

**UNIVERSIDADE NOVE DE JULHO - UNINOVE
PROGRAMA DE PÓS GRADUAÇÃO EM INFORMÁTICA E GESTÃO
DO CONHECIMENTO - PPGIGC**

CHARLES FERREIRA GOBBER

**ÚLTIMOS LEVELINGS COM BASE EM FUNÇÕES DE ENERGIA
APLICADOS A DETECÇÃO DE OBJETOS**

**São Paulo
2018**

CHARLES FERREIRA GOBBER

**ÚLTIMOS LEVELINGS COM BASE EM FUNÇÕES DE ENERGIA
APLICADOS A DETECÇÃO DE OBJETOS**

Esta versão da dissertação contém as correções e alterações sugeridas pela Comissão Julgadora durante a defesa da versão original do trabalho. Ademais, o trabalho confere ao autor o título de Mestre em Informática e Gestão do Conhecimento.

Comissão Julgadora:

- Prof. Dr. Wonder Alexandre Luz Alves (orientador) - PPGI-UNINOVE
- Prof. Dr. Ronaldo Fumio Hashimoto - IME-USP
- Prof. Dr. Sidnei Alves de Araújo - PPGI-UNINOVE

**São Paulo
2018**

AGRADECIMENTOS

Ao meu orientador e amigo de corridas Prof. **Wonder A. Luz Alves**, pela dedicação, paciência e contribuição neste trabalho.

A minha amada e compreensível **Luciene** pela paciência na falta das horas que não pude lhe dedicar, e pelo carinho e auxílio nesta dissertação.

Ao meu querido amigo **Saulo Daniel** pelos momentos e conversas sobre *machine learning* e visão computacional.

Aos amigos do **IME-USP** Prof. **Ronaldo F. Hashimoto, Denis, Alexandre** e meu orientador pelas calorosas reuniões de morfologia matemática.

Aos meus familiares (meus irmãos **Natalia** e **Marlon**, meus pais **Helena** e **Marco**) que compreenderam com amor e carinho os meus momentos de ausência.

A todos os outros amigos da **UNINOVE** que conquistei nesta minha jornada.

Aos membros desta banca examinadora, pelo tempo prestado na avaliação e correção do meu trabalho, que certamente será de valiosa contribuição.

Acima de tudo, obrigado Deus pela oportunidade desta encarnação.

A todos, muito obrigado!

RESUMO

Com o avanço da tecnologia muitas área de estudo tem ganhado notoriedade. Neste sentido os esforços envolvidos pela comunidade científica na área de visão computacional têm sido cada vez mais crescente, e muitas aplicações vêm sendo desenvolvidas nos mais diferentes âmbitos. Em geral, o interesse está em detectar e analisar objetos em imagens. Uma abordagem muito consolidada presente na literatura se baseia em analisar a forma de objetos dentro da imagem e então extrair características que os descrevam para que eles posteriormente sejam detectados. Nesse sentido, é explorada nesta dissertação uma teoria invariante a escala que é definida no âmbito da Morfologia Matemática. Mais especificamente, o nosso estudo é voltado para uma classe de operadores residuais chamados de últimos *levelings*. Esses operadores robustos analisam espaços de escalas baseado em *levelings* por meio de diferenças consecutivas (resíduos) e consideram os máximos resíduos. Em adição, esses resíduos revelam informações importantes sobre o contraste de uma imagem. Porém, devido a natureza dos operadores últimos *levelings* muitas vezes alguns resíduos extraídos de regiões da imagem acabam sendo indesejáveis. Uma abordagem para minimizar esse problema é construir estratégias para filtrar estes resíduos. Assim, apresenta-se nesta dissertação uma nova abordagem para a construção destas estratégias baseada nas chamadas funções de energia. Os resultados encontrados em duas aplicações: (i) reconhecimento de plantas via *bounding box detection* e (ii) segmentação de vasos sanguíneos em imagens de retinas revelam que a abordagem proposta é robusta e eficiente.

Palavras-chave: Últimos levelings, Funções de energia, Mumford-Shah, Árvores de componentes, Árvore de formas, Detecção de objetos.

ABSTRACT

With the advent of technology many research areas have gained notoriety. Following those ideas, the efforts and involvement from the scientific community in computer vision have been increased more and more, and a lot of applications have been developed in many different areas. Usually, the interest is in detect and analize objects in images. One interesting approach in literature is based on analizing the shape of an object inside the image, and extract some characteristics that can describe it to be detected later. At this sense, it is explored in this dissertation an interesting theory that is scale invariant and defined in Mathematical Morphology. More specifically, our research is focused in one class of residual operators called ultimate levelings. Those robust operators analyze a scale space based on levelings through consecutive differences and the maximal residues are considered. In addition, these operators reveal important information about the contrast of an image. However, because the nature of the ultimate levelings operators sometimes residual extracted by them it can be undesirable. One interesting approach to minimize this problem is construct strategys to filter the residuals. Thus, it is proposed in this dissertation a new approach to construct strategys to filter residuals from ultimate levelings based on energy functions. Finally, results obtained in two applications: (*i*) plant recognition by bounding box detection and (*ii*) image blood vessel retina segmentation show that our approach is robust and efficient.

Keywords: Ultimate levelings, Energy functions, Mumford-Shah, Component tree, Tree of shapes, Object detection.

SUMÁRIO

Lista de Figuras	8
Lista de Algoritmos	11
Lista de Abreviaturas	12
Lista de Símbolos	13
1 Introdução	16
1.1 Contextualização	16
1.2 Problema de pesquisa	17
1.3 Objetivos	20
1.4 Contribuições da pesquisa	20
1.5 Organização do trabalho	20
2 Conceitos preliminares	23
2.1 Definições sobre Imagens	23
3 Operadores conexos e leveling	27
3.1 Operadores conexos	27
3.1.1 Levelings	28
3.1.2 Espaço de escalas baseado em leveling	29
4 Representação de imagens a partir de árvores morfológicas	31
4.1 Introdução	31
4.2 Árvores e conjuntos parcialmente ordenados	31
4.3 Árvores a partir de conjuntos de níveis	34
4.3.1 Árvore de componentes	36
4.3.2 Árvore de formas	38
4.3.3 Reconstrução de árvores podadas	39
4.4 Algoritmos para construção de árvores morfológicas	41
5 Últimos leveling baseados em funções de energia	48
5.1 Espaço de escalas baseado em leveling obtido por podas sucessivas em árvores morfológicas	48
5.2 Extração residual em um espaço de escalas baseado em leveling	50
5.3 Estratégias para filtragem de resíduos indesejáveis	52
5.4 Estratégias baseadas em funções de energia	53
5.4.1 O funcional de Mumford-shah	53

5.4.2	Minimizando funções de energia a partir de árvores morfológicas	55
5.4.2.1	O funcional variacional	56
5.4.2.2	Ordem de remoção dos nós	60
5.4.2.3	O atributo funcional	62
5.4.3	Estratégias	64
5.4.4	Algoritmos para computação dos atributos de energia	65
6	Resultados e experimentos	68
6.1	Detecção de plantas	68
6.1.1	Bounding box detection	69
6.1.2	Abordagem proposta	69
6.1.3	Resultados	72
6.1.4	Conclusão	74
6.2	Análise de imagens de retina	75
6.2.1	Segmentação de vasos sanguíneos	75
6.2.2	Abordagem proposta	76
6.2.3	Resultados	80
6.2.4	Conclusão	82
7	Conclusões	84
7.1	Trabalhos futuros	84
Referências Bibliográficas		85
Índice Remissivo		93

LISTA DE FIGURAS

1.1 Exemplo de um espaço de escalas baseado em levelings construído a partir de uma imagem. As regiões em verde são as que foram removidas de uma escala para outra.	17
1.2 Computação dos últimos levelings a partir de subtrações consecutivas em um espaço de escalas. Note que, o resíduo 0 é a diferença entre o leveling 0 e o leveling 1 e assim sucessivamente.	18
1.3 Exemplo de espaço de busca do funcional de Mumford-Shah para algumas partições e imagens correspondentes. Neste exemplo fica mais fácil visualizar uma interpretação do funcional de Mumford-Shah e a sua relação como uma medida de avaliação de partições. Cada ponto é uma possível solução para o problema, o tamanho da marca representa o valor de energia do funcional de Mumford-Shah. Note que, esse espaço de busca pode ser enorme.	19
2.1 Exemplo de imagem multibanda f_{rgb} , as bandas são respectivamente: red, green e blue.	23
2.2 Exemplos de adjacências para um dado <i>pixel</i> p	24
2.3 Exemplos de conjuntos de componentes conexos para adjacências \mathcal{A}_4 e \mathcal{A}_8 . Note que, cada cor representa um CC em \mathcal{X} e os componentes laranja e verde na vizinhança-4 se tornaram apenas um componente em verde na vizinhança-8. . .	25
2.4 Exemplo de extração de zonas planas para uma imagem f sobre uma adjacência \mathcal{A}_4 . Note que, cada cor representa uma zona plana em $\mathcal{ZP}(f, \mathcal{A}_4)$	25
3.1 Exemplo de partições obtidas pela aplicação de operadores conexos ψ_1 e ψ_2 em f para a adjacência \mathcal{A}_4 . Note que, cada cor representa uma região.	27
3.2 Exemplos de operadores levelings aplicados em uma imagem f	29
4.1 Representação de imagens através de árvores.	31
4.2 Exemplos de diagramas de Hasse para dois <i>posets</i>	32
4.3 Definições gerais de árvores em um poset $\mathcal{T} = (\{\{1\}, \{3\}, \{4\}, \{5\}, \{1, 2\}, \{3, 4\}, \{5, 6\}, \{1, 2, 3, 4, 5, 6\}\}, \preceq)$	33
4.4 Exemplo de uma poda em uma dada árvore \mathcal{T}	34
4.5 Conjuntos de níveis $\mathcal{X}^{\lambda\dots}(f)$ e $\mathcal{X}_{\lambda\dots}(f)$ para uma dada imagem f	35
4.6 Exemplos de árvores de componentes para uma dada imagem f	36
4.7 Exemplos de árvores de componentes para uma dada imagem f , em vermelho são mostrados os CNPs e os menores CCs que os contêm.	37
4.8 Exemplo de árvore de formas para uma dada imagem f , em vermelho são mostrados os CPNs e os menores CCs que os contêm.	38

4.9 Um componente conexo C com buracos internos (destacados em verde) e seus conjuntos $sat(C)$ e $Ext(C)$	39
4.10 Exemplo de uma poda e uma reconstrução.	40
4.11 Construção de uma min-tree utilizando o Algoritmo 2.	42
4.12 Codificação de uma min-tree \mathcal{T} contida na estrutura $parent$	43
4.13 Visualização do Algoritmo 6 para construção da árvore de formas baseado em uma Gride Khalimsky simples. Note que, $F = \{F_{min}, F_{max}\}$ e os círculos vermelhos são a face 0, os retângulos verdes a face 1 e os quadrados em azul a face 2 e na estrutura parent os representantes estão em vermelho.	47
5.1 Espaço de escalas baseado em levelings obtido a partir de podas sucessivas em uma min-tree. Os nós representados em verdes são os marcados para serem podados.	49
5.2 Exemplos de aplicações de operadores últimos levelings. As imagens foram rotuladas para facilitar a visualização dos resíduos. Da esquerda para direita, UAO e UAC (MARCOTEGUI; HERNÁNDEZ; RETORNAZ, 2011) e UGF (ALVES; HASHIMOTO, 2014).	51
5.3 Comparação de um operador último leveling filtrado e não filtrado.	52
5.4 Exemplo do conjunto dos contornos ∂P	54
5.5 Exemplo de árvore simplificada \mathcal{T}_{i+1} computada a partir de \mathcal{T}_i . A árvore utilizada no exemplo é uma árvore de formas.	57
5.6 Exemplo de computação de árvores ótimas a partir de duas abordagens diferentes para explorar os nós, respectivamente Top-down e Bottom-up, com $\nu = 2$	61
5.7 Exemplo que não valida o princípio da causalidade, sendo $\nu_2 > \nu_1$	63
6.1 Evolução de plantas da espécie <i>Arabidopsis</i> analisadas durante algumas semanas. Note as diferenças encontradas na morfologia da planta durante seu ciclo de crescimento no período de aquisição das imagens.	68
6.2 Exemplo de uma imagem da base de imagens de plantas <i>Leaf Phenotyping dataset</i> com os bounding boxes detectados por nossa abordagem.	69
6.3 Exemplo ilustrando o contraste dos canais de uma imagem de planta f_{rgb} , note que o canal f_g apresenta maior contraste nas regiões da planta.	70
6.4 Representação da abordagem. A imagem f_g é a imagem do canal <i>green</i> e a imagem f_{ule} é a imagem de saída gerada pelo operador último leveling baseado nas estratégias combinadas $\Omega^1 = \Omega_{\kappa_{ms}}$ e $\Omega_{\kappa_{área}}$ e Ω_{knn} ou $\Omega^2 = \Omega_{\kappa_{mser}}$ e $\Omega_{\kappa_{área}}$ e Ω_{knn}	71
6.5 Comparação entre duas imagens de saída da nossa abordagem com o atributo κ_{ms} . O caso (a) apresenta um bom resultado enquanto o caso (b) mostra um caso em que a abordagem não deu bons resultados. A imagem que apresentou resultado ruim é da sub-base Ara2013-Rpi, que é a sub-base que contém imagens de qualidade inferior em comparação às demais.	72

6.6	Resultados obtidos pela estratégia de energia $\Omega_{\kappa_{ms}}$ através do conjunto de parâmetros. Note na linha vermelha o ganho de aproximadamente 4% obtido com a abordagem.	74
6.7	Exemplos de imagens de fundos de olhos e seus respectivos vasos sanguíneos segmentados.	75
6.8	Exemplo ilustrando a diferença de contraste entre os canais de uma imagem de retina f_{rgb} , note a ênfase de contraste nas regiões de vasos sanguíneos da banda f_g .	76
6.9	Passos da abordagem de pre-processamento aplicada em uma imagem de retina.	78
6.10	Representação da abordagem. A imagem f_g é a imagem do canal <i>green</i> e a imagem f_{ule} é a imagem de saída gerada pelo operador último leveling baseado nas estratégias combinadas $\Omega^1 = \Omega_{\kappa_{\Delta E}}$ e $\Omega_{\kappa_{área}}$ ou $\Omega^2 = \Omega_{\kappa_{ms}}$ e $\Omega_{\kappa_{área}}$.	79
6.11	Comparação entre algumas imagens resultantes da nossa abordagem com κ_{ms} . No caso (a) os vasos foram bem segmentados e o resultado foi bom, enquanto no caso (b) são apresentadas imagens onde a saída foi ruidosa e os resultados ruins.	79
6.12	Resultados comparativos de segmentações de duas imagens da base DRIVE. Note que, as imagens segmentadas pelo atributo funcional apresentam menos ruído. Os pixels em preto são verdadeiros positivos, em branco os verdadeiros negativos, em azul os falsos positivos, e em vermelho os falsos negativos	81
6.13	Resultados comparativos de segmentações de duas imagens da base STARE. Note que, as imagens segmentadas pelo funcional variacional apresentam menos ruído. Os pixels em preto são verdadeiros positivos, em branco os verdadeiros negativos, em azul os falsos positivos, e em vermelho os falsos negativos	81

LISTA DE ALGORITMOS

1	Algoritmo de <i>union-find</i>	41
2	Algoritmo para construção de árvores de componentes de Berger et al. (2007).	42
3	Algoritmo de ordenação baseado na fila de prioridade proposta por Falcao, Udupa e Miyazawa (2000)	44
4	Algoritmo que interpola a imagem em uma Grade Khalimsky simples.	45
5	Algoritmo de ordenação da Grade Khalimsky simples para construção da árvore de formas apresentado em Géraud et al. (2013).	46
6	Algoritmo que computa uma árvore de formas a partir de uma Grade Khalimsky simples.	47
7	Algoritmo para computação de atributos de área e volume para árvores morfológicas.	65
8	Implementação do algoritmo para computação do funcional variacional de Ballester et al. (2007).	66
9	Implementação do algoritmo para computação do atributo funcional de Xu, Géraud e Najman (2016).	67

LISTA DE ABREVIATURAS

MM	Morfologia Matemática
CC	Componente Conexo
EE	Elemento Estruturante
MS	Mumford-Shah
CNP	<i>Compact Node Pixel</i> (em português: pixel do nó compacto)
MSER	<i>Maximally Stable Extremal Regions</i> (em português: regiões extremais maximamente estáveis)
UAO	<i>Ultimate Attribute Opening</i> (em português: última abertura por atributo)
UAC	<i>Ultimate Attribute Closing</i> (em português: último fechamento por atributo)
UGF	<i>Ultimate Grain Filter</i> (em português: último filtro por grão)
RGB	<i>Red, Green and Blue</i> (em português: vermelho, verde e azul)
OCR	<i>Optical Character Recognition</i> (em português: reconhecimento ótico de caracteres)
<i>poset</i>	Acrônimo para a expressão em inglês <i>Partially Ordered Set</i> (em português: conjunto parcialmente ordenado)
<i>pixel</i>	Acrônimo para a expressão em inglês <i>Picture Element</i> (em português: elemento da imagem)

LISTA DE SÍMBOLOS

CONCEITOS BÁSICOS

\mathbb{Z}	Conjunto dos números inteiros
\mathbb{N}	Conjunto dos números naturais
\mathbb{R}	Conjunto dos números reais
\mathbb{R}^+	Conjunto dos números reais maiores ou iguais a zero
$. $	Cardinalidade de um conjunto
$\emptyset, \{\}$	Conjunto vazio
\in	Pertence ao conjunto
\notin	Não pertence ao conjunto
\cap	Operação de intersecção entre conjuntos
\cup	Operação de união entre conjuntos
\subseteq	É subconjunto de
\setminus	Operação de subtração entre conjuntos
\forall	Quantificador lógico universal
\exists	Quantificador lógico existencial
\Rightarrow	Condicional lógica, se então
\iff	Condicional lógica, se, e somente se
\rightsquigarrow	Símbolo que indica uma relação de ordem binária
\mathcal{X}^c	Complemento de um conjunto \mathcal{X}
$\{p : \star\}$	Conjunto de elementos que satisfazem uma propriedade \star
$\max\{p : \star\}$	Elemento máximo de um conjunto $\{p : \star\}$
$\min\{p : \star\}$	Elemento mínimo de um conjunto $\{p : \star\}$
$\arg \max_{p \in \mathcal{X}}\{f(p)\}$	Argumento $p \in \mathcal{X}$ que maximiza f
$\arg \min_{p \in \mathcal{X}}\{f(p)\}$	Argumento $p \in \mathcal{X}$ que minimiza f
\mathcal{O}	Notação assintótica do Big O
$(\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_n)$	Uma sequência qualquer de n conjuntos
$\sum_{p \in \mathcal{X}} f(p)$	Somatório dos valores de f aplicado nos elementos em \mathcal{X}

