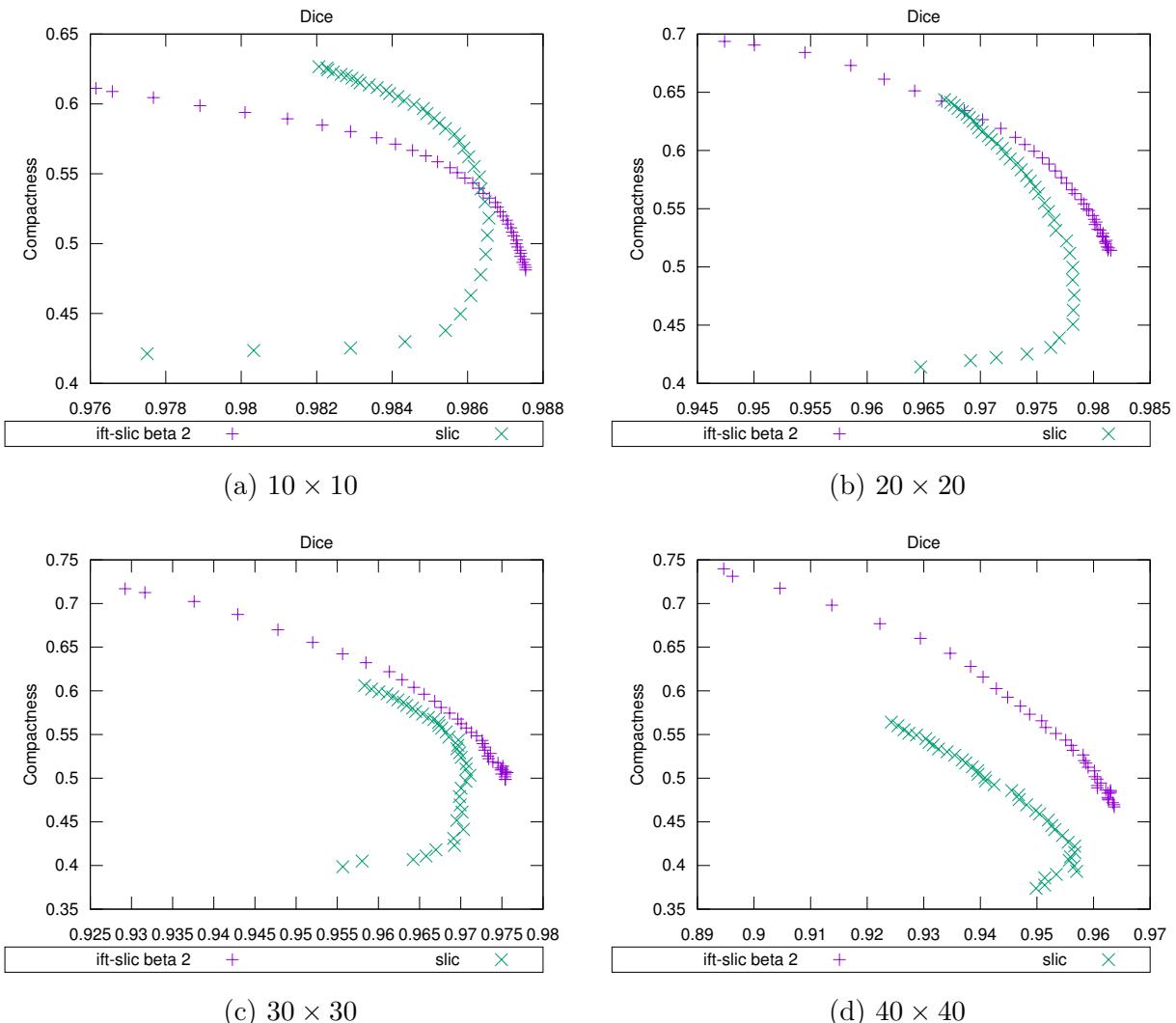


Gráfico 5.31: Curvas obtidas pela comparação entre acurácia e compacidade da variação dos parâmetros do SLIC e IFT-SLIC na base de imagens Liver.

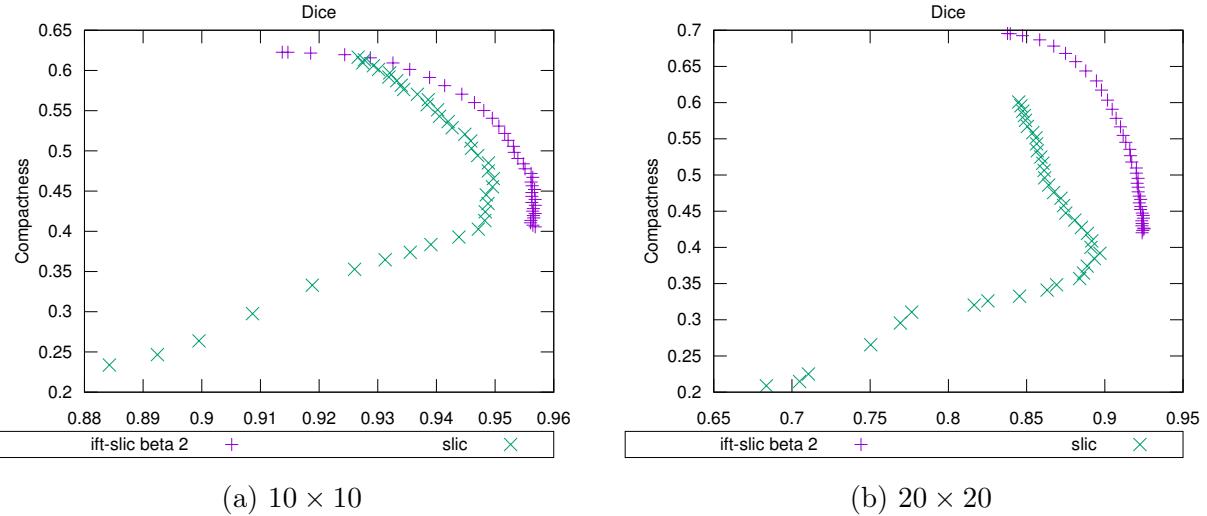


Gráfico 5.32: Curvas obtidas pela comparação entre acurácia e compacidade da variação dos parâmetros do SLIC e IFT-SLIC na base de imagens Talus.

5.5 Eficiência

De acordo com a Propriedade P.3, superpixels precisam ser gerados da forma mais rápida possível devido à sua natureza como uma etapa preparatória para um determinado *pipeline*. Nesta seção, foi comparada a velocidade de geração de superpixels entre o IFT-SLIC e SLIC.

Todos os resultados dessa seção foram executados em um computador com uma CPU Haswell Intel Core i5-4300U de 1.9 GHz com 2 cores e 4 threads, 8 GB de memória RAM DDR3-1600, GPU integrada Intel HD Graphics 4400 e 256 GB de armazenamento SSD. O sistema operacional utilizado foi o Gentoo GNU/Linux com *kernel* 4.11.5 rodando com *governor* “*performance*”.

Os resultados do IFT-SLIC foram separados em duas versões, uma usando os parâmetros originais da Tabela 5.1 (IFT-SLIC A) e outra com parâmetros focados em maior eficiência em troca de acurácia (IFT-SLIC B).