IMAGENS

f e g	Variáveis que representam imagens
$p, q, e \in \mathcal{D}$	Variáveis que representam pares (x, y) chamados de <i>pixels</i>
$f(p)$	Nível de cinza em f de um <i>pixel</i> p
\mathcal{D}	Conjunto de <i>pixels</i> que representa o domínio da imagem
\mathbb{K}	Conjunto que representa o contradomínio da imagem
\mathcal{X}, \mathcal{Z}	Variáveis utilizadas para representar subconjuntos de \mathcal{D}
\mathcal{A}	Relação de adjacência sobre \mathcal{D}
\mathcal{A}_4	Relação de adjacência <i>vizinhança-4</i> sobre \mathcal{D}
\mathcal{A}_8	Relação de adjacência <i>vizinhança-8</i> sobre \mathcal{D}
$\mathcal{A}(p)$	Conjunto de todos dos <i>pixels</i> vizinhos a p dado pela adjacência \mathcal{A}
$\mathcal{F}(\mathcal{D})$	Conjunto das imagens definidas com o domínio em \mathcal{D} e o contradomínio em \mathbb{K}
$\mathcal{P}(\mathcal{D})$	Conjunto das partes de \mathcal{D}
P	Uma partição sobre \mathcal{D}
R	Variável utilizada para representar uma região $R \in P$
$\mathcal{ZP}(f, \mathcal{A})$	Conjunto das zonas planas de f para uma adjacência \mathcal{A}
$\mathcal{G} = (\mathcal{V}, A_{\preceq})$	Variável utilizada para representar um grafo, onde \mathcal{V} é o conjunto de vértices e A_{\preceq} é o conjunto de arestas.
$\mathcal{G}_f = (\mathcal{D}, \mathcal{A})$	Um grafo de uma imagem f
ϕ e ψ	Variáveis que representam operadores de imagens

ÚLTIMOS LEVELINGS

$\mathcal{I}_{\preceq} \subseteq \mathbb{N}$	Conjunto de índices de uma família de primitivas, onde \preceq indica uma relação de pré-ordem
$\{\gamma_{\mathcal{B}_i} : i \in \mathcal{I}_{\geq}\}$	Família de primitivas aberturas por Elementos Estruturantes crescentes
$\{\varphi_{\mathcal{B}_i} : i \in \mathcal{I}_{\leq}\}$	Família de primitivas fechamentos por Elementos Estruturantes decrescentes
$\{\gamma_i^k : i \in \mathcal{I}_{\geq}\}$	Família de primitivas aberturas por atributos
$\{\varphi_i^k : i \in \mathcal{I}_{\leq}\}$	Família de primitivas fechamentos por atributos
\mathcal{R}_{θ}^{+} e \mathcal{R}_{θ}^{-}	Operador último <i>leveling</i> positivo, respectivamente negativo
$r_i^+(f)$ e $r_i^-(f)$	i -ésimo resíduo positivo e negativo, extraídos de f
$\mathcal{Nr}(i)$	Conjunto de nós removidos de \mathcal{T}_f^i para obter a sua versão podada \mathcal{T}_f^{i+1}
$r_{\mathcal{T}_f^i}^+(\tau)$ e $r_{\mathcal{T}_f^i}^-(\tau)$	i -ésimo resíduo positivo e negativo, aplicado em um dado nó $\tau \in \mathcal{Nr}(i)$
Ω	Mapeamento booleano que representa uma estratégia de filtragem residual para os últimos levelings

ÁRVORES

C	Variável que representa um componente conexo
$\mathcal{X}^\lambda(f)$	Conjunto de nível superior de f
$\mathcal{X}_\lambda(f)$	Conjunto de nível inferior de f
$\mathcal{U}(f)$	Conjunto dos CCs dos níveis superiores de f .
$\mathcal{L}(f)$	Conjunto dos CCs dos níveis inferiores de f .
$\mathcal{SAT}(f)$	Conjunto dos CCs dos níveis superiores e inferiores de f com os buracos preenchidos
$(\mathcal{U}(f), \subseteq)$	Árvore de componentes <i>max-tree</i>
$(\mathcal{L}(f), \subseteq)$	Árvore de componentes <i>min-tree</i>
$(\mathcal{SAT}(f), \subseteq)$	Árvore de formas ou <i>tree of shapes</i>
\mathcal{T}	Variável que representa uma árvore morfológica
κ	Variável que representa um atributo extraído de uma árvore morfológica
τ	Variável que representa um nó $\tau \in \mathcal{T}$
$\hat{\tau}$	Variável que representa um nó compacto que contém os pixels CNPs de um nó $\tau \in \mathcal{T}$
$(\mathcal{T}_1, \mathcal{T}_2, \dots, \mathcal{T}_n)$	Uma sequência qualquer de n árvores morfológicas

FUNÇÕES DE ENERGIA

∂P	Conjunto dos contornos de uma partição P
$\nu \in \mathbb{R}^+$	Variável que representa o valor de escala do funcional de Mumford-Shah
$E(f, \partial P)$	Funcional de Mumford-Shah aplicado sobre a imagem f e o conjunto ∂P dos contornos de uma partição P
$E\nu(f, P)$	Versão discreta do funcional de Mumford-Shah aplicada sobre a imagem f e uma partição P
$P_{\mathcal{T}}$	Partição associada dada por uma árvore morfológica \mathcal{T}
$\Delta E(\mathcal{T}, \tau)$	Funcional variacional aplicado a um nó $\tau \in \mathcal{T}$
$vol(\hat{\tau})$	Volume de um nó compacto $\hat{\tau} \in P_{\mathcal{T}}$
$E(\hat{\tau})$	Esperança de um nó compacto $\hat{\tau} \in P_{\mathcal{T}}$
$Var(\hat{\tau})$	Variância de um nó compacto $\hat{\tau} \in P_{\mathcal{T}}$
$\kappa_{ms}(\tau)$	Atributo funcional aplicado a um nó $\tau \in \mathcal{T}$
$\Omega_{\kappa_{ms}}(\tau)$	Estratégia construída sobre o atributo funcional aplicada a um nó $\tau \in \mathcal{T}$
$\varepsilon \in \mathbb{R}^+$	Variável que representa um valor de <i>threshold</i>
$\mathcal{G}_{\nabla}(\tau)$	Atributo de significância magnitude do gradiente ao longo do contorno aplicado a um nó $\tau \in \mathcal{T}$
$\nabla g(e)$	Valor do gradiente aplicado a um <i>pixel</i> e
\mathcal{T}^*	Árvore ótima obtida a partir de simplificações de uma árvore \mathcal{T}

INTRODUÇÃO

1.1 CONTEXTUALIZAÇÃO

Recentemente, muitos trabalhos têm demandado tempo e recursos no estudo de imagens digitais. Tipicamente, estamos interessados em reconhecer objetos nas cenas reproduzidas por essas imagens. Dessa forma, com a finalidade de compreender o conteúdo da cena, geralmente é necessário reconhecer os objetos presentes nela. A complexidade da cena e as características particulares de cada objeto tornam a tarefa de detecção do mesmo complexa, tendo em vista que esses objetos representam órgãos, células, caracteres, veículos, informações geográficas entre muitos outros (LONCARIC, 1998; CHENG; HAN, 2016). Nesse sentido, a forma de um objeto é uma importante característica básica que pode ser utilizada para representá-lo. Contudo, representar e descrever a forma do objeto é uma tarefa difícil, pois no mundo real os objetos estão presentes em um espaço 3D e durante a quantização eles são projetados em um plano 2D, ou seja, uma dimensão de informações é perdida (ZHANG; LU, 2004).

Na literatura são apresentadas em termos gerais, duas abordagens para representar a forma dos objetos, são elas: abordagens baseadas em regiões e em contornos (PAVLIDIS; LIOW, 1990; LONCARIC, 1998; ZHANG; LU, 2004). Assim, muitas técnicas para descrever formas têm sido propostas e estudadas ao longo do tempo, tais como: transformadas de Fourier (ZAHN; ROSKIES, 1972; PERSOON; FU, 1986), *medial axis transform* (MOTT-SMITH, 1970; BLUM; NAGEL, 1978), *chain code* (FREEMAN, 1961), energias (*context-energy-estimator* (XU; GéRAUD; NAJMAN, 2012), *functional variational* (BAL-LESTER et al., 2007) e *energy attribute* (XU; GéRAUD; NAJMAN, 2013)), entre outros (LONCARIC, 1998).

Levando em consideração o que foi apresentado, percebe-se que, os objetos de interesse em uma imagem muitas vezes não pertencem a uma escala específica, mas sim a diversas escalas, fato este que foi inicialmente observado por Koenderink (1984). Por isso, abordagens multiescalas têm sido estudadas ao longo dos anos. Uma importante área que concentra estudos neste sentido é a Morfologia Matemática (MM). Em poucas palavras, MM é o estudo e análise de formas e objetos utilizando teoria dos conjuntos e outras subáreas de matemática teórica. Contudo, ela não é somente teoria, mas também uma poderosa ferramenta de análise e processamento de imagens (SOILLE, 2003). Nesta dissertação, nós estudamos uma teoria invariante a escala que é definida no âmbito da MM. Essa teoria é previamente introduzida nas próximas seções e explorada com detalhes nos próximos capítulos.

1.2 PROBLEMA DE PESQUISA

Em meio aos estudos provados pela MM, focamos os estudos na área dos chamados operadores conexos. Esses operadores possuem propriedades muito interessantes a respeito de preservação de estruturas, pois como eles são construídos a partir da fusão de zonas planas adjacentes da imagem então não são gerados falsos contornos (SALEMBIER; OLIVERAS; GARRIDO, 1998). Dentre esses operadores, restringimos o estudo em uma especialização chamada *levelings*. O nome *leveling* está relacionado ao fato de que esses operadores trabalham nivellando a imagem. A forma que os *levelings* são construídos e a relação de um *leveling* para outro são fatores que são utilizados para construir um espaço de escalas. Esse espaço de escalas baseado em *levelings* contém representações da imagem cada vez mais simplificadas contendo diversas propriedades importantes. Esses fatos são evidenciados na Figura 1.1, onde pode ser observado que, a imagem original foi simplificada durante a construção dos *levelings* e o aninhamento resultante constitui o que chamamos de espaço de escalas.

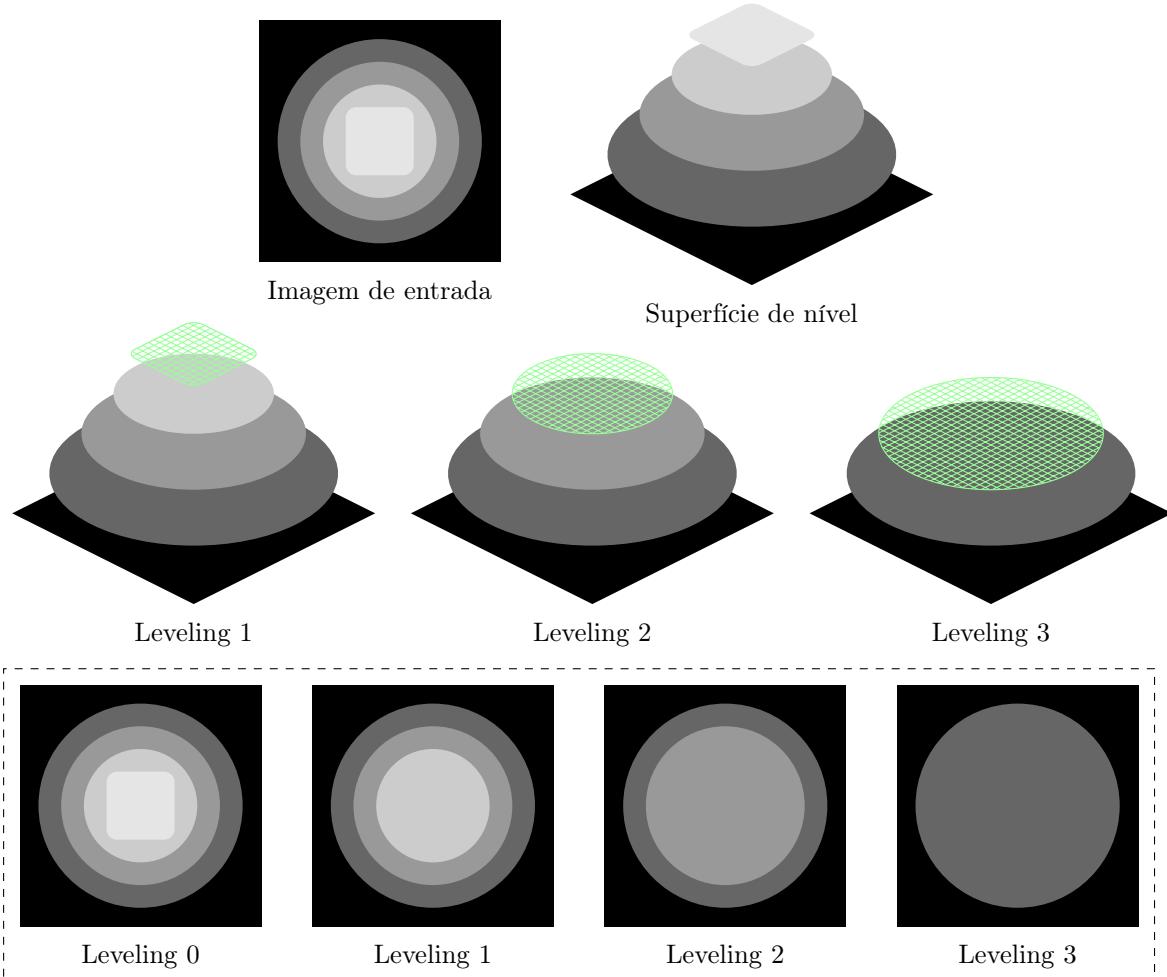


Figura 1.1: Exemplo de um espaço de escalas baseado em *levelings* construído a partir de uma imagem. As regiões em verde são as que foram removidas de uma escala para outra.

Nosso interesse está em extrair resíduos a partir de subtrações consecutivas em espaços de escalas baseado em *levelings*. Nesse caso, os operadores são chamados de *primitivas* e em particular a extração dos maiores resíduos dessa família de primitivas recebe o nome de últimos *levelings* (ALVES; HASHIMOTO; MARCOTEGUI, 2017). Na Figura 1.2 é apresentado um exemplo de computação de um operador último *leveling* em um espaço de escalas baseado em *levelings*, um fato importante a ser ressaltado é que os resíduos 0 e 1 foram sobrepostos pelo resíduo 2. A ocorrência de sobreposição de resíduos nem sempre é desejável, pois muitas vezes alguns resíduos podem ser indesejáveis (na Figura 1.2 por exemplo desejava-se detectar o resíduo 0). Dessa forma, uma estratégia interessante para minimizar esse problema é construir classificadores para selecionar os resíduos. Uma maneira de construir classificadores é utilizando descritores extraídos de regiões da imagem, tais como: área, perímetro, largura, altura, e também combinações dos mesmos, tais como: circularidade, retangularidade e momentos (ALVES; HASHIMOTO, 2014; ALVES; HASHIMOTO; MARCOTEGUI, 2017). Contudo, esses descritores muitas vezes não produzem resultados satisfatórios em problemas mais complexos.

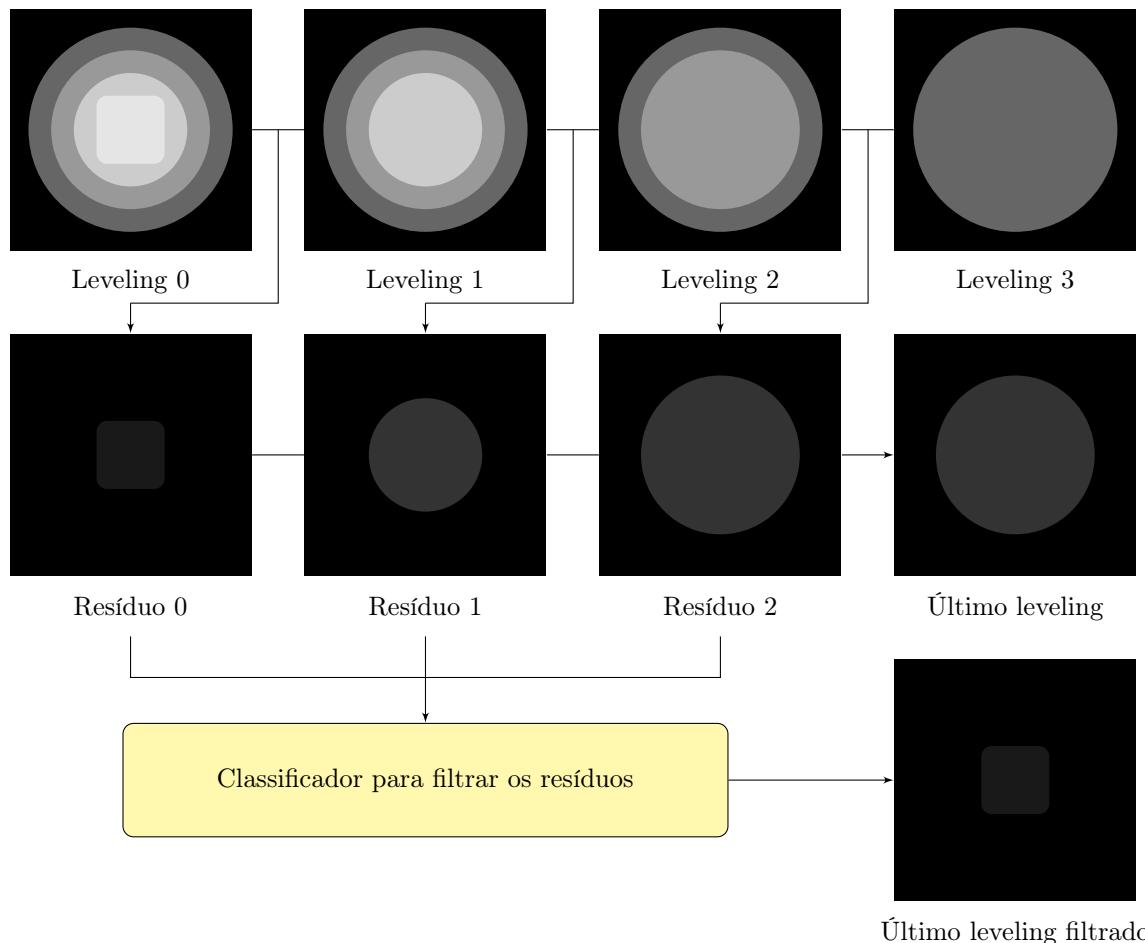


Figura 1.2: Computação dos últimos *levelings* a partir de subtrações consecutivas em um espaço de escalas. Note que, o resíduo 0 é a diferença entre o leveling 0 e o leveling 1 e assim sucessivamente.