Para o IFT-SLIC B, foram utilizados valores de ϵ_c em 15 e ϵ_s em 6 com o objetivo de diminuir o número de sementes computadas por iteração como demonstrado nos gráficos 4.1, 4.2 e 4.3 e também foi limitado o número de iterações no IFT-SLIC para 4.

Os gráficos 5.33, 5.34 e 5.35 justificam esta estratégia demonstrando uma estabilização da pontuação do *Dice* nesta iteração na maioria dos casos, otimizando seu tempo de execução com uma perda mínima de qualidade.

Diminuição da imagem de entrada e o uso do SLIC nas primeiras iterações para seleção mais robusta das sementes são outras estratégias possíveis para otimização do tempo de execução do IFT-SLIC. Essas estratégias apresentam uma ótima alternativa para ganhos de eficiência mantendo a qualidade dos superpixels gerados superior a do SLIC. Ambas foram utilizadas e avaliadas no artigo submetido à *IEEE Transactions on Image Processing* (tip, 2017) intitulado *An Iterative Spanning Forest Framework for Superpixel Segmentation*³.

Além das estratégias comentadas acima, ainda existe a possibilidade de otimização do algoritmo ao modificar o funcionamento da fila de prioridades utilizada, já que 80% de seu tempo de execução ocorre ao acessá-la. Implementações de complexidade linear são possíveis e abordadas em outros estudos (Falcao *et al.*, 1999).

Os gráficos 5.36, 5.37 e 5.38 compararam o tempo de execução médio e sua acurácia média dos 3 métodos nas bases de imagens *Grabcut*, *Liver* e *Talus* respectivamente. O tempo de execução de ambos os métodos foi obtido registrando o tempo corrido para execução do laço de atribuição e

³<http://www.ic.unicamp.br/~afalcao/isf-superpixels/>

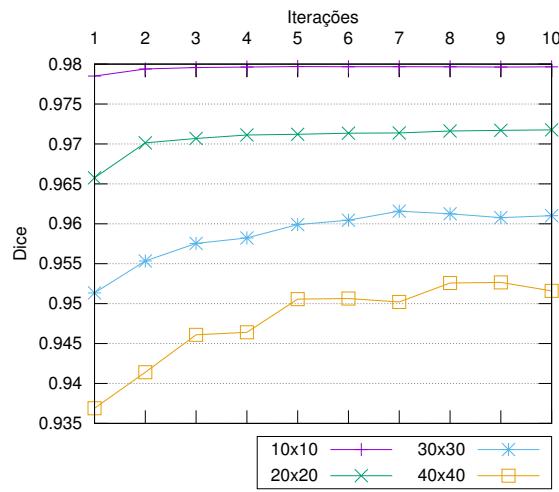


Gráfico 5.33: Exemplo de estabilização da pontuação (Dice) dos superpixels no IFT-SLIC em cada iteração para a base de imagens Grabcut.

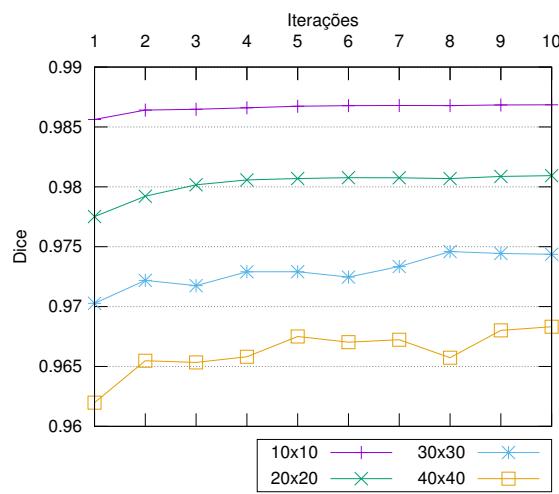


Gráfico 5.34: Exemplo de estabilização da pontuação (Dice) dos superpixels no IFT-SLIC em cada iteração para a base de imagens Liver.

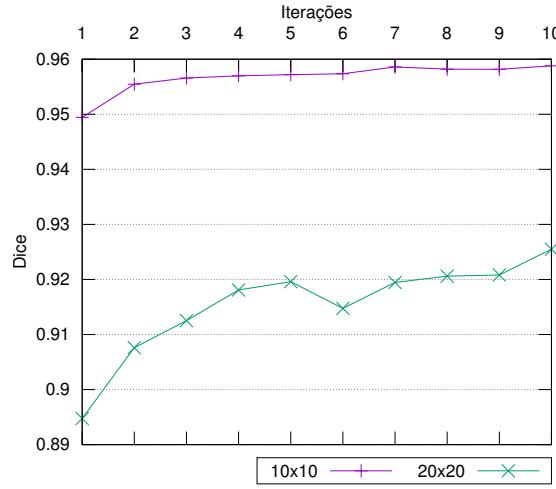


Gráfico 5.35: Exemplo de estabilização da pontuação (Dice) dos superpixels no IFT-SLIC em cada iteração para a base de imagens Tálu.

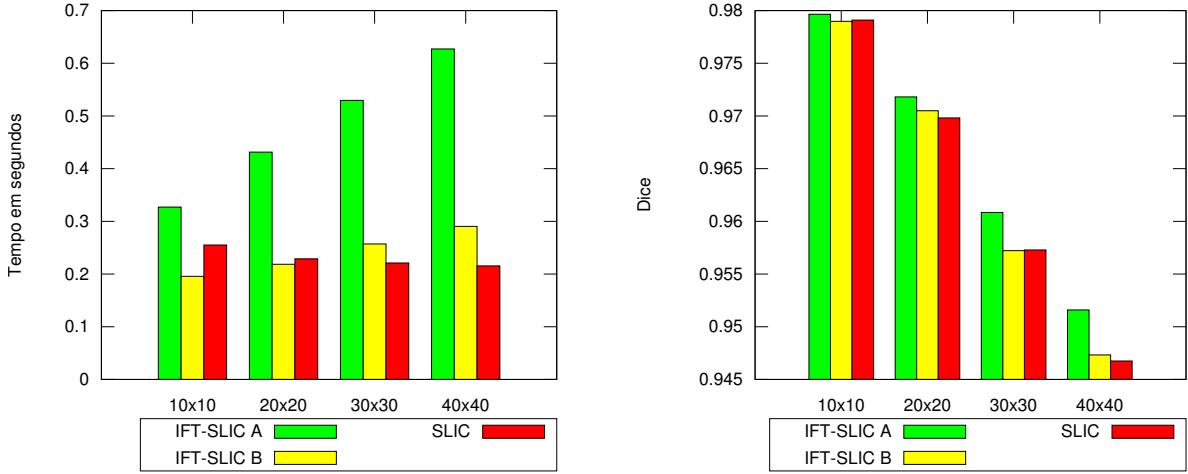


Figura 5.36: Tempo de execução e acurácia correspondente para a base de imagens Grabcut.

atualização, e no caso do SLIC também a etapa de pós-processamento. Essa comparação apresenta uma similaridade grande entre o IFT-SLIC B e o SLIC no gráfico 5.36, e, nos outros dois gráficos 5.37 e 5.38, o IFT-SLIC B mantém uma superioridade em relação ao SLIC em acurácia em todos os casos sem comprometer no tempo de execução. No caso do IFT-SLIC A, o mesmo obtém uma acurácia superior ao IFT-SLIC B e SLIC em troca de menor eficiência em todos os casos apresentados.

5.6 Aplicação de Alto Nível

Considerando que superpixels são normalmente usados como uma etapa de pré-processamento para aplicações de alto nível, é fundamental uma análise mais detalhada no impacto que um superpixel irá trazer ao resultado final da aplicação. Nesta seção, foi usado um algoritmo para detecção de céus em imagens desenvolvidas por Kostolansky (2016). Esse algoritmo pode ser dividido nas seguinte etapas:

1. Superpixels são gerados usando o SLIC;
2. Cada superpixel é trocado pela sua cor média (usando CIE-Lab);

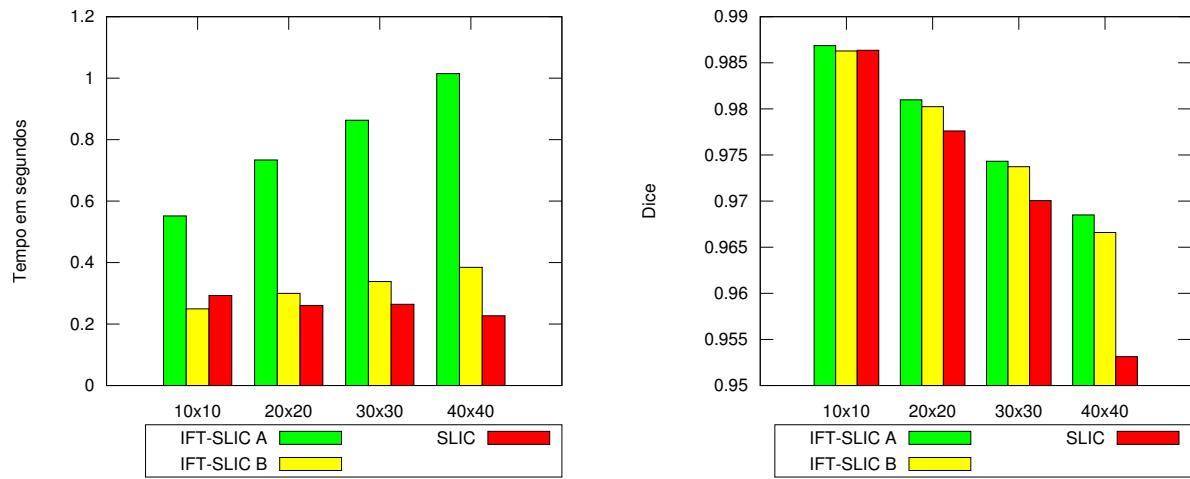


Figura 5.37: Tempo de execução e acurácia correspondente para a base de imagens Liver.

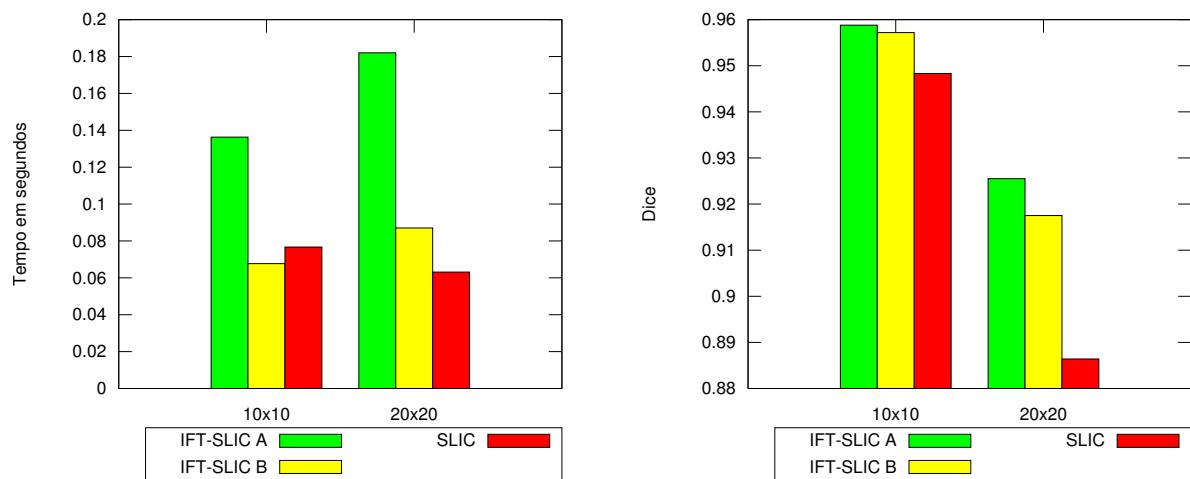


Figura 5.38: Tempo de execução e acurácia correspondente para a base de imagens Tálus.

3. Uma limiar T é calculada:

$$T = [d1 + (d2 - d1) * 0.3] * 1.15,^4 \quad (5.1)$$

em que, $d1$ é a distância média entre os superpixels nos primeiros 10% do topo da imagem e $d2$ é a distância média entre os superpixels da imagem sem $\frac{1}{3}$ das menores distâncias;

4. Superpixels com distância menor que T são mesclados;

5. O superpixel localizado na primeira linha, com a maior área, é considerado como céu.

A Figura 5.39 demonstra o passo a passo das etapas acima.

A base de imagens originalmente usada pelo autor não é distribuído para uso público, além disso, nenhuma base de imagens com gabarito no foco na segmentação do céu foi encontrada. Foi resolvido então pela criação de uma nova base de imagens chamada *Sky*⁵. Como ponto inicial, foi utilizada a base de imagens (Fergus, 2003) ao qual possui 1074 imagens de aviões. Dessas imagens, 60 propícias foram selecionadas a este escopo, ou seja, imagens que apresentam o céu em situações desafiadoras para segmentação, e um gabarito foi gerado para cada uma delas. A Figura 5.40 mostra alguns exemplos da base de imagens.



Figura 5.40: Imagens da base de imagens *Sky*, com seus respectivos gábaritos.

Foi utilizada a mesma gama de parâmetros da Tabela 5.1 para ambos os métodos, porém como esse algoritmo tem uma tendência a falhar na etapa 4 com superpixels pequenos, fazendo a mescla de superpixels errados, os tamanhos de superpixels foram aumentados para 50×50 , 100×100 , 150×150 e 200×200 .

No Gráfico 5.41 fica claro que as mesmas vantagens já demonstradas nos testes com a base de imagens *Grabcut*, *Liver* e *Talus* ocorrem na troca do uso do SLIC pelo IFT-SLIC para uma aplicação de mais alto nível obtendo uma melhor segmentação final. Exemplos dessa são demonstradas na Figura 5.42.

5.7 Conclusão

Neste trabalho, foi apresentado um novo arcabouço para geração de superpixels baseado na IFT e SLIC chamado ISF. Resultados demonstram que o arcabouço ISF com o uso da função de cone-

⁴Os valores 0.3 e 1.15 foram escolhidos como os melhores para a base de imagens do autor.