Dado o exposto, pretende-se explorar nesta dissertação novas estratégias para filtrar resíduos indesejáveis dos últimos *levelings*. Mais especificamente, é proposto o uso de funções de energia para a construção dessas estratégias. As funções de energia ganharam grande notoriedade depois dos *snakes* (KASS; WITKIN; TERZOPoulos, 1988) e também do funcional de Mumford-Shah (MUMFORD; SHAH, 1989). Desde então, muitas pesquisas vêm sendo desenvolvidas nessa área. Neste trabalho nos restringimos ao funcional de Mumford-Shah. Essa abordagem é muita aplicada em segmentação de imagens, onde se define o problema como a busca por uma partição ótima, que nesse caso é aquela que tem a menor energia, ou seja, esse é um problema de otimização combinatória. Uma outra forma de entender essa abordagem é pensar nela como uma medida de avaliação, que relaciona um número real para cada partição, dessa forma a partição com maior qualidade é a que tem menor energia. Na Figura 1.3 é apresentado um exemplo que sumariza os conceitos do funcional de Mumford-Shah, onde cada ponto (partição, imagem) representa uma solução para o problema. Assim, veremos mais adiante nos resultados obtidos que estratégias de filtragem de resíduos dos últimos *levelings* baseadas em funções de energia são robustas e eficientes.

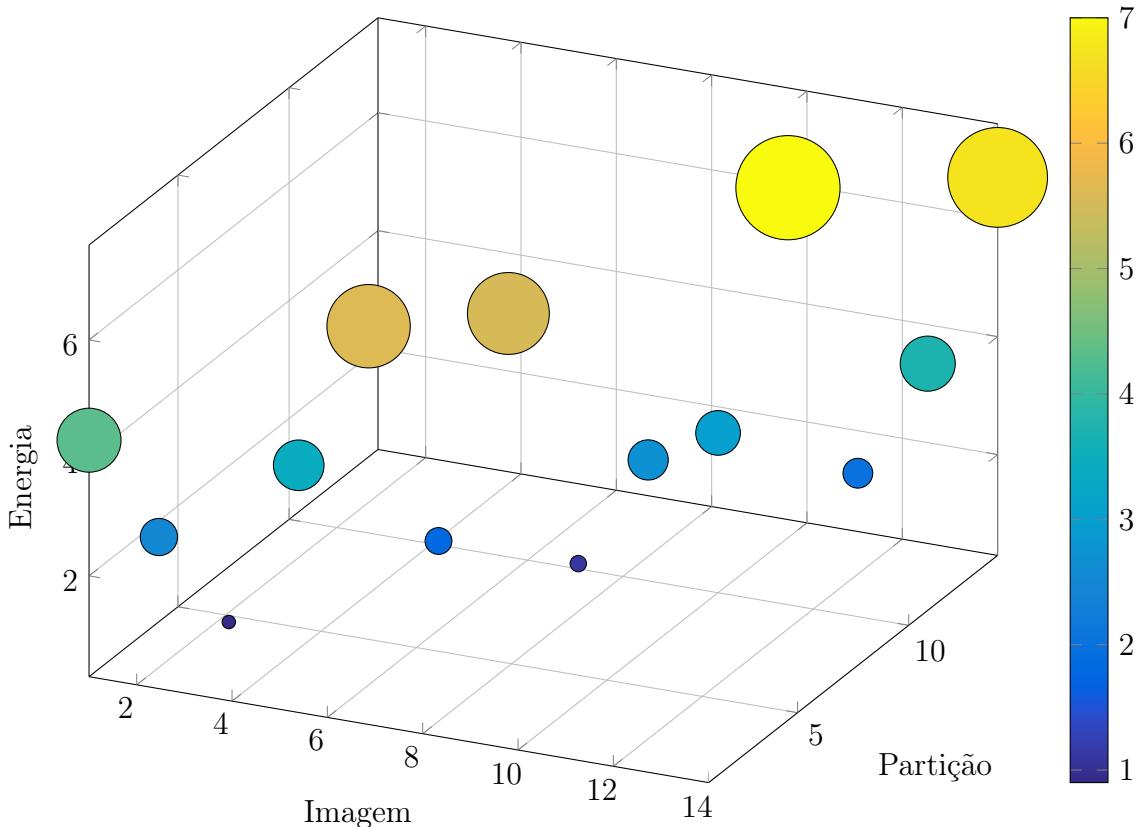


Figura 1.3: Exemplo de espaço de busca do funcional de Mumford-Shah para algumas partições e imagens correspondentes. Neste exemplo fica mais fácil visualizar uma interpretação do funcional de Mumford-Shah e a sua relação como uma medida de avaliação de partições. Cada ponto é uma possível solução para o problema, o tamanho da marca representa o valor de energia do funcional de Mumford-Shah. Note que, esse espaço de busca pode ser enorme.

1.3 OBJETIVOS

Esta dissertação tem como objetivo principal explorar técnicas de análise de formas no âmbito das funções de energia para construir novas estratégias com o intuito de classificar objetos de interesse em regiões desejáveis presentes nos resíduos numéricos extraídos pelos operadores últimos *levelings*. Mais especificamente, pretende-se:

- (i.) Investigar funções de energia derivadas da funcional de Mumford-Shah, restringindo-se aos trabalhos relacionados com árvores morfológicas.
- (ii.) Comparar estratégias baseadas em funções de energia com o MSER (MATAS et al., 2004) e o TBMR (XU et al., 2014) outras duas abordagens robustas para classificar resíduos dos últimos *levelings*.
- (iii.) Incluir implementações dos algoritmos explorados na biblioteca *open source* MM-Lib4J (Mathematical Morphology Library for Java) disponível no github:
<https://github.com/wonderalexandre/mmlib4j>.
- (iv.) Explorar aplicações práticas de estratégias baseadas em funções de energia em problemas de análise de imagens.

1.4 CONTRIBUIÇÕES DA PESQUISA

As principais contribuições apresentadas nessa dissertação são as seguintes:

- Novas estratégias para classificar resíduos dos últimos *levelings* a partir de funções de energia.
- Revisão literária do assunto voltada para abordagens aplicadas em árvores morfológicas, contendo propriedades e algoritmos eficientes para computação das estratégias baseadas em funções de energia.
- Implementação de algoritmos para computar as estratégias baseadas em funções de energias como *plugins* do ImageJ¹ utilizando a biblioteca *open source* de morfologia matemática MM-Lib4J.

1.5 ORGANIZAÇÃO DO TRABALHO

O restante deste trabalho está organizado em mais cinco capítulos que expõem e tratam dos seguintes tópicos:

¹ImageJ é um software *open source* feito em Java que contém uma série de algoritmos de visão computacional. Ele pode ser baixado [aqui](#).

- Capítulo 2 (Conceitos preliminares): Este capítulo apresenta alguns conceitos básicos sobre imagens que servem de base para o melhor entendimento dos assuntos abordados durante esta dissertação. Vale salientar que, grande parte dos conceitos aqui apresentados podem ser encontrados em um livro de topologia geral, como Newman (1939) ou Lipschutz (1971).
- Capítulo 3 (Operadores conexos e *levelings*): Este capítulo apresenta alguns conceitos sobre operadores conexos. Eles são importantes operadores morfológicos que possuem propriedades de preservação de contornos. Assim, trataremos de uma especialização dos operadores conexos: os *levelings*. Esses robustos operadores foram formalizados e inicialmente explorados por Fernand Meyer (MEYER, 1998a; MEYER, 1998b).
- Capítulo 4 (Representação de imagens a partir de árvores morfológicas): A representação de imagens a partir de árvores tem sido objeto de estudo há muitos anos. Na literatura vêm sendo desenvolvidos e mostrados algoritmos cada vez mais eficientes para tratar problemas que podem ser formalizados por grafos. Nesta abordagem constrói-se uma árvore a partir de uma imagem, realizam-se todas as etapas de extrações e modificações e então uma imagem é reconstruída (veja a Figura 4.1). Neste capítulo, uma classe de árvores a qual denotamos árvores morfológicas é obtida a partir dos conjuntos de níveis de uma decomposição da imagem por limiarização. Esses conjuntos induzem as árvores de componentes e a árvore de formas. Muitos dos conceitos abordados aqui podem ser encontrados em Caselles e Monasse (2009).
- Capítulo 5 (Últimos *levelings* baseados em funções de energia): Os últimos *levelings* são operadores robustos que analisam um espaço de escalas construído através de diferenças entre operadores *levelings* consecutivos e consideram os máximos resíduos. Em adição, os resíduos revelam informações importantes sobre o contraste de uma imagem. Assim, são apresentados conceitos sobre espaço de escalas baseado em *levelings* e a sua relação com árvores morfológicas, bem como a extração residual que da origem aos últimos *levelings*. Nesse contexto, é também apresentada uma revisão literária a respeito de funções de energia, que culmina na construção de estratégias para classificar resíduos dos últimos *levelings* com base em funções de energia. Muitos conceitos sobre últimos *levelings* mostrados aqui são encontrados em Alves (2013) e Alves, Hashimoto e Marcotegui (2017).
- Capítulo 6 (Resultados e experimentos): Este capítulo apresenta resultados das novas estratégias para filtragem dos últimos *levelings*. Duas aplicações são mostradas: reconhecimento de plantas via *bounding box detection* (atributo funcional) e segmentação de vasos sanguíneos de imagens de retina (atributo funcional e funcional

variacional). Os resultados obtidos revelam a robustez das funções de energia na filtragem de resíduos dos últimos *levelings*.

- Capítulo 7 (Conclusões): Finalmente, este capítulo apresenta conclusões sobre o trabalho aqui desenvolvido, além de perspectivas para trabalhos futuros.

CONCEITOS PRELIMINARES

Resumo do capítulo

Este capítulo apresenta alguns conceitos básicos sobre imagens que servem de base para o melhor entendimento dos assuntos abordados durante esta dissertação. Vale salientar que, grande parte dos conceitos aqui apresentados podem ser encontrados em um livro de topologia geral, como Newman (1939) ou Lipschutz (1971).

2.1 DEFINIÇÕES SOBRE IMAGENS

Existem diversas formas de se definir uma imagem. Neste trabalho, uma imagem f é vista como um mapeamento de uma grade retangular finita $\mathcal{D} \subset \mathbb{Z} \times \mathbb{Z}$ em um conjunto discreto $\mathbb{K} = \{0, \dots, K\} \subset \mathbb{Z}$ de níveis de cinza, onde $K = 2^b - 1$ é o nível de cinza máximo de uma imagem com $b > 0$ bits de profundidade (por exemplo: se $b = 8$, então $K = 255$). O conjunto que representa todas as possíveis imagens que podem ser construídas sobre \mathcal{D} é denotado por $\mathcal{F}(\mathcal{D})$. Os elementos do domínio da imagem são chamados *pixels*, ou seja, $p = (p_x, p_y) \in \mathcal{D}$ é um *pixel* onde o par (p_x, p_y) é respectivamente as coordenadas horizontal e vertical de p . A partir de um *pixel* p podemos obter seu valor de nível de cinza na imagem por $f(p)$. Uma imagem é dita binária quando $b = 1$ (isto é, $\mathbb{K} = \{0, 1\}$), e nesse caso frequentemente a imagem é representada como um subconjunto do domínio, ou seja, $\mathcal{X} = \{p \in \mathcal{D} : f(p) = 1\} \subseteq \mathcal{D}$. Pode-se também definir uma imagem multibanda (um exemplo é mostrado na Figura 2.1) como um vetor de imagens, ou seja, (f_a, f_b, \dots, f_z) onde os índices a, b, \dots, z são as bandas da imagem multibanda.

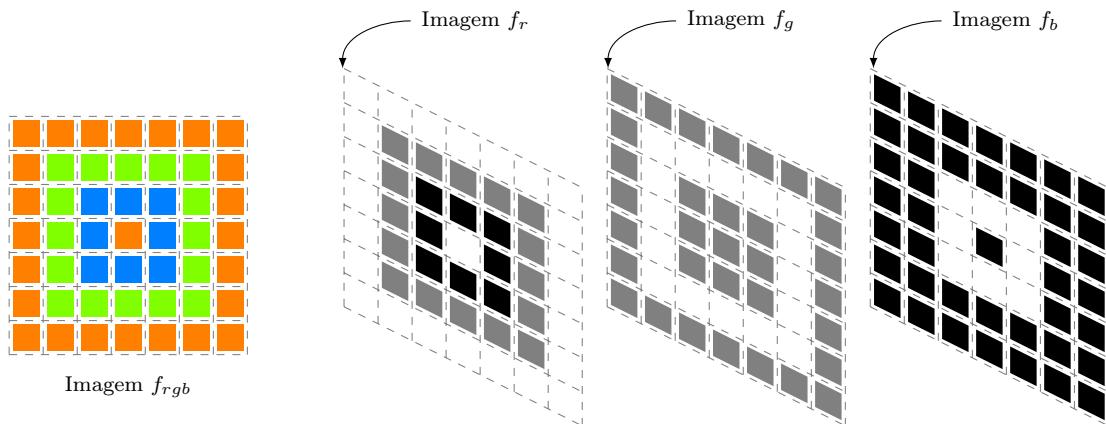


Figura 2.1: Exemplo de imagem multibanda f_{rgb} , as bandas são respectivamente: red, green e blue.

Muitas vezes a informação de um *pixel* não é suficiente para descrever a informação que temos interesse (veremos que utilizar regiões é uma abordagem muito mais vanta-

josa), dessa forma, na literatura muitos trabalhos, inclusive grande parte dos explorados nesta dissertação utilizam o conceito de conectividade entre os *pixels* e a sua vizinhança. Mais especificamente, um pixel p está conectado a outros pixels em seu entorno em \mathcal{D} e essa noção de conectividade entre os *pixels* nos leva à definir uma relação de adjacência, denominada \mathcal{A} sobre \mathcal{D} .

Definição 2.1 (Adjacência). Uma relação de adjacência (ou vizinhança) \mathcal{A} é uma relação binária entre os *pixels* de \mathcal{D} , isto é, $\mathcal{A} \subseteq \mathcal{D} \times \mathcal{D}$. Desta forma, se $(p, q) \in \mathcal{A}$ é dito que p é adjacente a q (ou p é vizinho de q).

Existem muitas formas de se construir uma adjacência, nesta dissertação são utilizadas apenas relações de adjacência circulares e simétricas, ou seja, $\mathcal{A} = \{(p, q) : p, q \in \mathcal{D}, d(p, q) \leq r\}$, onde $d(p, q) = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$ é a distância euclidiana entre p e q e r é um parâmetro de raio (o raio da menor circunferência que os *pixels* p e q estão contidos). Mais especificamente, duas adjacências muito conhecidas são utilizadas neste trabalho (veja alguns exemplos na Figura 2.2), uma delas quando $r = 1$, denotada por \mathcal{A}_4 ou *vizinhança-4* e a outra quando $r = \sqrt{2}$ denotada por \mathcal{A}_8 ou *vizinhança-8*. Nesta dissertação denota-se por $\mathcal{A}_{\mathcal{X}}(p) = \{q : p, q \in \mathcal{X}, p \text{ é vizinho de } q\}$ o conjunto contendo todos os *pixels* vizinhos a um dado pixel $p \in \mathcal{X}$.

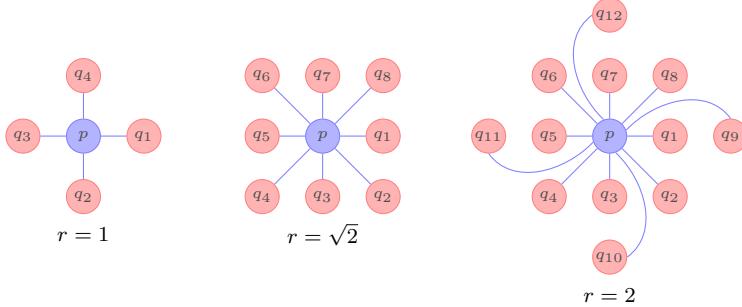


Figura 2.2: Exemplos de adjacências para um dado pixel p .

Uma imagem pode ser representada a partir de um grafo. Um grafo direcionado é uma dupla $G = (\mathcal{V}, \mathcal{E})$, onde \mathcal{V} é o conjunto de vértices e \mathcal{E} o conjunto de arestas. A partir de uma relação de adjacência podemos construir um grafo da imagem, também formando uma dupla, $G_f = (\mathcal{D}, \mathcal{A})$, onde os vértices (dados por \mathcal{D}) são os *pixels* do domínio da imagem, e as arestas são dadas pela adjacência, de forma que se $(p, q) \in \mathcal{A}$, então existe uma aresta que liga p até q . Dessa forma, baseado nas noções apresentadas sobre conectividade entre os *pixels*, definem-se:

Definição 2.2 (Caminho). Dados dois *pixels* p e q de $\mathcal{X} \subseteq \mathcal{D}$. Existe um caminho que leva de p até q , denotado por $\pi(p, q)$ se, e somente se, existir uma sequência de *pixels* (vértices em G_f) adjacentes, ou seja, (p_1, p_2, \dots, p_n) tal que $p_1 = p$, $p_n = q$ e $p_{i+1} \in \mathcal{A}_{\mathcal{X}}(p_i)$, para $1 \leq i \leq n - 1$.