⁵<https://www.ime.usp.br/~eduardob/datasets/sky/>

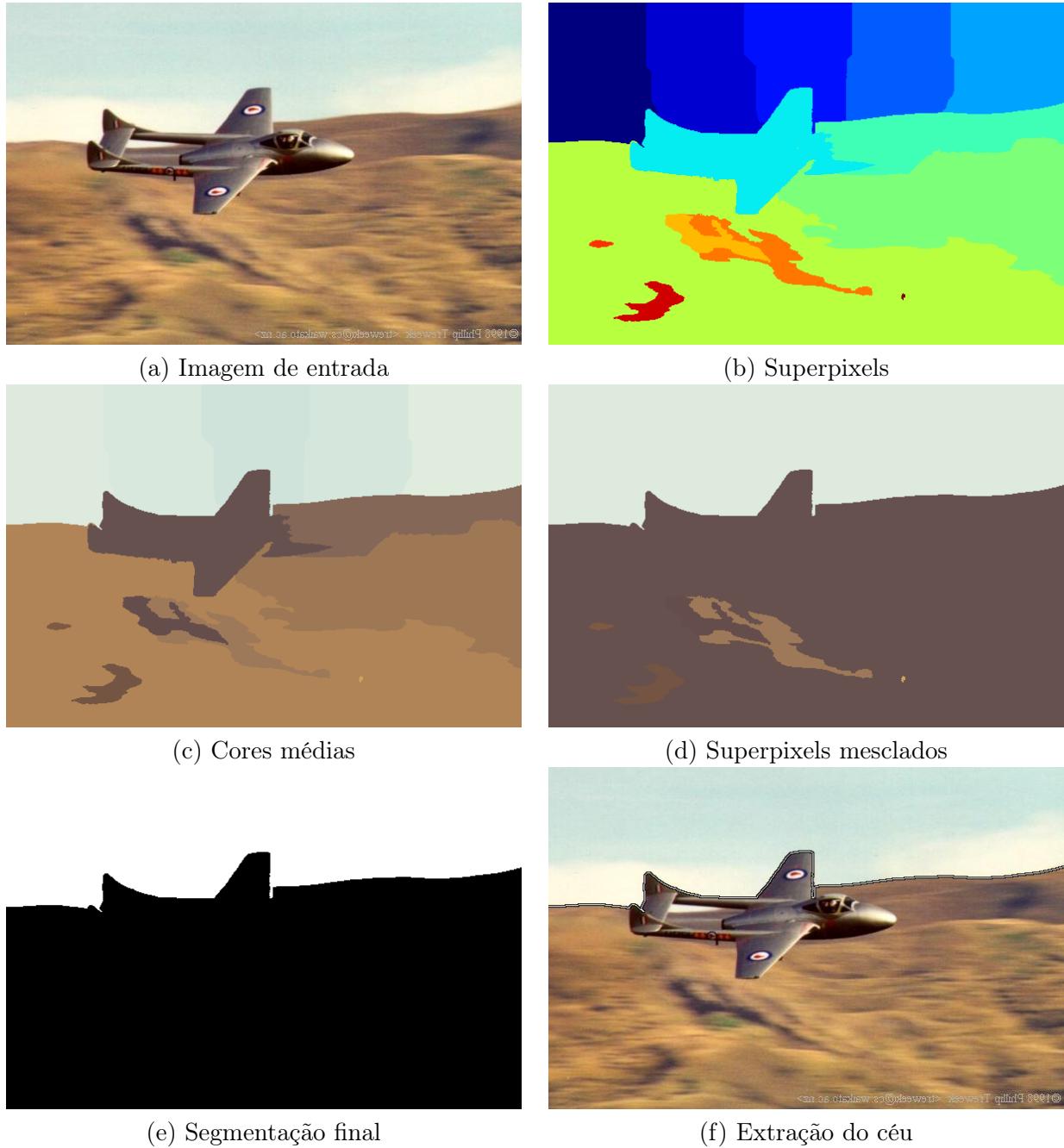


Figura 5.39: Demonstração das etapas do algoritmo de segmentação do céu.

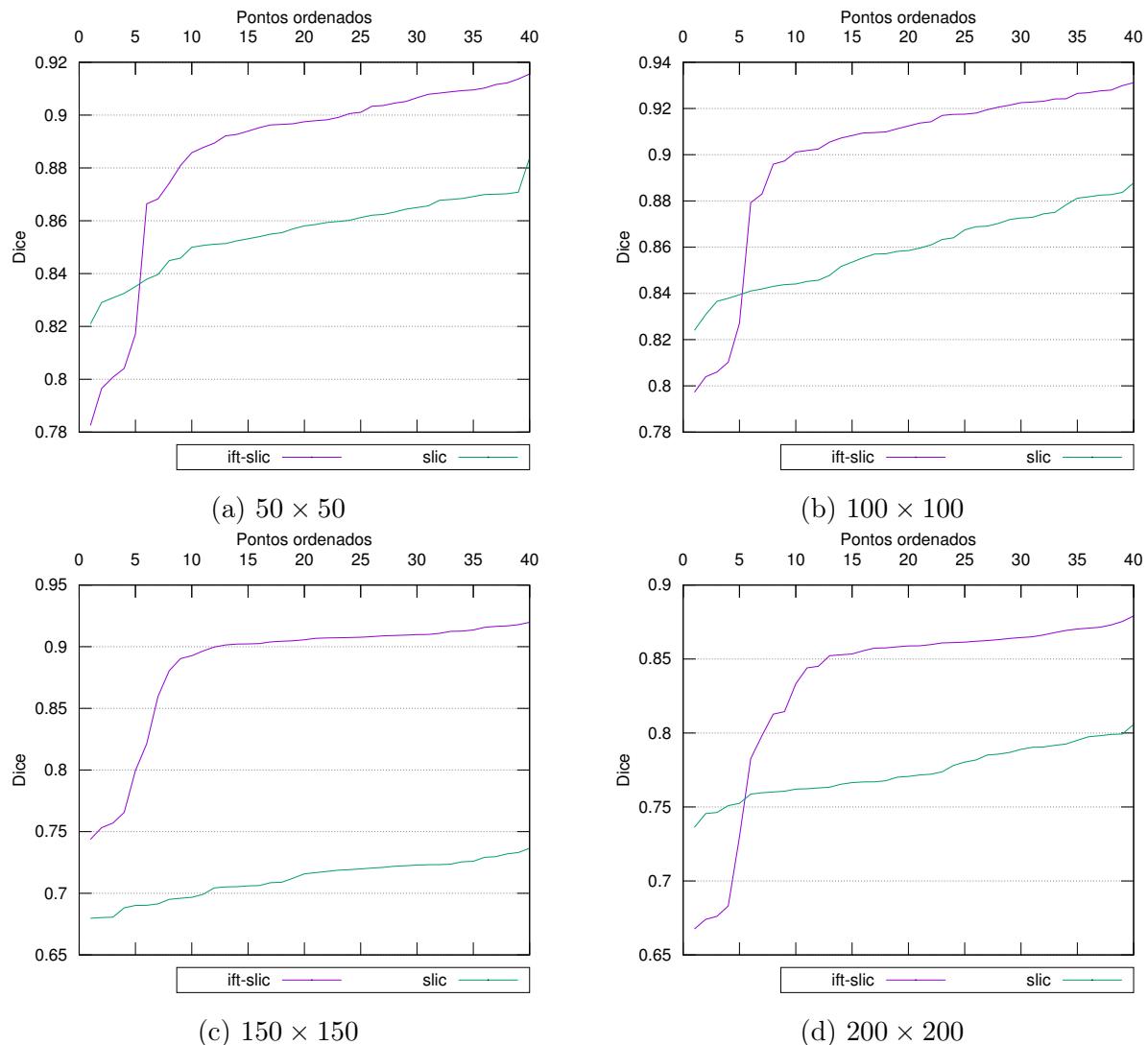
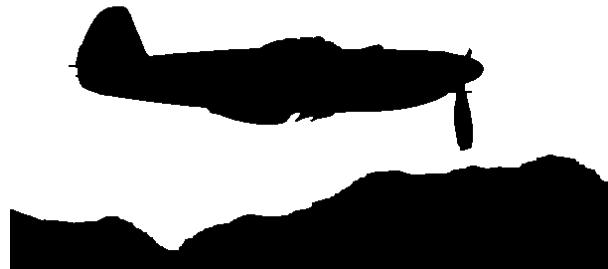


Gráfico 5.41: Curvas obtidas pela ordenação em ordem crescente de acurácia na base de imagens Sky.



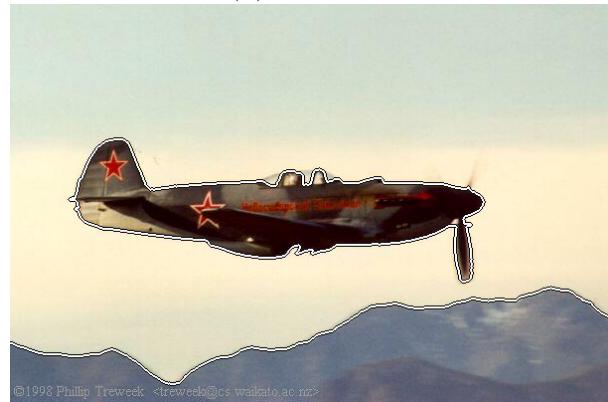
(a) Imagem original



(b) Gabarito



(c) SLIC



(d) IFT-SLIC



(a) Imagem original



(b) Gabarito



(c) SLIC



(d) IFT-SLIC

Figura 5.42: Exemplos de segmentação da base de imagens Sky. Na primeira linha, a pontuação Dice do SLIC (c) é 0.991623 e do IFT-SLIC (d) é 0.995687. Na segunda, a pontuação Dice do SLIC (c) é 0.884502 e do IFT-SLIC (d) é 0.99737.

xidade não monotonicamente incremental f_D , chamado IFT-SLIC, obtém melhorias na qualidade dos superpixels gerados tanto em acurácia quanto em compacidade, também fica clara a maior flexibilidade do IFT-SLIC quanto à escolha dos valores de seus parâmetros apresentando uma grande gama de valores aos quais possuem pontuações próximas ao valor ótimo, além do fato do ISF ser um arcabouço e não apenas um método como o SLIC, possibilitando a fácil mudança da função de conexidade para obtenção de melhores resultados de acordo com as particularidades de uma certa aplicação.

Testes extensivos foram efetuados no intuito de demonstrar o comportamento do IFT-SLIC nas mais variadas bases de imagens com diferentes objetivos de segmentação. Também foram demonstrados suas vantagens quanto ao seu uso como apenas um pré-processamento em uma aplicação de alto nível.

Este trabalho foi desenvolvido em parceria com o Professor Alexandre Xavier Falcão, professor titular do Instituto de Computação (IC) da Universidade de Campinas (UNICAMP), Campinas, Brasil⁶ e o Professor Ananda Shankar Chowdhury, professor associado do Departamento de Engenharia em Eletrônica e Telecomunicações (ETCE) da Universidade Jadavpur, Kolkata, India⁷.

Dessa parceria, um artigo foi submetido, aprovado e apresentado para a *Twenty-Eighth Brazilian Conference on Graphics, Patterns and Images* (2015), Salvador, Brasil, intitulado *A general framework for superpixel generation based on simple linear iterative clustering and image foresting transform* (Barreto-Alexandre *et al.*, 2015) o qual trata dos trabalhos iniciais de desenvolvimento do IFT-SLIC abordados neste trabalho.

Outro artigo foi submetido e atualmente está em fase de revisão para a revista *IEEE Transactions on Image Processing* (2017) intitulado *An Iterative Spanning Forest Framework for Superpixel Segmentation*. Este artigo inicia a abordagem a pesquisas futuras expandindo o escopo deste trabalho com novas técnicas para seleção inicial de sementes, o uso de novas funções de conexidade não monotonicamente incrementais e comparação com outros métodos além do SLIC.

5.8 Trabalhos Futuros

A extensão deste estudo para o uso de novas funções de conexidade em conjunto com o arcabouço ISF será o próximo passo deste trabalho. Este estudo já foi iniciado durante a elaboração do artigo *An Iterative Spanning Forest Framework for Superpixel Segmentation* no qual são apresentados variações da função f_D .

Além disso, o arcabouço ISF também pode ser adaptado para o uso em outros domínios de aplicação, como em cenários 3D, imagem médica volumétrica, segmentação espaço-temporal em vídeos, entre outros, de modo a popularizar as técnicas empregadas nessa dissertação.

Por fim, também se faz necessário o desenvolvimento e distribuição livre de uma biblioteca com a implementação tanto do IFT-SLIC quanto do ISF para o fácil acesso a terceiros e a pesquisadores.

⁶<http://www.ic.unicamp.br/~afalcao/>

⁷<https://sites.google.com/site/anandachowdhury>

Referências Bibliográficas

- sib (2015)** *Twenty-Eighth Brazilian Conference on Graphics, Patterns and Images*. IEEE Computer Society. ISBN 978-1-4673-7962-5. URL <http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7312722>. Citado na pág. 76, 77
- tip (2017)** *IEEE Transactions on Image Processing*. IEEE Transactions on Image Processing. Citado na pág. 68, 76
- Achanta et al. (2010)** Radhakrishna Achanta, Kevin Smith, Aurelien Lucchi, Pascal Fua e Sabine Süsstrunk. Slic superpixels*. Relatório Técnico Technical Report 149300, EPFL École polytechnique fédérale de Lausanne. Citado na pág. 2, 35, 43
- Achanta et al. (2012)** Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua e Sabine Susstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(11):2274–2282. ISSN 0162-8828. doi: 10.1109/TPAMI.2012.120. URL <http://dx.doi.org/10.1109/TPAMI.2012.120>. Citado na pág. 1, 2, 7, 43
- Audigier e Lotufo (2007)** R. Audigier e R.A. Lotufo. Watershed by image foresting transform, tie-zone, and theoretical relationship with other watershed definitions. Em *Proceedings of the 8th Intl. Symposium on Mathematical Morphology*, páginas 277–288, Rio de Janeiro, Brazil. Citado na pág. 6
- Axel et al. (1987)** L. Axel, J. Costantini e J. Listerud. Intensity correction in surface-coil mr imaging. *American Journal of Roentgenology*, 148:418–420. Citado na pág. 7
- Ayvaci e Soatto (2009)** A. Ayvaci e S. Soatto. Motion segmentation with occlusions on the superpixel graph. Em *Proc. of the Workshop on Dynamical Vision, Kyoto, Japan*, páginas 727–734. Citado na pág. 1
- Ballester et al. (2003)** C. Ballester, V. Caselles e P. Monasse. The Tree of Shapes of an Image. *ESAIM-COCV*, páginas 1–18. URL <http://citeseer.ist.psu.edu/ballester01tree.html>. Citado na pág. 6
- Banerjee e Maji (2013)** A. Banerjee e P. Maji. Contraharmonic mean based bias field correction in mr images. Em *Computer Analysis of Images and Patterns (CAIP)*, volume 8047, páginas 523–530. Citado na pág. 8
- Barreto-Alexandre et al. (2015)** Eduardo Barreto-Alexandre, Ananda Shankar Chowdhury, Alexandre Xavier Falcão e Paulo André Vechiatto Miranda. IFT-SLIC: A general framework for superpixel generation based on simple linear iterative clustering and image foresting transform. In *28th SIBGRAPI Conference on Graphics, Patterns and Images, SIBGRAPI 2015, Salvador, Bahia, Brazil, August 26-29, 2015* sib (2015), páginas 337–344. ISBN 978-1-4673-7962-5. doi: 10.1109/SIBGRAPI.2015.20. URL <http://dx.doi.org/10.1109/SIBGRAPI.2015.20>. Citado na pág. 2, 76
- Bergo e Falcão (2007)** Felipe P. G. Bergo e Alexandre X. Falcão. A partitioned algorithm for the image foresting transform. *International Symposium on Mathematical Morphology*. Citado na pág. 20