Definição 2.3 (Componente conexo). Um componente conexo (CC) C é um subconjunto maximal de *pixels* de $\mathcal{X} \subseteq \mathcal{D}$, tal que para quaisquer dois *pixels* $p, q \in C$, existe um caminho de p até q em C . Desta forma, denota-se por $CC(\mathcal{X}, \mathcal{A})$ o conjunto de todos os CCs de \mathcal{X} dados de acordo com uma adjacência \mathcal{A} . Na Figura 2.3 são mostrados exemplos de conjuntos de CCs para adjacências \mathcal{A}_4 e \mathcal{A}_8 .

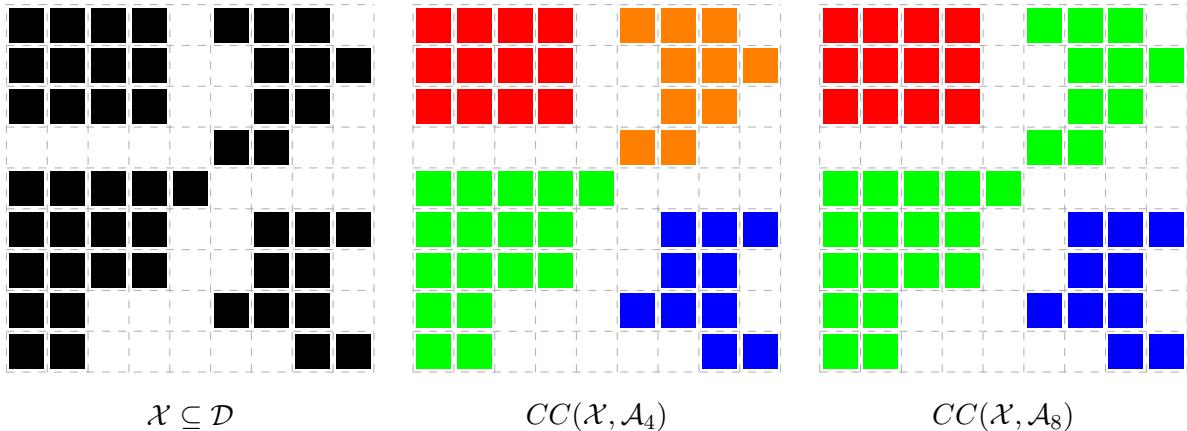


Figura 2.3: Exemplos de conjuntos de componentes conexos para adjacências \mathcal{A}_4 e \mathcal{A}_8 . Note que, cada cor representa um CC em \mathcal{X} e os componentes laranja e verde na vizinhança-4 se tornaram apenas um componente em verde na vizinhança-8.

Definição 2.4 (Zona plana). Uma zona plana C de uma imagem f é um subconjunto maximal de *pixels* de \mathcal{D} , tal que para quaisquer dois *pixels* $p, q \in C$, existe uma sequência de *pixels* (p_1, p_2, \dots, p_n) , tal que $p_1 = p$, $p_n = q$, satisfazendo $p_{i+1} \in \mathcal{A}(p_i)$ e $f(p_i) = f(p_{i+1})$, para $1 \leq i \leq n - 1$. Denota-se por $\mathcal{ZP}(f, \mathcal{A})$ o conjunto das zonas planas presentes em uma imagem f definidas com a relação de adjacência \mathcal{A} . Na Figura 2.4 é mostrado um exemplo de extração de zonas planas de f para a adjacência \mathcal{A}_4 .

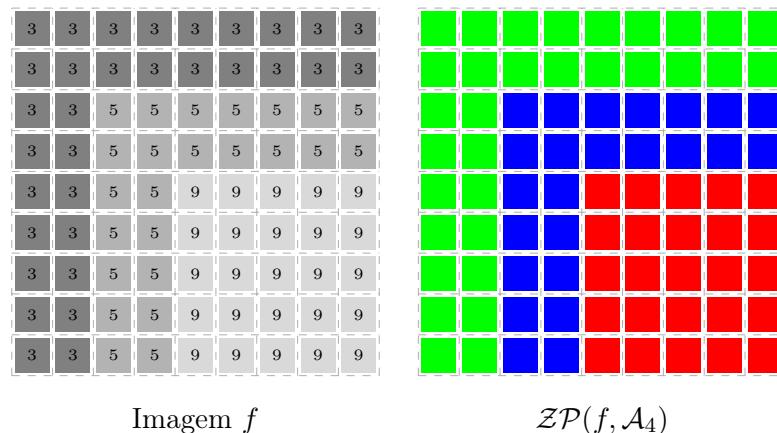


Figura 2.4: Exemplo de extração de zonas planas para uma imagem f sobre uma adjacência \mathcal{A}_4 . Note que, cada cor representa uma zona plana em $\mathcal{ZP}(f, \mathcal{A}_4)$.

Definição 2.5 (Extrema regional). Uma zona plana C é uma máxima (respectivamente, mínima) regional de uma imagem f , se $\forall p \in C, f(p) \geq f(q)$ (respectivamente, $f(p) \leq f(q)$) e $q \in \mathcal{A}(p)$. Desta forma, C é uma extrema regional se ela é mínima ou máxima regional. A região em vermelho da Figura 2.4 é uma máxima regional, enquanto a região em verde é uma mínima regional.

Definição 2.6 (Partição). Uma partição P sobre o conjunto \mathcal{D} é um conjunto não vazio de n regiões $\{R_1, R_2, \dots, R_n\}$ distintas de \mathcal{D} , onde a união destas regiões é igual ao conjunto \mathcal{D} , isto é:

- (i.) Nenhum elemento de P é vazio, ou seja, se $R \in P$ então $R \neq \emptyset$;
- (ii.) Os elementos de P são disjuntos, ou seja, se $R_i, R_j \in P$ então $R_i \cap R_j = \emptyset$;
- (iii.) A união dos elementos de P é igual a \mathcal{D} , ou seja, $\cup\{R : R \in P\} = \mathcal{D}$.

O conjunto de todas as partições sobre \mathcal{D} é denotado por $\mathbb{P}(\mathcal{D})$, isto é, $\mathbb{P}(\mathcal{D}) = \{P : P \text{ é uma partição sobre } \mathcal{D}\}$. Neste trabalho, denota-se por P_f uma partição de \mathcal{D} construída pelas zonas planas de uma imagem f e uma adjacência \mathcal{A} , isto é, $P_f = \{R : R \in \mathcal{ZP}(f, \mathcal{A})\}$. Também é denotado por $P_f(p)$ a região em P_f que contém o pixel $p \in \mathcal{D}$.

OPERADORES CONEXOS E LEVELINGS

Resumo do capítulo

Este capítulo apresenta alguns conceitos sobre operadores conexos. Eles fazem parte de uma importante classe de operadores morfológicos que possuem propriedades de preservação de contornos. Assim, trataremos de uma especialização dos operadores conexos: os levelings. Esses robustos operadores foram formalizados e inicialmente explorados por Fernand Meyer (MEYER, 1998a; MEYER, 1998b).

3.1 OPERADORES CONEXOS

Conforme brevemente introduzido, operadores conexos podem ser computados a partir da fusão de zonas planas adjacentes da imagem. Dessa forma, os operadores conexos possuem propriedades de preservação de estruturas, no sentido que as fusões de zonas planas adjacentes não criam novos contornos, apenas mantém os que já existiam na imagem (SALEMBIER; OLIVERAS; GARRIDO, 1998). Esses operadores têm sido usados em um grande número de aplicações, tais como: filtragem (JONES, 1999; SALEMBIER; SERRA, 1995) e segmentação (MARQUÉS; VILAPLANA, 2002) entre outras. Na Figura 3.1 são apresentados exemplos de partições obtidas por operadores conexos, note que, a fusão entre as regiões em azul e vermelho (respectivamente, laranja e amarelo) não gerou novos contornos.

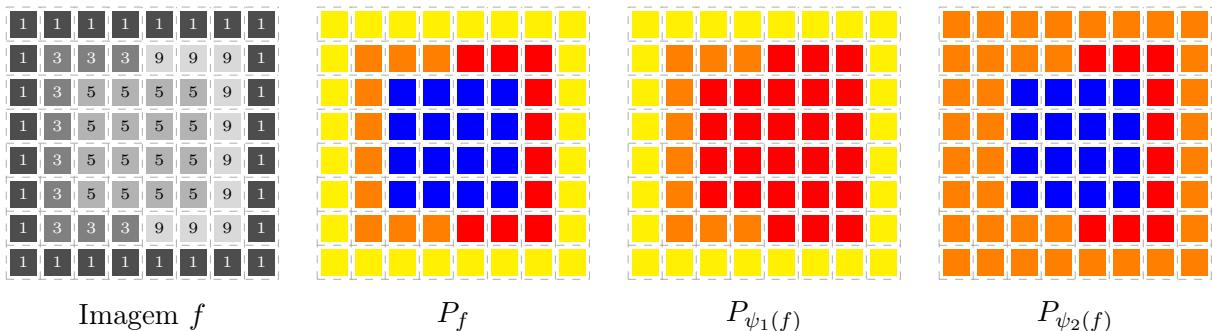


Figura 3.1: Exemplo de partições obtidas pela aplicação de operadores conexos ψ_1 e ψ_2 em f para a adjacência A_4 . Note que, cada cor representa uma região.

Em seu trabalho Meyer (1998a) formalizou o conceito de operadores conexos a partir das fronteiras entre regiões. Ele concluiu que, um operador é conexo quando existe uma transição de nível entre dois *pixels* adjacentes, digamos p e $q \in \mathcal{A}$ na imagem de entrada (isto é, $f(p) \neq f(q)$) e ela também ocorre nos mesmos *pixels* na imagem de saída resultante do operador (isto é, $[\psi(f)](p) \neq [\psi(f)](q)$). Isso nos leva a definição de operador conexo.

Definição 3.1 (Operador conexo (MEYER, 1998a; MEYER, 1998b)). Um operador $\varphi : \mathcal{F}(\mathcal{D}) \rightarrow \mathcal{F}(\mathcal{D})$ é dito conexo, se $\forall f \in \mathcal{F}(\mathcal{D})$, a seguinte relação é válida para todos os pares de *pixels* adjacentes, ou seja, $\forall (p, q) \in \mathcal{A}$, temos que:

$$[\psi(f)](p) \neq [\psi(f)](q) \Rightarrow f(p) \neq f(q). \quad (3.1)$$

Durante a computação dos operadores conexos novas zonas planas são criadas a partir da fusão de zonas planas adjacentes. Dessa forma, muitas zonas planas podem se tornar extremas regionais na imagem de saída (SALEMBIER; SERRA, 1995). Explorando esse fato, Fernand Meyer (MEYER, 1998a; MEYER, 1998b) estudou duas especializações dos operadores conexos: os *levelings* e os *monotone planings*.

3.1.1 LEVELINGS

Os operadores *levelings* são uma importante especialização de operadores conexos. Eles são poderosos e robustos filtros de simplificação que não criam novas estruturas (extremas regionais e contornos), e além disso seus valores são delimitados pela vizinhança de *pixels* (MEYER, 2010). Na literatura, normalmente os *levelings* são vistos como a intersecção de duas subclasses de operadores: os *lower-levelings* e os *upper-levelings* (MEYER; MARAGOS, 2000).

Definição 3.2 (Operador *lower-leveling* (MEYER, 1998a; MEYER, 1998b)). Um operador $\psi : \mathcal{F}(\mathcal{D}) \rightarrow \mathcal{F}(\mathcal{D})$ é dito *lower-leveling*, se $\forall f \in \mathcal{F}(\mathcal{D})$, a seguinte relação é válida para todos os pares de *pixels* adjacentes, ou seja, $\forall (p, q) \in \mathcal{A}$, temos que:

$$[\psi(f)](p) > [\psi(f)](q) \Rightarrow [\psi(f)](q) \geq (f)(q). \quad (3.2)$$

Definição 3.3 (Operador *upper-leveling* (MEYER, 1998a; MEYER, 1998b)). Um operador $\psi : \mathcal{F}(\mathcal{D}) \rightarrow \mathcal{F}(\mathcal{D})$ é dito *upper-leveling*, se $\forall f \in \mathcal{F}(\mathcal{D})$, a seguinte relação é válida para todos os pares de *pixels* adjacentes, ou seja, $\forall (p, q) \in \mathcal{A}$, temos que:

$$[\psi(f)](p) > [\psi(f)](q) \Rightarrow (f)(p) \geq [\psi(f)](p). \quad (3.3)$$

Os operadores *lower-levelings* (respectivamente, *upper-levelings*) removem detalhes de mínimas (respectivamente, máximas) regionais a partir das fusões de zonas planas, dessa forma eles ampliam as regiões escuras (respectivamente, claras) da imagem (veja um exemplo na Figura 3.2).

Definição 3.4 (Operador *leveling* (MEYER, 1998a; MEYER, 1998b)). Um operador $\psi : \mathcal{F}(\mathcal{D}) \rightarrow \mathcal{F}(\mathcal{D})$ é dito *leveling*, se $\forall f \in \mathcal{F}(\mathcal{D})$, a seguinte relação é válida para todos

os pares de *pixels* adjacentes, ou seja, $\forall(p, q) \in \mathcal{A}$, temos que:

$$[\psi(f)](p) > [\psi(f)](q) \Rightarrow f(p) \geq [\psi(f)](p) \wedge [\psi(f)](q) \geq f(q). \quad (3.4)$$

Note que, segundo a Definição 3.4 se ocorre uma transição de (p, q) em $\psi(f)$ então essa transição também ocorre em f . Isso acontece porque, $[\psi(f)](p) > [\psi(f)](q) \Rightarrow f(p) \geq [\psi(f)](p) > [\psi(f)](q) \geq f(q)$. Além disso, o intervalo de transição $[[\psi(f)](q), [\psi(f)](p)]$ está contido no intervalo $[f(q), f(p)]$. Esses fatos podem ser observados na Figura 3.2 através de exemplos de operadores *levelings*.

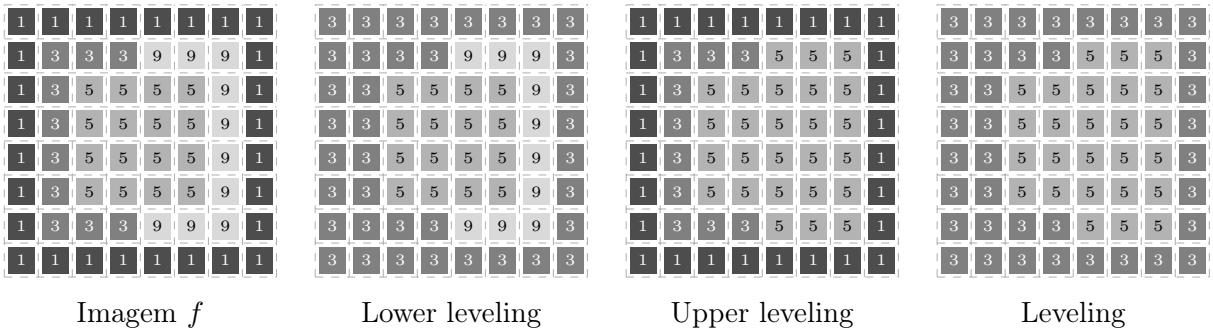


Figura 3.2: Exemplos de operadores levelings aplicados em uma imagem f .

Como já é conhecido, os operadores conexos podem ser computados a partir da fusão de zonas planas da imagem. Nesta dissertação, estamos interessados em decompor uma imagem de forma hierárquica utilizando os conjuntos de níveis. Esses conjuntos podem ser eficientemente representados por árvores morfológicas. Essas árvores armazenam os componentes conexos da imagem em uma relação de inclusão hierárquica. Dessa forma, a fusão de zonas planas pode ser computada simplesmente pela remoção de componentes da árvore. Além do mais, essas árvores permitem induzir um espaço de escalas baseado em *levelings*. Esse conceito de espaço de escalas será introduzido na próxima seção, e então, mais adiante veremos a relação que permite construí-lo através das árvores morfológicas.

3.1.2 ESPAÇO DE ESCALAS BASEADO EM LEVELINGS

Em muitos problemas de análise e processamento de imagens os objetos de interesse não estão restritos a uma escala específica, mas sim a diversas escalas. Nesse sentido, abordagens multiescalas têm sido desenvolvidas, onde uma série de representações cada vez mais *grosseiras* da imagem são construídas de forma a obter um espaço de escalas. Os *levelings* podem ser aninhados para gerar este espaço de escalas. Mais especificamente, conforme mostrado em Meyer e Maragos (2000), se ignorarmos imagens constantes, para uma dada imagem $f \in \mathcal{F}(\mathcal{D})$ pode-se construir uma sequência de *levelings* aninhados ($g_0 = f, g_1, \dots, g_n$), onde g_i é *leveling* de g_{i-1} , e como consequência da transitividade g_i é também *leveling* de g_j , para todo $j < i$. Essa sequência de *levelings* simplificam mais e

mais a imagem f , e constituem um espaço de escalas com as seguintes propriedades:

- (i.) **Invariâncias:** Os *levelings* são invariantes a transformações de rotação, translação e mudanças de iluminação, isto é, como as adjacências são simétricas então temos as invariâncias a rotação e translação, e a respeito da invariância a iluminação é conhecido que se g_i é *leveling* de g_{i-1} e, g_i e g_{i-1} forem submetidos a uma transformação linear α então $\alpha(g_i)$ ainda é *leveling* de $\alpha(g_{i-1})$.
- (ii.) **Simplificação:** Na transição entre duas escalas acontece uma simplificação real da imagem, onde algumas informações são perdidas de uma escala para outra.
- (iii.) **Princípio do máximo:** Uma forma particular de simplificação é expressa pelo princípio do máximo: em cada transição de escala, a intensidade máxima em escalas mais grosseiras é sempre menor do que a intensidade máxima das escalas finas.
- (iv.) **Princípio da causalidade:** O que ocorre em escalas mais grosseiras só pode ser causado pelo que aconteceu em escalas mais finas.
- (v.) **Fidelidade:** Não criar novas estruturas em escalas mais grosseiras: cada mínima (respectivamente, máxima) regional do *leveling* contém uma mínima (respectivamente, máxima) regional da imagem de referência.