- Beucher e Meyer (1993)** S. Beucher e F. Meyer. The morphological approach to segmentation: The watershed transformation. Em *Mathematical Morphology in Image Processing*, chapter 12, páginas 433–481. Marcel Dekker. Citado na pág. 7
- Boyes et al. (2008)** R.G. Boyes, J.L. Gunter, C. Frost, A.L. Janke, T. Yeatman, D.L.G. Hill, M.A. Bernstein, P.M. Thompson, M.W. Weiner, N. Schuff *et al.* Intensity non-uniformity correction using N3 on 3-T scanners with multichannel phased array coils. *Neuroimage*, 39(4):1752–1762. Citado na pág. 8
- Boykov e Funka-Lea (2006)** Y. Boykov e G. Funka-Lea. Graph cuts and efficient N-D image segmentation. *International Journal of Computer Vision*, 70(2):109–131. ISSN 0920-5691. doi: <http://dx.doi.org/10.1007/s11263-006-7934-5>. Citado na pág. 6
- Boykov e Jolly (2001)** Y.Y. Boykov e M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. Em *International Conference on Computer Vision (ICCV)*, volume 1, páginas 105–112. Citado na pág. 6
- Brinkmann et al. (1998)** B. H. Brinkmann, A. Manduca e Richard A. Robb. Optimized homomorphic unsharp masking for MR grayscale inhomogeneity correction. *Medical Imaging, IEEE Transactions on*, 17(2):161–171. ISSN 0278-0062. doi: 10.1109/42.700729. URL <http://dx.doi.org/10.1109/42.700729>. Citado na pág. 7
- Canny (1986)** J Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698. ISSN 0162-8828. doi: 10.1109/TPAMI.1986.4767851. URL <http://dx.doi.org/10.1109/TPAMI.1986.4767851>. Citado na pág. 8
- Cappabianco et al. (2007)** Fabio AM Cappabianco, Guido Araujo e Alexandre X Falcao. The image forest transform architecture. Em *Field-Programmable Technology, 2007. ICFPT 2007. International Conference on*, páginas 137–144. IEEE. Citado na pág. 20
- Cappabianco et al. (2012)** F.A.M. Cappabianco, P.A.V. Miranda, J.S. Ide, C.L. Yasuda e A.X. Falcão. Unraveling the compromise between skull stripping and inhomogeneity correction in 3T MR images. Em *Sibgrapi (Conference on Graphics, Patterns and Images)*, páginas 1–8, Ouro Preto, MG, Brazil. Citado na pág. 8
- Carballido-Gamio et al. (2004)** J. Carballido-Gamio, S.J. Belongie e S. Majumdar. Normalized cuts in 3D for spinal MRI segmentation. *IEEE Transactions on Medical Imaging*, 23(1):36–44. Citado na pág. 5
- Cheng (1995)** Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799. Citado na pág. 6
- Comaniciu e Meer (2002)** D. Comaniciu e P. Meer. Mean shift: a robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5): 603–619. ISSN 0162-8828. doi: 10.1109/34.1000236. Citado na pág. 6
- Cormen et al. (2001)** T. H. Cormen, C. E. Leiserson, R. L. Rivest e C. Stein. Dijkstra's algorithm. Em *Introduction to Algorithms 2nd edition*, chapter 24, páginas 595–599. MIT Press. Citado na pág. 20
- Cuadros et al. (2012)** O. Cuadros, G. Botelho, F. Rodrigues e J.B. Neto. Segmentation of large images with complex networks. Em *Graphics, Patterns and Images (SIBGRAPI), 2012 25th SIBGRAPI Conference on*, páginas 24–31. doi: 10.1109/SIBGRAPI.2012.13. Citado na pág. 1, 6
- Dice (1945)** L. R. Dice. Measures of the Amount of Ecologic Association Between Species. *Ecology*, 26(3):297–302. Citado na pág. 15

- Dorini *et al.* (2007)** L.B. Dorini, R. Minetto e N.J. Leite. White blood cell segmentation using morphological operators and scale-space analysis. Em *XX Brazilian Symposium on Computer Graphics and Image Processing, SIBGRAPH*, páginas 294–304. doi: 10.1109/SIBGRAPI.2007.33. Citado na pág. 28
- Duda e Hart (1973)** R.O. Duda e P.E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, New York. Citado na pág. 5
- Falcao *et al.* (1999)** Alexandre X. Falcao, Jayaram K. Udupa e Flavio K. Miyazawa. Ultrafast user-steered image segmentation paradigm: live-wire-on-the-fly. *Proc. SPIE*, 3661:184–191. doi: 10.1117/12.348573. URL <http://dx.doi.org/10.1117/12.348573>. Citado na pág. 68
- Falcão e Bergo (2004)** A. X. Falcão e F. P. G. Bergo. Interactive volume segmentation with differential image foresting transforms. *IEEE Trans. on Medical Imaging*, 23(9):1100–1108. Citado na pág. 20, 29
- Falcão *et al.* (2001)** A.X. Falcão, B.S. da Cunha e R.A. Lotufo. Design of connected operators using the image foresting transform. Em *Proceedings of SPIE on Medical Imaging*, volume 4322, páginas 468–479. Citado na pág. 6
- Falcão *et al.* (2002)** A.X. Falcão, L.F. Costa e B.S. da Cunha. Multiscale skeletons by image foresting transform and its applications to neuromorphometry. *Pattern Recognition*, 35(7):1571–1582. Citado na pág. 3
- Falcão *et al.* (2004)** A.X. Falcão, J. Stolfi e R.A. Lotufo. The image foresting transform: Theory, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1):19–29. Citado na pág. 2, 3, 21, 37
- Felzenszwalb e Huttenlocher (2004)** Pedro F. Felzenszwalb e Daniel P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181. Citado na pág. 5, 7
- Fergus (2003)** R. Fergus. Caltech airplanes (side) dataset. http://www.vision.caltech.edu/Image_Datasets/airplanes_side/airplanes_side.tar, 2003. Accessed: 2016-11-12. Citado na pág. 72
- Fourier (1822)** Jean Baptiste Joseph Fourier. *Théorie Analytique de la Chaleur*. Cambridge University Press. ISBN 9780511693229. URL <http://dx.doi.org/10.1017/CBO9780511693229>. Cambridge Books Online. Citado na pág. 11
- Fulkerson *et al.* (2009a)** Brian Fulkerson, A. Vedaldi e S. Soatto. Class segmentation and object localization with superpixel neighborhoods. Em *Proc. IEEE International Conf. on Computer Vision (ICCV)*, páginas 670–677. Citado na pág. 1
- Fulkerson *et al.* (2009b)** Brian Fulkerson, A. Vedaldi e S. Soatto. Class segmentation and object localization with superpixel neighborhoods. Em *Computer Vision, 2009 IEEE 12th International Conference on*, páginas 670–677. doi: 10.1109/ICCV.2009.5459175. Citado na pág. 7
- Grady (2006)** L. Grady. Random walks for image segmentation. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 28(11):1768–1783. ISSN 0162-8828. doi: <http://dx.doi.org/10.1109/TPAMI.2006.233>. Citado na pág. 6
- Géraud *et al.* (2013)** Thierry Géraud, Edwin Carlinet, Sébastien Crozet e Laurent Najman. A quasi-linear algorithm to compute the tree of shapes of nD images. *Mathematical Morphology and Its Applications to Signal and Image Processing*, 7883:98–110. Citado na pág. 6
- Helldén (1980)** U. Helldén. *A Test of Landsat-2 Imagery and Digital Data for Thematic Mapping: Illustrated by an Environmental Study in Northern Kenya*. Rapporter och notiser. Laboratory of Remote Sensing, Department of Physical Geography, University of Lund. URL <https://books.google.com.br/books?id=X9PPHAAACAAJ>. Citado na pág. 15