REPRESENTAÇÃO DE IMAGENS A PARTIR DE ÁRVORES MORFOLÓGICAS

Resumo do capítulo

A representação de imagens a partir de árvores tem sido objeto de estudo há muitos anos. Na literatura vêm sendo desenvolvidos e mostrados algoritmos cada vez mais eficientes para tratar problemas que podem ser formalizados por grafos. Nesta abordagem constrói-se uma árvore a partir de uma imagem, realizam-se todas as etapas de extrações e modificações e então uma imagem é reconstruída (veja a Figura 4.1). Neste capítulo, uma classe de árvores a qual denotamos árvores morfológicas é obtida a partir dos conjuntos de níveis de uma decomposição da imagem por limiarização. Esses conjuntos induzem as árvores de componentes e a árvore de formas. Muitos dos conceitos abordados aqui podem ser encontrados em Caselles e Monasse (2009).

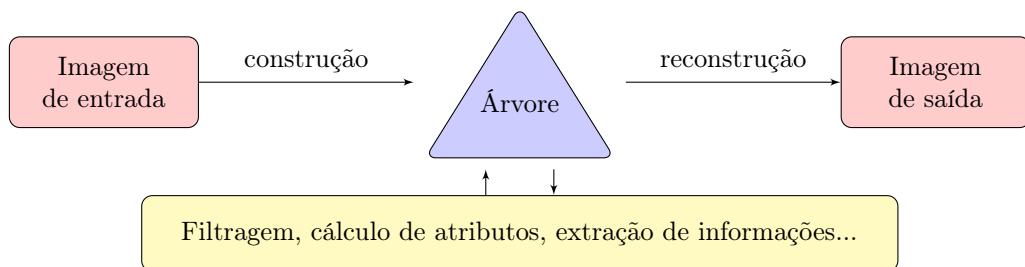


Figura 4.1: Representação de imagens através de árvores.

4.1 INTRODUÇÃO

Antes de apresentar conceitos sobre árvores morfológicas é necessário introduzir o conceito geral de árvores a partir de teoria dos conjuntos, onde com algumas propriedades pode-se concluir que uma árvore é um conjunto parcialmente ordenado (do inglês, *poset*). A partir dessa conceituação geral de árvores, pode-se deduzir que os conjuntos de níveis gerados pela decomposição de uma imagem por limiares são árvores. Isso consequentemente nos leva a definir as árvores de componentes e de formas, que são chamadas nesta dissertação de árvores morfológicas.

4.2 ÁRVORES E CONJUNTOS PARCIALMENTE ORDENADOS

Uma relação binária \preceq sobre \mathcal{V} é dita relação de ordem se, e somente se, ela obedece três propriedades: (i) transitividade: $\forall a, b, c \in \mathcal{V}$, se $a \preceq b$ e $b \preceq c$ então $a \preceq c$; (ii) reflexividade: $\forall x \in \mathcal{V}, x \preceq x$; (iii) anti-simetria: $\forall a, b \in \mathcal{V}$, se $a \preceq b$ e $b \preceq a$ então $a = b$.

Dessa forma, o par $(\mathcal{V}, \preccurlyeq)$ é chamado de *conjunto ordenado*. Quando existem elementos em \mathcal{V} que não são comparáveis, então o par $(\mathcal{V}, \preccurlyeq)$ é chamado de *conjunto parcialmente ordenado* (do inglês, *partially ordered set* ou simplesmente *poset*).

Uma ferramenta muito conhecida em teoria da ordem é o *Diagrama de Hasse* (BRÜG-GEMANN; PATIL, 2011). Podemos representar um *poset* por um diagrama de Hasse na forma de um grafo não direcionado $\mathcal{G} = (\mathcal{V}, A_{\preccurlyeq})$, onde \mathcal{V} é um conjunto de vértices e A_{\preccurlyeq} é um conjunto de arestas definidas pela seguinte propriedade: uma aresta $\{v, w\} \in A_{\preccurlyeq}$ se, e somente se, $v, w \in \mathcal{V}$, $v \neq w$, v e w são comparáveis ($v \preccurlyeq w$ ou $w \preccurlyeq v$) e $\nexists x \in \mathcal{V}$, $x \neq v$ e $x \neq w$, tal que ou $v \preccurlyeq x \preccurlyeq w$, ou $w \preccurlyeq x \preccurlyeq v$. Para elucidar o assunto, na Figura 4.2 é mostrado um exemplo com dois *posets*, onde os vértices \mathcal{V} são representados por círculos e as arestas por segmentos de retas que conectam os vértices.

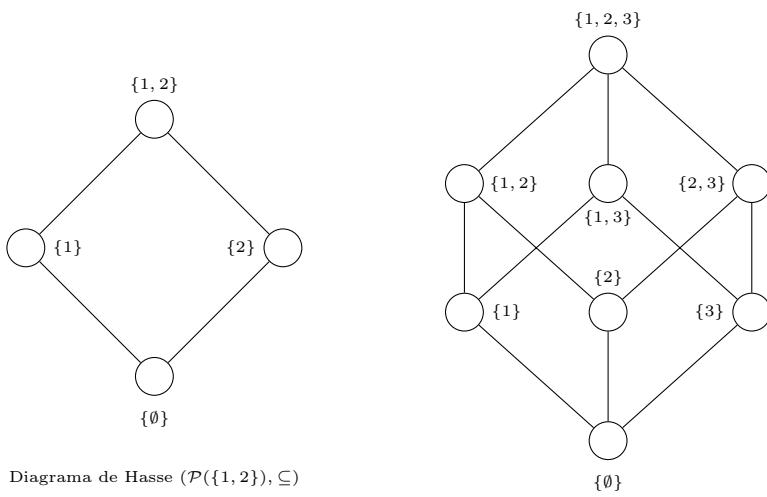


Figura 4.2: Exemplos de diagramas de Hasse para dois posets.

Levando em conta o que foi previamente introduzido, um grafo $\mathcal{G} = (\mathcal{V}, A_{\preccurlyeq})$ do diagrama de Hasse é uma árvore se, e somente se, o *poset* $(\mathcal{V}, \preccurlyeq)$ verifica as propriedades da Definição 4.1.

Definição 4.1 (Árvore). Um grafo $\mathcal{G} = (\mathcal{V}, A_{\preccurlyeq})$ do diagrama de Hasse de um *poset* $(\mathcal{V}, \preccurlyeq)$ é uma árvore se, e somente se, as seguinte propriedades são verificadas:

- (i) $\exists R \in \mathcal{V}$, tal que $\forall A \in \mathcal{V}$, $A \preccurlyeq R$. Nesse caso, o elemento R é chamado raiz da árvore.
- (ii) $\forall A, B \in \mathcal{V}$, se $A \cap B \neq \emptyset$ então ou $A \preccurlyeq B$ ou $B \preccurlyeq A$. Nos últimos dois casos, é dito que A e B estão aninhados.

A primeira condição da Definição 4.1 verifica a conexidade do *poset* (isto é, todo vértice em \mathcal{V} é comparável com a raiz R da árvore) e a segunda condição verifica que não há ciclos no *poset*. Dessa forma, denotamos o *poset* (agora chamado de árvore) por $\mathcal{T} = (\mathcal{V}, \preccurlyeq)$. Assim, apresentamos algumas definições gerais sobre uma árvore \mathcal{T} (veja a Figura 4.3).

Definição 4.2 (Raiz). Um vértice R é dito vértice raiz da árvore \mathcal{T} se, e somente se, $\forall A \in \mathcal{V}, A \preceq R$. Dessa forma, denota-se por $Raiz(\mathcal{T})$ o vértice R raiz da árvore, isto é, $Raiz(\mathcal{T}) = R$.

Definição 4.3 (Pai). Dados dois vértices $P, B \in \mathcal{V}$, é dito que P é pai de B se, e se somente se, $B \preceq P$ e $\nexists C \in \mathcal{V}, C \neq P, C \neq B$, tal que $B \preceq C \preceq P$. Dessa forma, denota-se por $Pai(B)$ o vértice P pai de B , isto é, $Pai(B) = P$.

Definição 4.4 (Filhos). Os vértices de \mathcal{T} que têm como pai o vértice $P \in \mathcal{V}$ são chamados de filhos de P e são representados pelo conjunto $Filhos(P) = \{A : A \in \mathcal{V}, Pai(A) = P\}$.

Definição 4.5 (Folha). Um vértice $F \in \mathcal{V}$ é chamado de folha se, e somente se, o vértice F não possui filhos em \mathcal{T} , ou seja, $Filhos(F) = \emptyset$.

Definição 4.6 (Ascendente e Descendente). Dados dois vértices $A, B \in \mathcal{V}$, se $A \preceq B$ então é dito que B é ascendente de A ou que A é descendente B .

Definição 4.7 (Intervalo). Dados dois vértices $A, B \in \mathcal{V}$, se $A \preceq B$ então o conjunto de vértices comparáveis entre A e B é dito intervalo entre A e B e é denotado por $[A, B]_{\mathcal{T}} = \{I : I \in \mathcal{V}, A \preceq I \preceq B\}$.

Definição 4.8 (Bifurcação). Dados dois vértices $A, B \in \mathcal{V}$, se $A \preceq B$ então existe uma bifurcação entre A e B se $\exists C \in \mathcal{V}$, tal que $C \preceq B$ e A e C não são comparáveis.

Definição 4.9 (Ramo). Dados dois vértices $A, B \in \mathcal{V}$, um intervalo $[A, B]_{\mathcal{T}}$ é chamado ramo da árvore \mathcal{T} , se não existir uma bifurcação entre A e B .

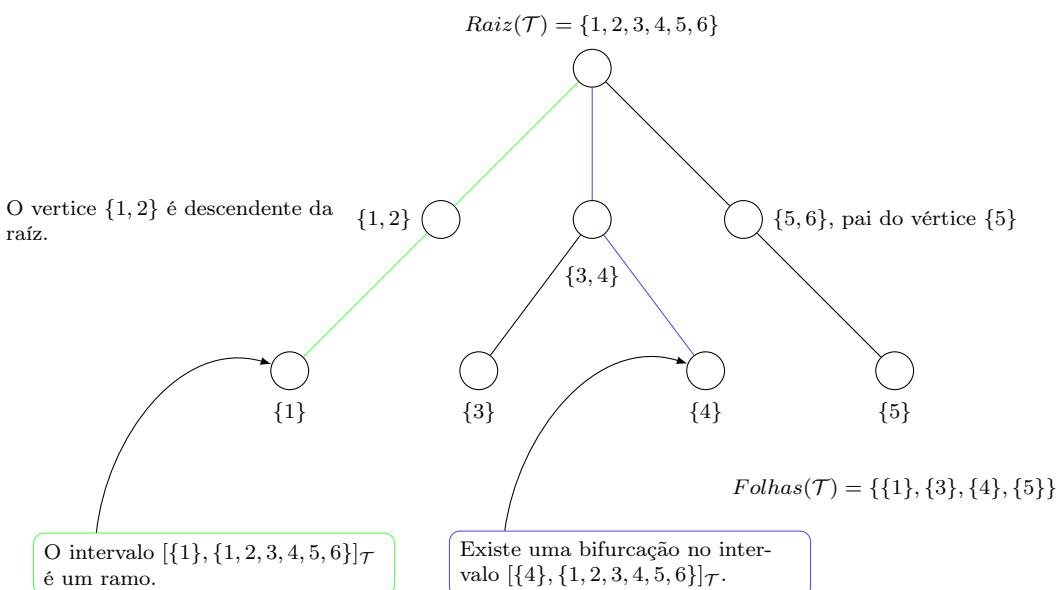


Figura 4.3: Definições gerais de árvores em um poset $\mathcal{T} = (\{\{1\}, \{3\}, \{4\}, \{5\}, \{1, 2\}, \{3, 4\}, \{5, 6\}, \{1, 2, 3, 4, 5, 6\}\}, \preceq)$.

Uma operação muito importante a respeito de árvores é a poda (veja a Figura 4.4). A poda é uma operação sobre uma árvore \mathcal{T} onde geramos uma árvore \mathcal{T}' pela remoção de alguns vértices e arestas de \mathcal{T} , isto é, a poda sobre \mathcal{T} gera uma nova árvore \mathcal{T}' . Assim, é dito que \mathcal{T}' é obtida por uma operação de poda em \mathcal{T} se a Definição 4.10 é verdadeira.

Definição 4.10 (Poda). Diz-se que uma árvore $\mathcal{T}' = (\mathcal{V}', \preccurlyeq)$ é obtida por uma poda sobre \mathcal{T} se, e somente se, $\mathcal{V}' \subseteq \mathcal{V}$ e para qualquer vértice $A \in \mathcal{V}'$, $\nexists B \in \{\mathcal{V} \setminus \mathcal{V}'\}$ tal que $A \preccurlyeq B$. Dessa forma, a poda é denotada por $\mathcal{T}' = Poda(\mathcal{T})$.

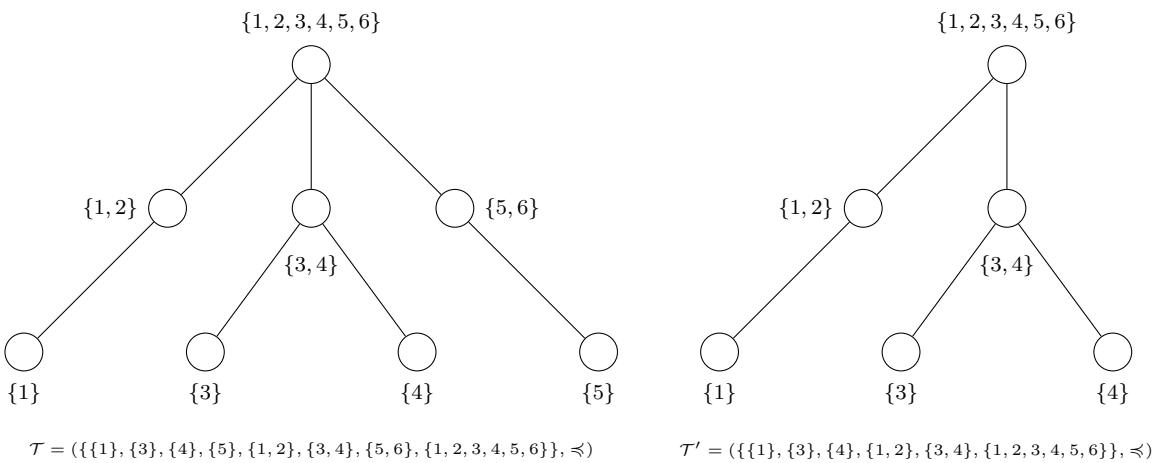


Figura 4.4: Exemplo de uma poda em uma dada árvore \mathcal{T} .

Sob uma outra perspectiva, a operação de poda é vista como uma simplificação da árvore. Uma outra forma de simplificar uma árvore é a partir da remoção de um nó. Isso nos leva a Definição 4.11.

Definição 4.11 (Remoção de um nó). Chama-se remoção de um nó, a operação que gera uma árvore simplificada $\mathcal{T}' = (\mathcal{V}', \preccurlyeq)$ a partir da remoção de um vértice $A \in \mathcal{V}$, tal que $\mathcal{V}' = \{\mathcal{V} \setminus A\}$ e $A \neq Raiz(\mathcal{T})$. Essa operação é denotada por $\mathcal{T}' = Remove(\mathcal{T}, A)$.

4.3 ÁRVORES A PARTIR DE CONJUNTOS DE NÍVEIS

A MM concentra estudos em objetos que são invariantes a contraste. No sentido global, os conjuntos de níveis são representações de imagens que armazenam em sua relação tal propriedade. Esses conjuntos são construídos a partir do conjuntos de *pixels* de f e são classificados como superiores e inferiores. O conjunto de níveis superiores (respectivamente, inferiores) é um conjunto de *pixels* de f que estão acima (respectivamente, abaixo) de um nível de cinza $\lambda \in \mathbb{K}$. Mais especificamente, esses conjuntos são definidos da seguinte forma:

Definição 4.12 (Conjunto de nível). O conjunto de nível superior $\mathcal{X}^\lambda(f)$ e o conjunto de nível inferior $\mathcal{X}_\lambda(f)$ de uma imagem f para um nível $\lambda \in \mathbb{K}$, são definidos por:

$$\mathcal{X}^\lambda = \{p : p \in \mathcal{D}, f(p) \geq \lambda\} \text{ e } \mathcal{X}_\lambda = \{p : p \in \mathcal{D}, f(p) < \lambda\} \quad (4.1)$$

Um fato importante a respeito destes conjuntos é que eles são aninhados pela relação de inclusão. Mais especificamente, os conjuntos de níveis superiores são decrescentes:

$$\mathcal{X}^{\lambda_i}(f) \subseteq \mathcal{X}^{\lambda_{i-1}}(f) \subseteq \dots \subseteq \mathcal{X}^{\lambda_0}(f) \quad (4.2)$$

respectivamente, os conjuntos de níveis inferiores são crescentes:

$$\mathcal{X}_{\lambda_1}(f) \subseteq \mathcal{X}_{\lambda_2}(f) \subseteq \dots \subseteq \mathcal{X}_{\lambda_{i+1}}(f) \quad (4.3)$$

Os conjuntos de níveis superiores e inferiores (veja um exemplo na Figura 4.5) são representações equivalentes de uma imagem f , no sentido que ela pode ser reconstruída a partir deles. Dessa forma, podemos reconstruir uma imagem f a partir de uma família de conjuntos de níveis superiores e inferiores extraídos de f :

$$\forall p \in \mathcal{D}, f(p) = \max\{\lambda : p \in \mathcal{X}^\lambda(f), \lambda \in \mathbb{K}\} = \min\{\lambda - 1 : p \in \mathcal{X}_\lambda(f), \lambda \in \mathbb{K}\} \quad (4.4)$$

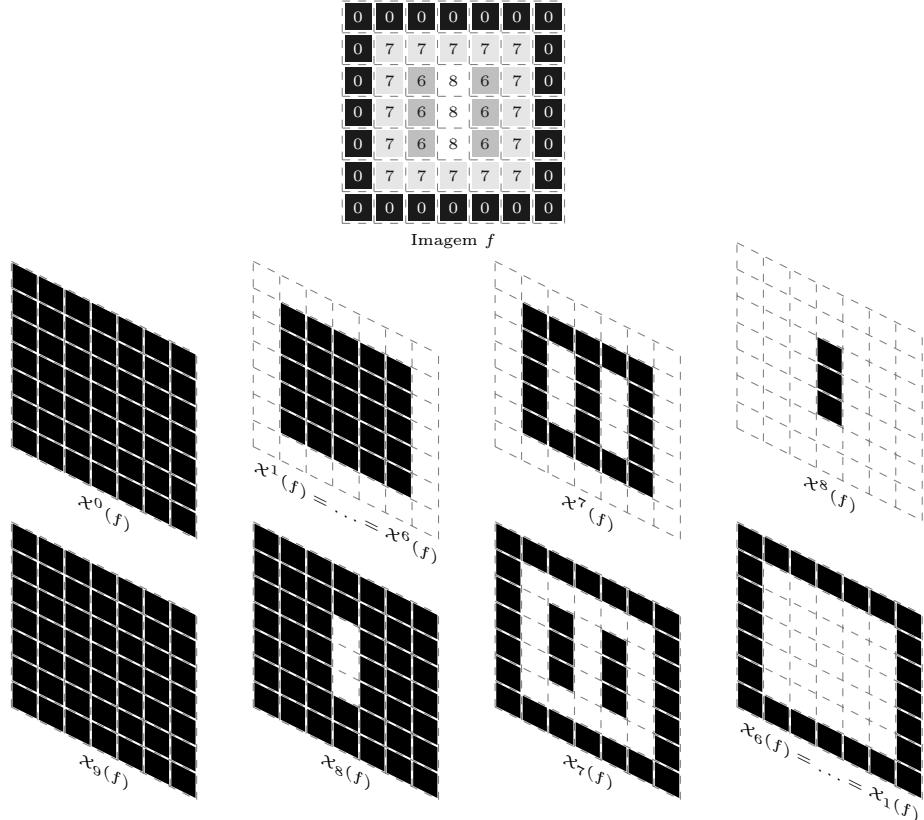


Figura 4.5: Conjuntos de níveis $\mathcal{X}^{\lambda\dots}(f)$ e $\mathcal{X}_{\lambda\dots}(f)$ para uma dada imagem f .

4.3.1 ÁRVORE DE COMPONENTES

A partir dos conjuntos de níveis superiores e inferiores é possível construir uma estrutura de dados hierárquica dos CCs extraídos de uma imagem. A essa estrutura é dado o nome de árvore de componentes. Essas árvores têm sido estudadas e utilizadas em diversas aplicações, tais como: filtragem e processamento de vídeo (Salembier, Oliveras e Garrido (1998)), extração de características com MSER (Matas et al. (2004)), extração de resíduos (Alves e Hashimoto (2014), Alves, Hashimoto e Marcotegui (2017)), filtragem por atributos (Breen e Jones (1996)), filtragem conexa (Jones (1999)) entre outras.

Vamos considerar $\mathcal{U}(f)$ como a família dos CCs dos conjuntos de níveis superiores extraídos de f , isto é, $\mathcal{U}(f) = \{C : C \in CC(\mathcal{X}^\lambda(f), \mathcal{A}), \lambda \in \mathbb{K}\}$ e respectivamente $\mathcal{L}(f)$ a família dos CCs dos conjuntos de níveis inferiores extraídos de f , isto é, $\mathcal{L}(f) = \{C : C \in CC(\mathcal{X}_\lambda(f), \mathcal{A}), \lambda \in \mathbb{K}\}$. Note que, se $\lambda_1 \geq \lambda_2$ e $A \in CC(\mathcal{X}^{\lambda_1}(f), \mathcal{A})$, $B \in CC(\mathcal{X}^{\lambda_2}(f), \mathcal{A})$ então ou, $A \cap B = \emptyset$ ou $A \subseteq B$. De forma similar, se $A \in CC(\mathcal{X}_{\lambda_1}(f), \mathcal{A})$, $B \in CC(\mathcal{X}_{\lambda_2}(f), \mathcal{A})$ então ou, $A \cap B = \emptyset$ ou $B \subseteq A$, e isso mostra que, A e B são disjuntos ou estão aninhados. Assim, os conjuntos das famílias dos CCs de níveis superiores $\mathcal{U}(f)$ e de níveis inferiores $\mathcal{L}(f)$ e a relação de inclusão \subseteq definem a árvore de componentes dos conjuntos de níveis superiores $(\mathcal{U}(f), \subseteq)$ e níveis inferiores $(\mathcal{L}(f), \subseteq)$ (veja a Figura 4.6).

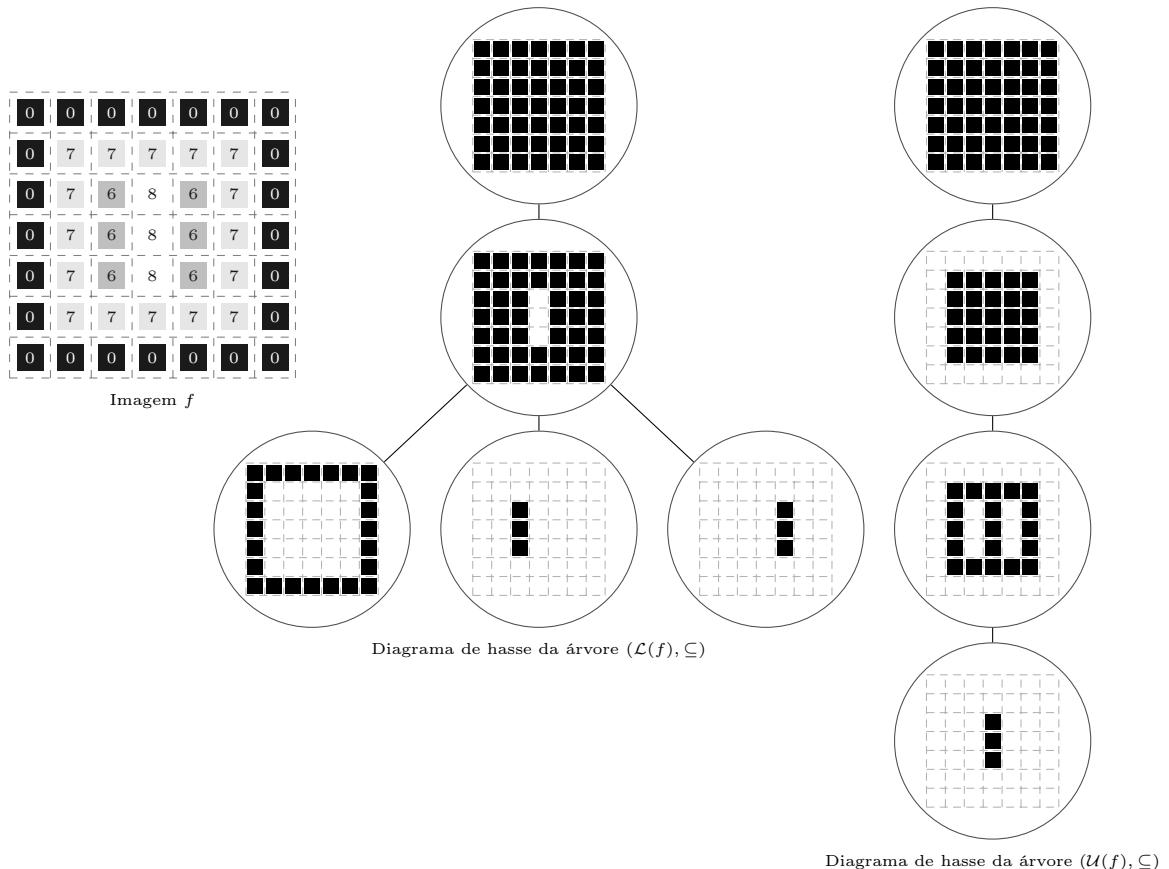


Figura 4.6: Exemplos de árvores de componentes para uma dada imagem f .

A árvore de componentes dos conjuntos de níveis superiores (respectivamente, inferiores) pode ser representada de forma não redundante por uma estrutura de dados chamada *max-tree* (respectivamente, *min-tree*). Dizemos que, essas estruturas de dados não são redundantes, pois um *pixel* $p \in \mathcal{D}$ está armazenado apenas no menor CC que o contém, e a relação de parentesco dada pelos vértices da árvore faz com que ele seja associado aos demais CCs que são seus ancestrais na árvore. Isso nos leva a definição do menor componente.

Definição 4.13 (Menor componente). Chama-se de menor componente de um *pixel* $p \in \mathcal{D}$ um vértice $A \in \mathcal{T}$ em que $p \in A$ e não existe descendentes de A em \mathcal{T} contendo p . Dessa forma, denotamos por $SC(\mathcal{T}, p)$ o menor CC que contém p em \mathcal{T} .

Similarmente, dizemos que $p \in \mathcal{D}$ é um *pixel* do nó compacto (CNP) de um dado componente $\tau \in \mathcal{T}$ se, e somente se, $\tau = SC(\mathcal{T}, p)$ e nesse caso nós denotamos por $\hat{\tau}$ o conjunto composto somente por CNPs de τ , isto é, $\hat{\tau} = \{p \in \mathcal{D} : \tau = SC(\mathcal{T}, p)\}$. Na Figura 4.7 são apresentados exemplos de árvores de componentes com os *pixels* e os respectivos CCs que os contêm.

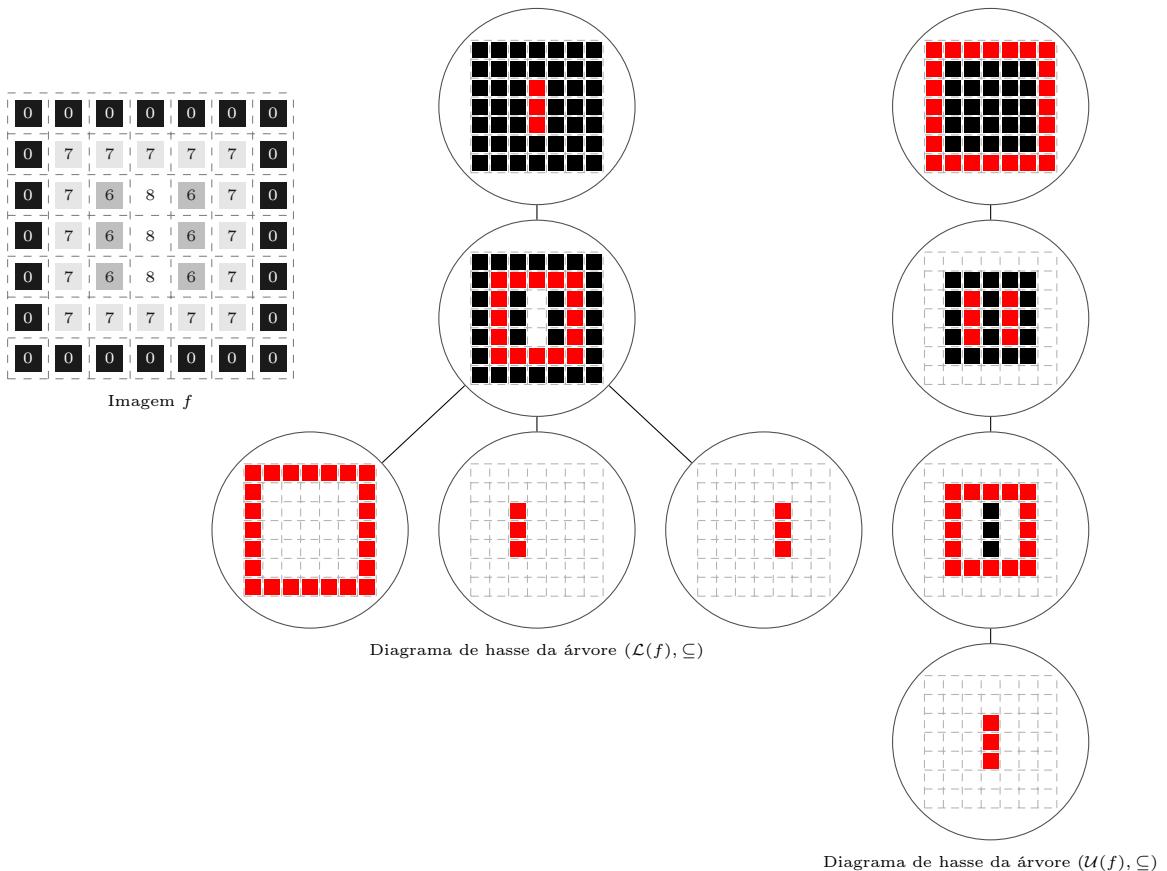


Figura 4.7: Exemplos de árvores de componentes para uma dada imagem f , em vermelho são mostrados os CNPs e os menores CCs que os contêm.

4.3.2 ÁRVORE DE FORMAS

As árvores de componentes são representações eficientes dos conjuntos de níveis superiores e inferiores. Uma outra representação considerada mais completa, porém, igualmente eficiente é a árvore de formas. A árvore de formas é uma importante estrutura de dados que representa uma imagem de maneira auto-dual. Ela pode ser obtida a partir da fusão das árvores de componentes, *max-tree* e *min-tree* com os buracos preenchidos e foi proposta por [Monasse e Guichard \(2000\)](#). Diversos trabalhos relacionados a árvores de formas podem ser encontrados na literatura com aplicações em: filtragem ([Caselles e Monasse \(2002\)](#)), extração de resíduos ([Alves e Hashimoto \(2014\)](#), [Alves, Hashimoto e Marcotegui \(2017\)](#)), funções de energia ([Ballester et al. \(2007\)](#), [Xu, Géraud e Najman \(2012\)](#), [Xu, Géraud e Najman \(2013\)](#), [Xu, Géraud e Najman \(2016\)](#)), entre outras.

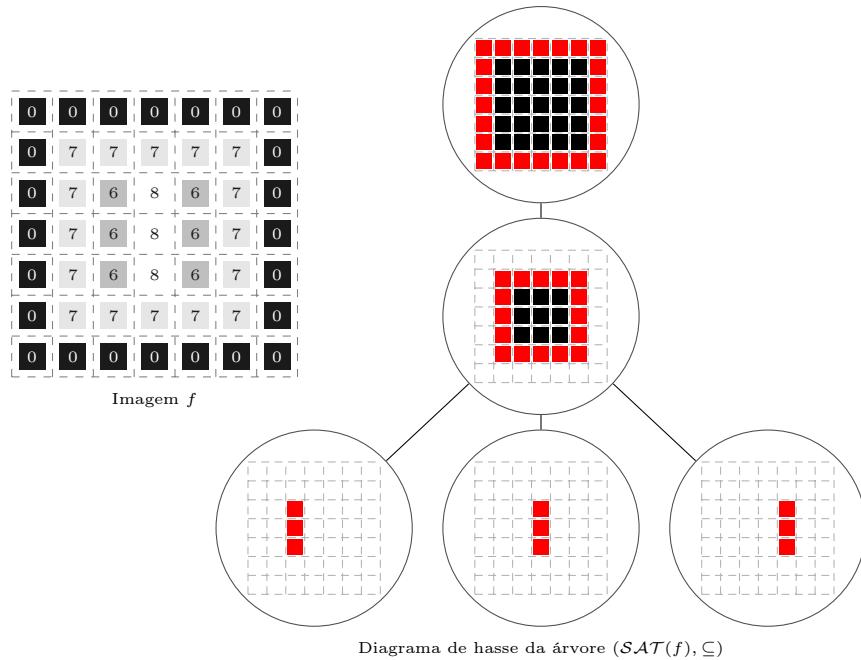


Figura 4.8: Exemplo de árvore de formas para uma dada imagem f , em vermelho são mostrados os CPNs e os menores CCs que os contêm.

Mais especificamente, a árvore de formas é obtida através da fusão entre as árvores de componentes pela noção de buracos internos dos CCs de $\mathcal{U}(f)$ e $\mathcal{L}(f)$, na Figura 4.8 é visto o resultado dessa fusão a partir dos buracos internos dos CCs da *max-tree* preenchidos por CCs da *min-tree* e da mesma forma, por buracos internos dos CCs da *min-tree* preenchidos por CCs da *max-tree*, ambas as árvores da Figura 4.6. Vale ressaltar que, só é possível obter esse resultado, se considerarmos conectividades duais para os CCs da *max-tree* e *min-tree*, como por exemplo, 4-conexos para os CCs da *min-tree*, isto é, $\mathcal{L}(f) = \{C : C \in CC(\mathcal{X}_\lambda, \mathcal{A}_4), \lambda \in \mathbb{K}\}$, e 8-conexos para os CCs da *max-tree*, ou seja, $\mathcal{U}(f) = \{C : C \in CC(\mathcal{X}^\lambda, \mathcal{A}_8), \lambda \in \mathbb{K}\}$. Segundo [Géraud et al. \(2013\)](#) a escolha de conectividades duais evita alguns problemas de topologia discreta. Para definirmos a

árvore de formas, vamos primeiramente considerar $sat : \mathcal{P}(\mathcal{D}) \rightarrow \mathcal{P}(\mathcal{D})$ como um operador que preenche buracos internos de um CC.

Definição 4.14 (Buracos internos). Denomina-se de buracos internos de $C \in \mathcal{L}(f) \cup \mathcal{U}(f)$, denotado por $Int(C)$, os CCs de $\mathcal{D} \setminus C$ que estão em $sat(C)$. A partir de $Int(C)$ definimos também o exterior de C como $Ext(C) = \mathcal{D} \setminus Int(C)$. Assim:

$$sat(C) = C \cup \bigcup_{H \in Int(C)} H.$$

Na Figura 4.9 é apresentado um exemplo de um componente conexo C com buracos internos e seus conjuntos $sat(C)$ e $Ext(C)$. Assim, com o operador de preenchimento de buracos internos, definimos $\mathcal{SAT}_{\mathcal{U}}(f) = \{sat(C) : C \in \mathcal{U}(f)\}$ e $\mathcal{SAT}_{\mathcal{L}}(f) = \{sat(C) : C \in \mathcal{L}(f)\}$ como as famílias de CCs dos conjuntos de níveis superiores e inferiores com os buracos internos preenchidos. A união destes conjuntos $\mathcal{SAT}(f) = \mathcal{SAT}_{\mathcal{U}}(f) \cup \mathcal{SAT}_{\mathcal{L}}(f)$ e a relação de inclusão definem a árvore de formas, ou seja, $(\mathcal{SAT}(f), \subseteq)$.