- Herman e Carvalho (2001)** G.T. Herman e B.M. Carvalho. Multiseeded segmentation using fuzzy connectedness. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:460–474. Citado na pág. 3
- Kiryati e Székely (1993)** Nahum Kiryati e Gabor Székely. Estimating shortest paths and minimal distances on digitized three-dimensional surfaces. *Pattern Recognition*, 26(11):1623 – 1637. Citado na pág. 16
- Kostolansky (2016)** Juraj Kostolansky. Sky segmentation using Slic superpixels, 2016. URL <http://vgg.fit.stuba.sk/2015-02/sky-detection-using-slic-superpixels/>. Citado na pág. 1, 9, 70
- Labatut e Cherifi (2012)** Vincent Labatut e Hocine Cherifi. Accuracy measures for the comparison of classifiers. *CoRR*, abs/1207.3790. Citado na pág. 15
- Levinshtein et al. (2009)** A. Levinshtein, A. Stere, K.N. Kutulakos, D.J. Fleet, S.J. Dickinson e K. Siddiqi. Turbopixels: Fast superpixels using geometric flows. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(12):2290–2297. ISSN 0162-8828. doi: 10.1109/TPAMI.2009.96. Citado na pág. 7
- Liu et al. (2014)** Zhi Liu, Xiang Zhang, Shuhua Luo e Olivier Le Meur. Superpixel-based spatio-temporal saliency detection. *IEEE Trans. Circuits Syst. Video Techn.*, 24(9):1522–1540. Citado na pág. 1
- Lotufo e Falcão (2000)** R.A. Lotufo e A.X. Falcão. The ordered queue and the optimality of the watershed approaches. Em *Proceedings of the Intl. Symposium on Mathematical Morphology and its Applications to Signal and Image Processing (ISMM)*, volume 18, páginas 341–350. Kluwer. Citado na pág. 6
- Lotufo et al. (2002)** R.A. Lotufo, A.X. Falcão e F. Zampirolli. IFT-Watershed from gray-scale marker. Relatório Técnico IC-02-12, Institute of Computing, University of Campinas. Citado na pág. 6, 7
- Mansilla e Miranda (2013a)** L.A.C. Mansilla e P.A.V. Miranda. Image segmentation by oriented image foresting transform: Handling ties and colored images. Em *18th Intl. Conf. on Digital Signal Processing (DSP)*, páginas 1–6, Santorini, Greece. Citado na pág. 3, 5
- Mansilla e Miranda (2013b)** L.A.C. Mansilla e P.A.V. Miranda. Image segmentation by oriented image foresting transform with geodesic star convexity. Em *15th International Conference on Computer Analysis of Images and Patterns*, volume 8047, páginas 572–579, York, UK. Citado na pág. 3, 5
- Mansilla et al. (2013a)** L.A.C. Mansilla, F.A.M. Cappabianco e P.A.V. Miranda. Image segmentation by image foresting transform with non-smooth connectivity functions. Em *Conference on Graphics, Patterns, and Images (SIBGRAPI)*, páginas 147–154, Arequipa, Peru. Citado na pág. 3, 8, 35
- Mansilla et al. (2013b)** L.A.C. Mansilla, M.P. Jackowski e P.A.V. Miranda. Image foresting transform with geodesic star convexity for interactive image segmentation. Em *Proc. of the International Conference on Image Processing (ICIP)*, páginas 4054–4058, Melbourne, Australia. IEEE. Citado na pág. 5
- Mansilla e Miranda (2016)** Lucy AC Mansilla e Paulo AV Miranda. Oriented image foresting transform segmentation: Connectivity constraints with adjustable width. *29th SIBGRAPI Conference on Graphics, Patterns and Images, SIBGRAPI 2016, Salvador, Bahia, Brazil, October 4-7, 2016*. Citado na pág. 5

- Mansilla et al. (2016)** Lucy AC Mansilla, Paulo AV Miranda e Fábio AM Cappabianco. Oriented image foresting transform segmentation with connectivity constraints. Em *Image Processing (ICIP), 2016 IEEE International Conference on*, páginas 2554–2558. IEEE. Citado na pág. 5
- Miranda e Falcão (2009)** Paulo A.V. Miranda e Alexandre X. Falcão. Links between image segmentation based on optimum-path forest and minimum cut in graph. *Journal of Mathematical Imaging and Vision*, 35(2):128–142. ISSN 0924-9907. doi: 10.1007/s10851-009-0159-9. URL <http://dx.doi.org/10.1007/s10851-009-0159-9>. Citado na pág. 22
- Miranda e Mansilla (2014)** P.A.V. Miranda e L.A.C. Mansilla. Oriented image foresting transform segmentation by seed competition. *IEEE Transactions on Image Processing*, 23(1):389–398. Citado na pág. 3, 5, 6
- Miranda et al. (2008)** P.A.V. Miranda, A.X. Falcão, A. Rocha e F.P.G. Bergo. Object delineation by κ -connected components. *EURASIP Journal on Advances in Signal Processing*, páginas 1–14. Citado na pág. 3
- Miranda et al. (2011)** P.A.V. Miranda, A.X. Falcão e T.V. Spina. The riverbed approach for user-steered image segmentation. Em *Proc. of the International Conference on Image Processing*, páginas 3133–3136, Brussels, Belgium. IEEE. doi: 10.1109/ICIP.2011.6116330. Citado na pág. 3
- Miranda et al. (2012)** P.A.V. Miranda, A.X. Falcão e T.V. Spina. Riverbed: A novel user-steered image segmentation method based on optimum boundary tracking. *IEEE Transactions on Image Processing*. doi:10.1109/TIP.2012.2188034. Citado na pág. 3
- Moore et al. (2008)** A.P. Moore, S. Prince, J. Warrell, U. Mohammed e G. Jones. Superpixel lattices. Em *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, páginas 1–8. doi: 10.1109/CVPR.2008.4587471. Citado na pág. 7
- Moore et al. (2009)** A.P. Moore, S.J.D. Prince, J. Warrell, U. Mohammed e G. Jones. Scene shape priors for superpixel segmentation. Em *Computer Vision, 2009 IEEE 12th International Conference on*, páginas 771–778. doi: 10.1109/ICCV.2009.5459246. Citado na pág. 7
- Moore et al. (2010)** A.P. Moore, S.J.D. Prince e J. Warrell. Lattice cut - constructing superpixels using layer constraints. Em *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, páginas 2117–2124. doi: 10.1109/CVPR.2010.5539890. Citado na pág. 7
- Mori et al. (2004)** Greg Mori, Xiaofeng Ren, Alexei A. Efros e Jitendra Malik. Recovering human body configurations: combining segmentation and recognition. Em *Proc. IEEE Conf. Comput. Vision and Pattern Recogn.*, volume 2, páginas 326–333. Citado na pág. 8
- Najman e Couprie (2006)** L. Najman e M. Couprie. Building the component tree in quasi-linear time. *IEEE Transactions on Image Processing*, 15(11):3531–3539. Citado na pág. 6
- Panagiotakis et al. (2013)** Costas Panagiotakis, Harris Papadakis, Elias Grinias, Nikos Komodakis, Paraskevi Fragopoulou e Georgios Tziritas. Interactive image segmentation based on synthetic graph coordinates. *Pattern Recognition*, 46(11):2940 – 2952. ISSN 0031-3203. doi: <http://dx.doi.org/10.1016/j.patcog.2013.04.004>. URL <http://www.sciencedirect.com/science/article/pii/S003132031300174X>. Citado na pág. 1
- Rauber et al. (2013)** P.E. Rauber, A.X. Falcao, T.V. Spina e P.J. De Rezende. Interactive segmentation by image foresting transform on superpixel graphs. Em *Graphics, Patterns and Images (SIBGRAPI), 2013 26th SIBGRAPI - Conference on*, páginas 131–138. doi: 10.1109/SIBGRAPI.2013.27. Citado na pág. 1, 15
- Ren e Malik (2003)** Xiaofeng Ren e Jitendra Malik. Learning a classification model for segmentation. Em *Proc. 9th Int'l. Conf. Computer Vision*, volume 1, páginas 10–17. Citado na pág. 8