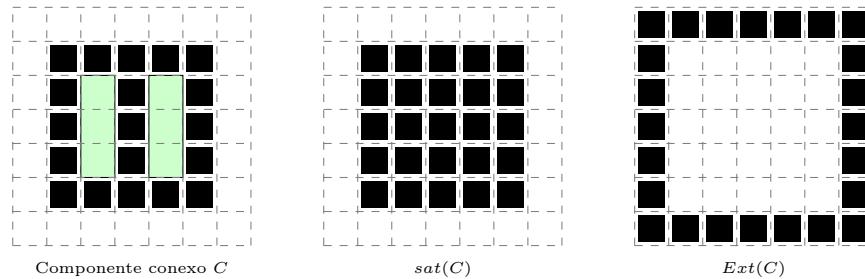


Figura 4.9: Um componente conexo C com buracos internos (destacados em verde) e seus conjuntos $sat(C)$ e $Ext(C)$.

A árvore de formas é uma estrutura de dados igualmente não redundante, assim como as árvores de componentes. Dessa forma, um pixel $p \in \mathcal{D}$ está armazenado somente no menor componente que o contém, isto é, $SC(\mathcal{SAT}(f), p)$. Conclui-se então que, a Definição 4.13 também é válida para a árvore de formas.

4.3.3 RECONSTRUÇÃO DE ÁRVORES PODADAS

A abordagem por representação de imagens em árvores tem como objetivo possibilitar a computação de algoritmos eficientes já conhecidos para grafos, e no final uma imagem é reconstruída a partir desta árvore. Dessa forma, os conjuntos de níveis são representações equivalentes de uma imagem f no sentido que ela pode ser reconstruída pela Equação 4.4. A partir dessa equação, podemos derivar funções $level_{\mathcal{L}}(C) : \mathcal{L}(C) \rightarrow \mathbb{K}$, $level_{\mathcal{U}}(C) : \mathcal{U}(C) \rightarrow \mathbb{K}$ e $level_{\mathcal{SAT}}(C) : \mathcal{SAT}(C) \rightarrow \mathbb{K}$ que mapeiam os CCs das árvores de componentes e de formas em níveis de cinza:

$$level_{\mathcal{L}}(C) = \{\lambda - 1 : C \in CC(\mathcal{X}_{\lambda}(f), \mathcal{A}), \lambda \in \mathbb{K}\},$$

$$\begin{aligned} level_{\mathcal{U}}(C) &= \{\lambda : C \in CC(\mathcal{X}^{\lambda}(f), \mathcal{A}), \lambda \in \mathbb{K}\} \text{ e} \\ level_{\mathcal{SAT}}(C) &= f(y) \text{ tal que } y \in \arg \max \{|SC(\mathcal{T}_f, x)| : x \in C\}. \end{aligned}$$

onde \mathcal{T}_f é uma árvore morfológica construída a partir de f . Vale ressaltar que, a partir de agora por simplicidade de notação será adotada a nomenclatura *level* para todas estas funções. Dessa forma, a partir de \mathcal{T}_f podemos reconstruir f da seguinte forma:

$$\forall p \in \mathcal{D}, f(p) = level(SC(\mathcal{T}_f, p)). \quad (4.5)$$

Nesta dissertação, essa operação é escrita como $f = Rec(\mathcal{T}_f)$. Quando se tratar de $Rec((\mathcal{U}(f), \subseteq))$ (respectivamente, $Rec((\mathcal{L}(f), \subseteq))$ e $Rec((\mathcal{SAT}(f), \subseteq))$) será dita reconstução superior (respectivamente, inferior e por formas). Conforme já foi definido, árvores podem ser obtidas por operações de poda. Nesse sentido, dizemos que \mathcal{T}_g é obtida por uma poda em \mathcal{T}_f se, e somente se, $\mathcal{T}_g = Poda(\mathcal{T}_f)$ e $g = Rec(\mathcal{T}_g)$. Um exemplo ilustrando esta abordagem é apresentado na Figura 4.10.

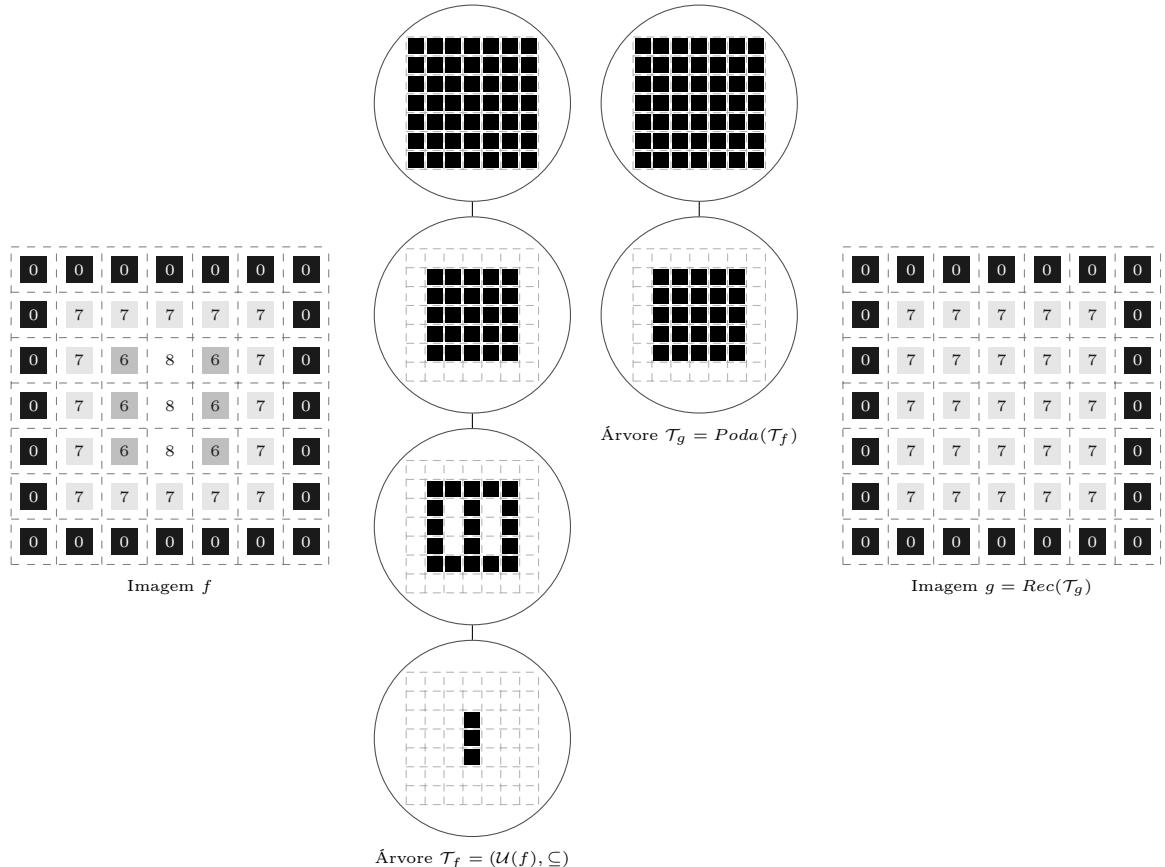


Figura 4.10: Exemplo de uma poda e uma reconstrução.

4.4 ALGORITMOS PARA CONSTRUÇÃO DE ÁRVORES MORFOLÓGICAS

Na literatura são apresentados diversos algoritmos para construção de árvores morfológicas. De acordo com Carlinet e Géraud (2014), no que diz respeito a construção de árvores de componentes podemos dividir os algoritmos encontrados na literatura em três classes: (i) *flooding*: nesse algoritmo a árvore é construída da raiz para as folhas (SALEMBIER; OLIVERAS; GARRIDO, 1998; NISTÉR; STEWÉNIUS, 2008; WILKINSON et al., 2011); (ii) *merge*: essa abordagem consiste em computar a árvore a partir de sub-partes da imagem e então juntá-las em um processo de *merging* para formar a árvore de componentes (MATAS et al., 2008; WILKINSON et al., 2008) e (iii) *union-find*: nesse algoritmo proposto por Tarjan (1975) a computação da árvore é feita das folhas para a raiz (NAJMAN; COUPRIE, 2006; BERGER et al., 2007). Uma implementação de *union-find* é apresentada no Algoritmo 1.

Algoritmo 1: Algoritmo de *union-find*.

\mathcal{S}	vetor de <i>pixels</i> ordenados	q, p, r	<i>pixels</i>
<i>zpar</i>	estrutura temporária de parentesco	<i>parent</i>	estrutura de parentesco

```

função find-root(zpar, p)
    se zpar(p)  $\neq p$  então
        |   zpar(p)  $\leftarrow$  find-root(zpar, zpar(p))
    fim
    retorna zpar(p)
fim
função union-find( $\mathcal{S}$ )
    para todo i  $\in \mathcal{S}$  faça
        |   p  $\leftarrow \mathcal{S}[i]
        |   parent(p)  $\leftarrow -1$ 
    fim
    para todo i  $\in \mathcal{S}$  em ordem decrescente faça
        |   p  $\leftarrow \mathcal{S}[i]
        |   parent(p)  $\leftarrow zpar(p) \leftarrow p$ 
        |   para todo q  $\in \mathcal{A}(p)$  tal que parent(p)  $\neq -1$  faça
            |       r  $\leftarrow$  find-root(zpar, q)
            |       se r  $\neq p$  então
            |           |   parent(r)  $\leftarrow zpar(r) \leftarrow p$ 
            |       fim
        |   fim
    |   fim
    retorna parent
fim$$ 
```

Analizando o trabalho de Carlinet e Géraud (2014), na maioria dos casos o algoritmo baseado em *union-find* apresentado por Berger et al. (2007) é o mais indicado para a

construção de árvores de componentes. Vale ressaltar que, a respeito da complexidade de memória o Algoritmo 1 conta com duas estruturas $zpar$, $parent$ e um vetor $\mathcal{S} : \mathbb{N} \rightarrow \mathcal{D}$ com os *pixels* ordenados, assim temos um gasto de memória de $\mathcal{O}(3|\mathcal{D}|)$, já a complexidade de tempo do pior caso é $\mathcal{O}(\log(|\mathcal{D}|) \times |\mathcal{D}|)$. Assim, o Algoritmo 2 apresenta uma implementação do algoritmo para construção de árvores de componentes baseado em *union-find* de Berger et al. (2007). Um exemplo de construção de uma *min-tree* utilizando o Algoritmo 2 é apresentado na Figura 4.11.

Algoritmo 2: Algoritmo para construção de árvores de componentes de Berger et al. (2007).

\mathcal{S}	vetor de <i>pixels</i> ordenados	q, p	<i>pixels</i>
f	imagem de entrada	<i>parent</i>	estrutura de parentesco

```

função canonize( $f, \mathcal{S}, parent$ )
  para todo  $i \in \mathcal{S}$  em ordem decrescente faça
     $p \leftarrow \mathcal{S}[i]$ 
     $q \leftarrow parent(p)$ 
    se  $f(parent(q)) = f(q)$  então
      |  $parent(p) \leftarrow parent(q)$ 
    fim
  fim
  retorna parent
fim

função computa-arvore-de-componentes( $f$ )
   $\mathcal{S} \leftarrow \text{ordena}(f)$ 
   $parent \leftarrow \text{union-find}(\mathcal{S})$ 
   $parent \leftarrow \text{canonize}(f, \mathcal{S}, parent)$ 
  retorna parent
fim

```

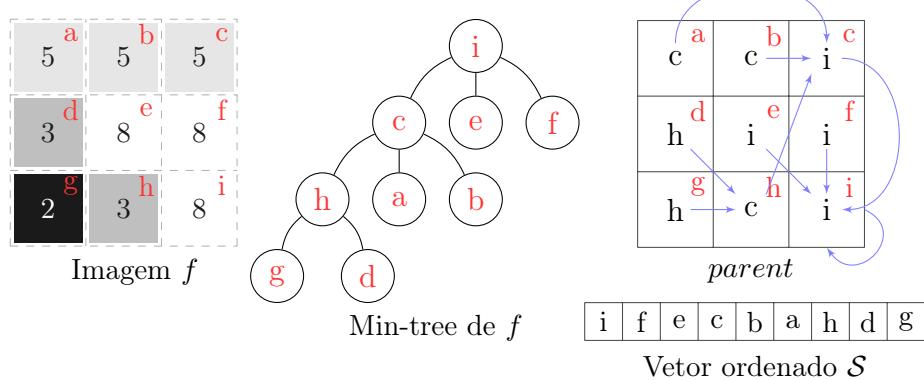


Figura 4.11: Construção de uma *min-tree* utilizando o Algoritmo 2.

A estrutura $parent$ obtida pelo Algoritmo 2 armazena a relação de parentesco entre os CCs da árvore \mathcal{T} a partir dos *pixels* que são chamados de *canônicos*. Cada um desses *pixels* representa um CC da árvore \mathcal{T} , assim, seu nível de cinza pode ser utilizado para

obter o menor conjunto de nível que contém o CC. Dessa forma, é possível construir uma estrutura mais simples para representar a árvore de componentes, porém ainda não redundante. Esses fatos podem ser visualizados na Figura 4.12.

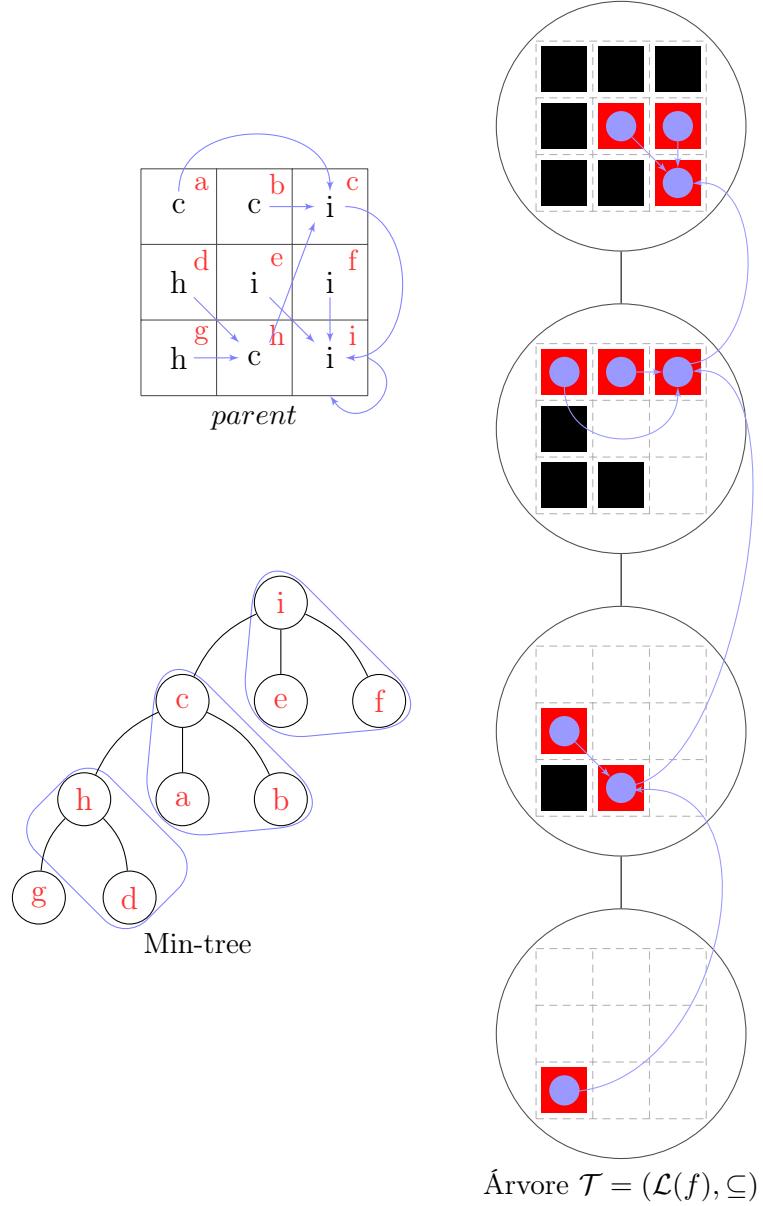


Figura 4.12: Codificação de uma min-tree \mathcal{T} contida na estrutura parent.

Os algoritmos baseados em *union-find* precisam que os *pixels* estejam ordenados pelos seus níveis de cinza (em ordem crescente para a *min-tree* e ordem decrescente para a *max-tree*). Nesse sentido, como a imagem em geral tem baixa quantização e os níveis de cinza são valores discretos, um algoritmo de ordenação eficiente que pode ser utilizado é o *bucket-sort*. Assim, optamos por utilizar uma implementação do *bucket sort* em fila de prioridade proposta por [Falcao, Udupa e Miyazawa \(2000\)](#) que realiza as operações de consulta, inserção e remoção em complexidade $\mathcal{O}(1)$. A respeito da complexidade de memória é necessário um vetor com C *buckets*, e para cada *bucket* é alocado uma lista

duplamente ligada que são preenchidas pelos *pixels* consumindo $|\mathcal{D}|$ posições, assim o gasto total de memória da fila de prioridade é $\mathcal{O}(|\mathcal{D}| + C)$. Os *pixels* são inseridos em política LIFO (*last in first out*) e removidos pela ordem de prioridade da fila, considerando que uma zona plana da imagem é processada por vez. Assim, a complexidade de tempo é $\mathcal{O}(|\mathcal{D}| \times |\mathcal{A}|)$. A implementação desse algoritmo é apresentada no Algoritmo 3.

Algoritmo 3: Algoritmo de ordenação baseado na fila de prioridade proposta por Falcao, Udupa e Miyazawa (2000)

\mathcal{S}	vetor de <i>pixels</i> ordenados	prioridade	mínima ou máxima
f	imagem de entrada	q, p	<i>pixels</i>
Q	fila	i	contador

```

função ordena( $f$ )
    inicia( $Q, C$ )
    para todo  $p \in \mathcal{D}$  faça
        | enqueue( $Q, p, f(p)$ )
    fim
     $i \leftarrow 0$ 
    enquanto  $Q$  não está vazia faça
        |  $p \leftarrow \text{dequeue}(Q, \text{prioridade})$ 
        |  $\mathcal{S}[i] = p$ 
        | para todo  $p \in \mathcal{A}(p)$  faça
            |   | se  $f(p) = f(q)$  e contem( $Q, q$ ) então
            |   |   | remove( $Q, q$ )
            |   |   | enqueue( $Q, q, f(q)$ )
            |   | fim
        | fim
        |  $i \leftarrow i + 1$ 
    fim
    retorna  $\mathcal{S}$ 
fim

```

Para a construção da árvore de formas o primeiro algoritmo surgiu com o trabalho de Monasse e Guichard (2000). Posteriormente, Song (2007) apresentou uma abordagem *top-down* para a construção da árvore de formas. Contudo, essas implementações são onerosas e a complexidade de ambas é $\mathcal{O}(|\mathcal{D}|^2)$. Recentemente, Géraud et al. (2013) propuseram um algoritmo para computar a árvore de formas em complexidade quase linear que é baseado no algoritmo de construção de árvores de componentes proposto por Berger et al. (2007). Esse algoritmo necessita de uma outra representação da imagem chamada Grade Khalimsky onde os *pixels* são interpolados através de faces (0, 1 e 2). Além disso, para construir a árvore de formas as faces devem ser ordenadas através de um algoritmo que permita passar entre os *pixels* da Grade (GÉRAUD et al., 2013). Mais detalhes sobre estas teorias podem ser encontrados em Najman e Géraud (2013). Um exemplo de implementação para interpolação simples é apresentado no Algoritmo 4.