- Rijsbergen (1979)** C. J. Van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 2nd ed. ISBN 0408709294. Citado na pág. 15
- Rocha et al. (2009)** Leonardo Marques Rocha, Fábio A. M. Cappabianco e Alexandre Xavier Falcão. Data clustering as an optimum-path forest problem with applications in image analysis. *International Journal of Imaging Systems and Technology*, 19(2):50–68. Citado na pág. 3, 6, 7
- Salembier e Serra (1995)** P. Salembier e J. Serra. Flat zones filtering, connected operators, and filters by reconstruction. *IEEE Transactions on Image Processing*, 4(8):1153–1160. Citado na pág. 6
- Salembier et al. (1998)** P. Salembier, A. Oliveras e L. Garrido. Antiextensive connected operators for image and sequence processing. *IEEE Transactions on Image Processing*, 7(4):555–570. Citado na pág. 6
- Schick et al. (2012)** Alexander Schick, Mika Fischer e Rainer Stiefelhagen. Measuring and evaluating the compactness of superpixels. Em *Proc. IEEE International Conf. on Pattern Recognition (ICPR), Tsukuba, Japan, November*, páginas 930–934. Citado na pág. 16, 43
- Shi e Malik (2000)** J. Shi e J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905. Citado na pág. 5, 7, 8
- Shu et al. (2013)** Guang Shu, Afshin Dehghan e Mubarak Shah. Improving an object detector and extracting regions using superpixels. Em *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Portland, OR, USA*, páginas 3721–3727. Citado na pág. 1
- Silva e Lotufo (2011)** Alexandre Gonçalves Silva e Roberto de Alencar Lotufo. Efficient computation of new extinction values from extended component tree. *Pattern Recogn. Lett.*, 32(1):79–90. ISSN 0167-8655. doi: 10.1016/j.patrec.2010.07.019. URL <http://dx.doi.org/10.1016/j.patrec.2010.07.019>. Citado na pág. 7
- Sled et al. (1998)** J. G. Sled, A. P. Zijdenbos e A. C. Evans. A nonparametric method for automatic correction of intensity nonuniformity in MRI data. *IEEE transactions on medical imaging*, 17(1):87–97. ISSN 0278-0062. doi: 10.1109/42.668698. URL <http://dx.doi.org/10.1109/42.668698>. Citado na pág. 7
- Sørensen (1948)** T. Sørensen. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons. *Biol. Skr.*, 5:1–34. Citado na pág. 15
- Stehman (1997)** Stephen V. Stehman. Selecting and interpreting measures of thematic classification accuracy. *Remote Sensing of Environment*, 62(1):77 – 89. ISSN 0034-4257. doi: [http://dx.doi.org/10.1016/S0034-4257\(97\)00083-7](http://dx.doi.org/10.1016/S0034-4257(97)00083-7). URL <http://www.sciencedirect.com/science/article/pii/S0034425797000837>. Citado na pág. 15
- Strand et al. (2013)** R. Strand, K.C. Ciesielski, F. Malmberg e P.K. Saha. The minimum barrier distance. *Computer Vision and Image Understanding*, 117:429–437. Citado na pág. 3
- Torres et al. (2004)** R.S. Torres, A.X. Falcão e L.F. Costa. A graph-based approach for multiscale shape analysis. *Pattern Recognition*, 37(6):1163–1174. Citado na pág. 3
- Wang e Siskind (2003)** S. Wang e J.M. Siskind. Image segmentation with ratio cut. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(6):675–690. Citado na pág. 5
- Wang e Siskind (2001)** S. Wang e J.M. Siskind. Image segmentation with minimum mean cut. Em *International Conference on Computer Vision (ICCV)*, volume 1, páginas 517–525. Citado na pág. 5

- Wu *et al.* (2014)** W. Wu, A. Y. C. Chen, L. Zhao e J. J. Corso. Brain tumor detection and segmentation in a CRF (conditional random fields) framework with pixel-pairwise affinity and superpixel-level features. *International Journal of Computer Assisted Radiology and Surgery*, 9(2):241–253. Citado na pág. 1
- Xu *et al.* (2012)** Yongchao Xu, T. Geraud e L. Najman. Context-based energy estimator: Application to object segmentation on the tree of shapes. Em *Proc. of the International Conference on Image Processing*, páginas 1577–1580, Orlando, FL. IEEE. doi: 10.1109/ICIP.2012.6467175. Citado na pág. 6
- Yang *et al.* (2014)** Fan Yang, Huchuan Lu e Ming-Hsuan Yang. Robust superpixel tracking. *IEEE Trans. Image Processing*, 23(4):1639–1651. Citado na pág. 1
- Yang *et al.* (2012)** Yi Yang, Sam Hallman, Deva Ramanan e Charless C. Fowlkes. Layered object models for image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 34(9):1731–1743. Citado na pág. 1
- Zitnick e Kang (2007)** C. Lawrence Zitnick e Sing Bing Kang. Stereo for image-based rendering using image over-segmentation. *International Journal of Computer Vision*, 75(1):49–65. Citado na pág. 1
- Çigla e Alatan (2010)** Cevahir Çigla e A.A. Alatan. Efficient graph-based image segmentation via speeded-up turbo pixels. Em *Proc. of the International Conference on Image Processing*, páginas 3013–3016, Hong Kong. IEEE. doi: 10.1109/ICIP.2010.5653963. Citado na pág. 1, 5