Algoritmo 4: Algoritmo que interpola a imagem em uma Grade Khalimsky simples.

F	conjunto de <i>pixels</i> interpolados	i	índice das faces da Grade
W	largura da imagem	H	altura da imagem
$lower$	variável de mínimo	$upper$	variável de máximo
m	mediana de f	p, q	<i>pixels</i>

```

função interpola-simples( $f$ )
     $m \leftarrow median(a)$ 
    para todo  $p \in \mathcal{D}$  faça
         $h \leftarrow (2 \times p_y + 1) \times (W \times 2 + 1) + (2 \times p_x + 1)$ 
         $F(h) \leftarrow \{f(p), f(p)\}$ 
    fim
    para todo  $h \in \{(0, 0), (0, 1), \dots, (W \times 2 + 1, H \times 2 + 1)\}$  faça
         $\mathcal{A} \leftarrow \{\}$ 
         $lower \leftarrow +\infty$ 
         $upper \leftarrow -\infty$ 
        se ( $h_x \ mod \ 2 = 0$ ) e ( $h_y \ mod \ 2 = 0$ ) então
             $\mathcal{A} \leftarrow \{(-1, -1), (+1, -1), (-1, +1), (+1, +1)\}$ 
        senão se ( $h_x \ mod \ 2 = 0$ ) e ( $h_y \ mod \ 2 = 1$ ) então
             $\mathcal{A} \leftarrow \{(+1, 0), (-1, 0)\}$ 
        senão se ( $h_x \ mod \ 2 = 1$ ) e ( $h_y \ mod \ 2 = 0$ ) então
             $\mathcal{A} \leftarrow \{(0, -1), (0, 1)\}$ 
        fim
        para todo  $i \in \mathcal{A}$  faça
             $q \leftarrow (h_x + i_x, h_y + i_y)$ 
            se  $q$  não está fora da borda então
                 $upper \leftarrow \max\{F(q), upper\}$ 
                 $lower \leftarrow \min\{F(q), lower\}$ 
            senão
                 $upper \leftarrow \max\{m, upper\}$ 
                 $lower \leftarrow \min\{m, lower\}$ 
            fim
        fim
        se  $\mathcal{A} \neq \{\}$  então
             $F(h) = \{lower, upper\}$ 
        fim
    fim
    retorna  $(F, m)$ 
fim

```

A implementação para interpolação da Grade Khalimsky simples apresentada no Algoritmo 4 percorre 4 vezes o tamanho de \mathcal{D} , assim a complexidade de tempo é $\mathcal{O}(4|\mathcal{D}|)$. O Algoritmo 6 apresenta uma implementação para construção da árvore de formas baseado na Grade Khalimsky Simples, note que, após a interpolação e a ordenação (veja o Algoritmo 5), aplicam-se as funções union-find e canonize do algoritmo de Berger et al. (2007).

A respeito da complexidade de tempo, considerando o algoritmo de Berger et al. (2007) e a Grade Khalimsky Simples o Algoritmo 6 consome no pior caso $\mathcal{O}(\log(4|\mathcal{D}|) \times 4|\mathcal{D}|)$.

Algoritmo 5: Algoritmo de ordenação da Grade Khalimsky simples para construção da árvore de formas apresentado em Géraud et al. (2013).

F	conjunto de <i>pixels</i> interpolados	f_S	imagem das faces ordenadas
\mathcal{Q}	vetor de filas de tamanho $ \mathbb{K} $	m	mediana de f

```

função priority-enqueue( $\mathcal{Q}, h, F, l$ )
     $\{lower, upper\} \leftarrow F(h)$ 
    se  $lower > l$  então
        |  $l' \leftarrow lower$ 
    senão se  $upper < l$  então
        |  $l' \leftarrow upper$ 
    senão
        |  $l' \leftarrow l$ 
    fim
    enqueue( $\mathcal{Q}[l'], h$ )
fim
função priority-dequeue( $\mathcal{Q}, l$ )
    se  $\mathcal{Q}[l]$  está vazia então
        |  $l' \leftarrow$  próximo nível que  $\mathcal{Q}[l'] \neq \emptyset$ 
        |  $l \leftarrow l'$ 
    fim
    retorna dequeue( $(\mathcal{Q}[l]), l$ )
fim
função ordena-faces( $F, m$ )
    para todo  $h \in F$  faça
        |  $deja\_vu[h] \leftarrow false$ 
    fim
     $i \leftarrow 0$ 
    enqueue( $\mathcal{Q}[m], i$ )
     $deja\_vu[i] \leftarrow true$ 
     $l \leftarrow m$ 
    enquanto  $\mathcal{Q}$  não está vazia faça
        |  $(h, l) \leftarrow$  priority-dequeue( $\mathcal{Q}, l$ )
        |  $f_S(h) \leftarrow l$ 
        |  $S[i] \leftarrow h$ 
        | para todo  $q \in \mathcal{A}_4(h)$  tal que  $deja\_vu[q] = false$  faça
            |     | priority-enqueue( $\mathcal{Q}, q, F, l$ )
            |     |  $deja\_vu[q] \leftarrow true$ 
        | fim
        |  $i \leftarrow i + 1$ 
    fim
    retorna ( $S, f_S$ )
fim

```

A construção da árvore de formas via *union-find* garante na estrutura *parent* a mesma

relação de parentesco estabelecida nas árvores de componentes, isto é, em $parent$ está codificada a árvore de formas \mathcal{T} . Um exemplo de construção da árvore de formas utilizando o Algoritmo 6 é apresentado na Figura 4.13.

Algoritmo 6: Algoritmo que computa uma árvore de formas a partir de uma Grade Khalimsky simples.

F	conjunto de <i>pixels</i> interpolados	f_S	imagem das faces ordenadas
S	vetor de <i>pixels</i> das faces ordenados	$parent$	estrutura de parentesco

função computa-arvore-de-formas(f)

```
( $F, m$ )  $\leftarrow$  interpola-simples( $f$ )
( $S, f_S$ )  $\leftarrow$  ordena-faces( $F, m$ )
parent  $\leftarrow$  union-find( $S$ )
parent  $\leftarrow$  canonize( $f_S, S, parent$ )
retorna parent
```

fim

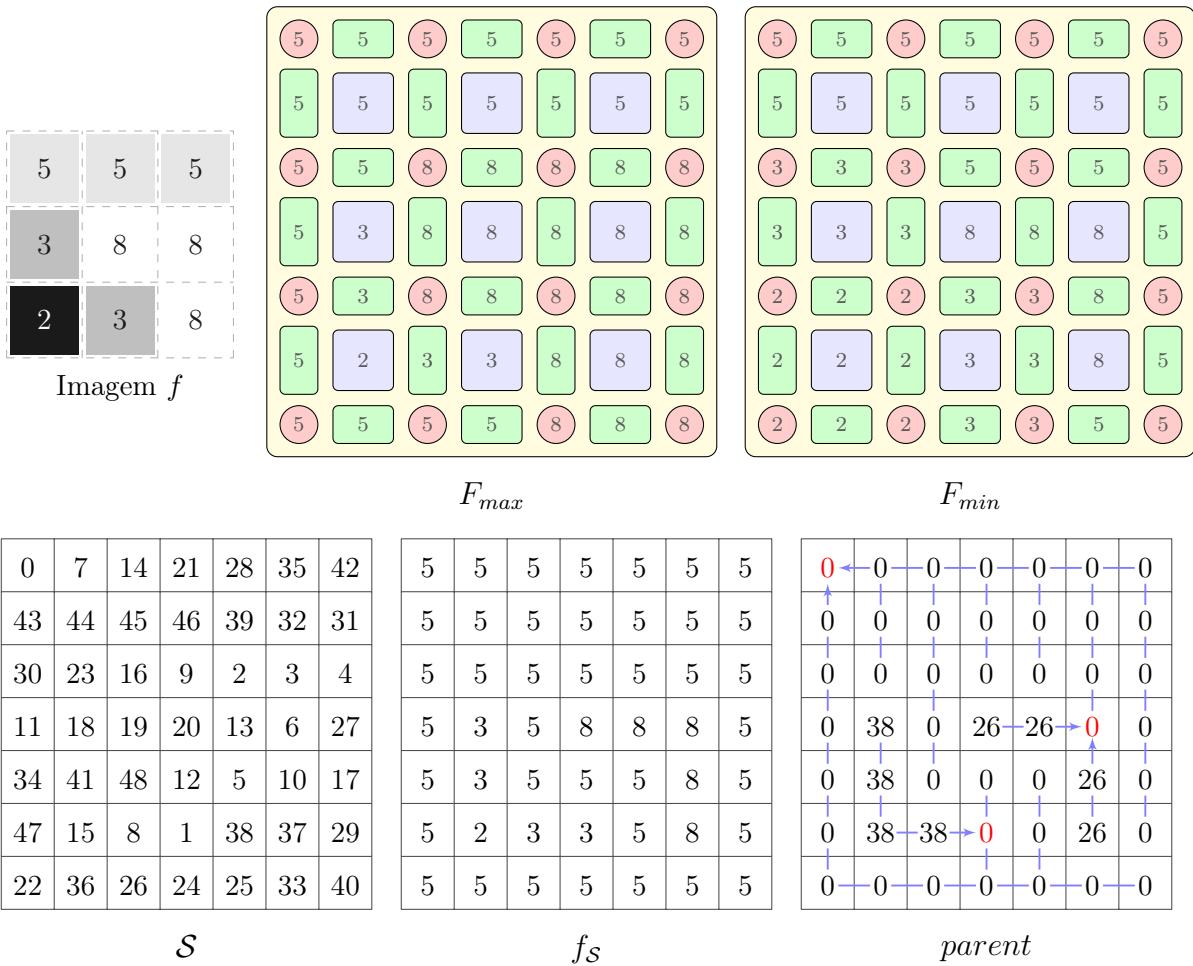


Figura 4.13: Visualização do Algoritmo 6 para construção da árvore de formas baseado em uma Grade Khalimsky simples. Note que, $F = \{F_{min}, F_{max}\}$ e os círculos vermelhos são a face 0, os retângulos verdes a face 1 e os quadrados em azul a face 2 e na estrutura parent os representantes estão em vermelho.

ÚLTIMOS LEVELINGS BASEADOS EM FUNÇÕES DE ENERGIA

Resumo do capítulo

Os últimos levelings são operadores robustos que analisam um espaço de escalas baseado em levelings por meio de diferenças consecutivas (resíduos) e consideram os máximos resíduos. Em adição, esses resíduos revelam informações importantes sobre o contraste de uma imagem. Assim, são apresentados nesse capítulo conceitos sobre espaço de escalas baseado em levelings e a sua relação com árvores morfológicas, bem como a extração residual que da origem aos últimos levelings. Nesse contexto, é também apresentada uma revisão literária a respeito de funções de energia, que culmina na construção de estratégias para classificar resíduos dos últimos levelings com base em funções de energia. Muitos conceitos sobre últimos levelings mostrados aqui são encontrados em Alves (2013) e Alves, Hashimoto e Marcotegui (2017).

5.1 ESPAÇO DE ESCALAS BASEADO EM LEVELINGS OBTIDO POR PODAS SUCESSIVAS EM ÁRVORES MORFOLÓGICAS

Os *levelings* são robustos filtros de simplificação que não criam novas estruturas na imagem (extremas regionais e contornos). Conforme introduzido anteriormente, estamos interessados em construir um espaço de escalas baseado em *levelings* e, uma forma de obter esse espaço eficientemente é através de árvores morfológicas. Mais especificamente, seja \mathcal{T}_f uma árvore morfológica e seja $(\mathcal{T}_f^0, \mathcal{T}_f^1, \mathcal{T}_f^2, \dots)$ uma sequência de árvores morfológicas obtida a partir de podas sucessivas, isto é:

$$\mathcal{T}_f^i = \begin{cases} \mathcal{T}_f, & \text{se } i = 0 \\ Poda(\mathcal{T}_f^{i-1}), & \text{se } i > 0 \end{cases} \quad \text{para } i = 0, 1, \dots \quad (5.1)$$

Conforme mostrado por Alves, Morimitsu e Hashimoto (2015) reconstruções de árvores podadas são *levelings* (veja a Figura 5.1). Além disso, conforme mostrado por Meyer e Maragos (2000) os *levelings* podem ser aninhados para criar um espaço de escalas de uma imagem. Assim, a sequência de reconstruções de podas sucessivas:

$$(\psi_0(f), \psi_1(f), \dots, \psi_{I_{MAX}}(f)), \text{ com } \psi_i(f) = Rec(\mathcal{T}_f^i), \text{ para } i \in \mathcal{I} = \{0, 1, \dots, I_{MAX}\} \quad (5.2)$$

constitui um espaço de escalas morfológico baseado em *levelings* e dessa forma somos levados ao Teorema 5.1.

Teorema 5.1 (Alves, Morimitsu e Hashimoto (2015)). *Seja \mathcal{T}_f uma árvore morfológica que representa uma imagem f . Seja $(\mathcal{T}_f^0, \mathcal{T}_f^1, \dots, \mathcal{T}_f^n)$ uma sequência de n árvores morfoló-*

gicas obtida por podas sucessivas. Então, a sequência de reconstruções $(\psi_0, \psi_1, \dots, \psi_{I_{MAX}})$ é um espaço de escalas morfológico baseado em levelings.

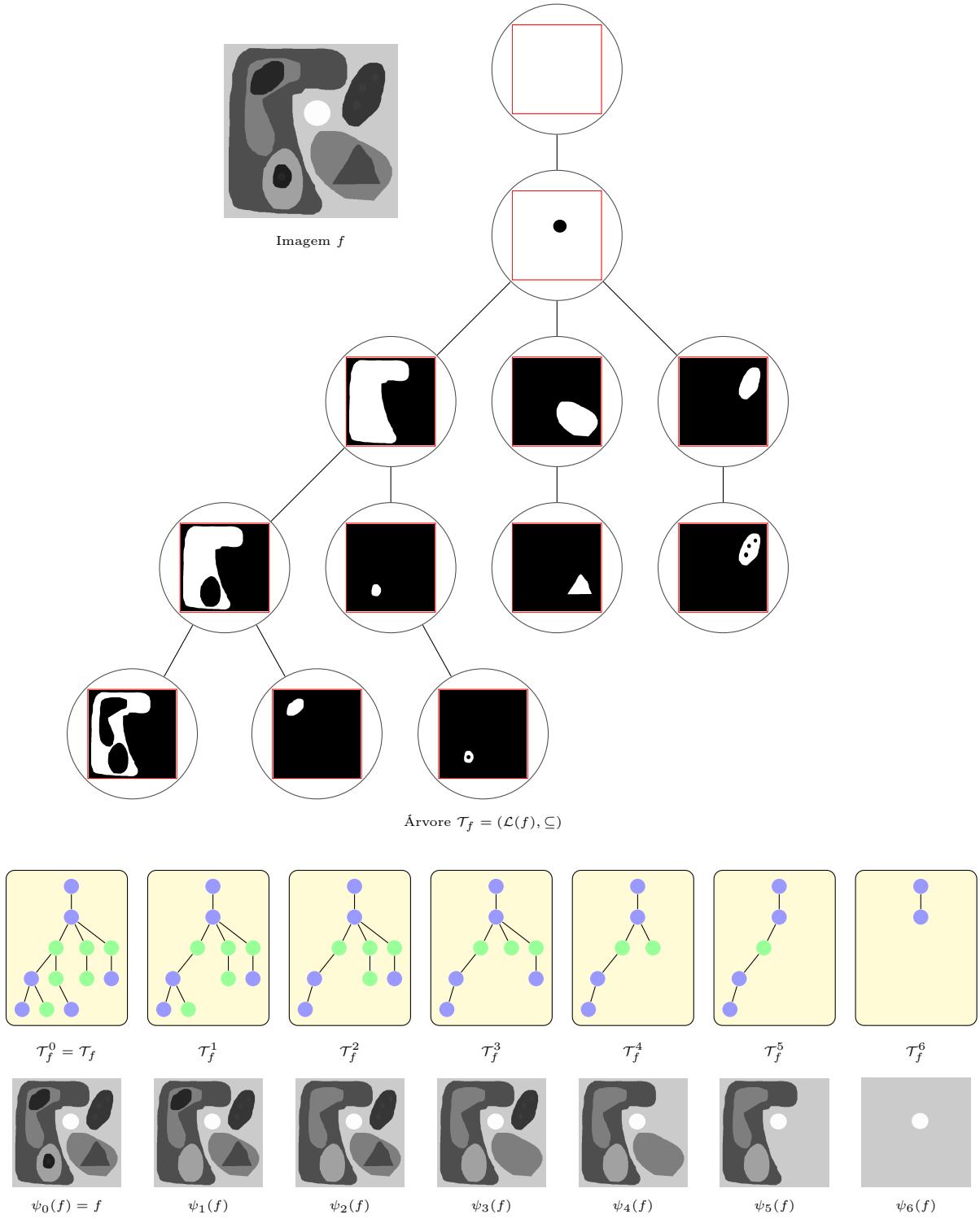


Figura 5.1: Espaço de escalas baseado em levelings obtido a partir de podas sucessivas em uma min-tree. Os nós representados em verdes são os marcados para serem podados.

Fonte : Alves (2013)(Adaptado pelo autor.)

5.2 EXTRAÇÃO RESIDUAL EM UM ESPAÇO DE ESCALAS BASEADO EM LEVELINGS

Em grande parte dos problemas de visão computacional detectar objetos de interesse não é uma tarefa fácil. Nesse sentido, algumas vezes é interessante remover o conteúdo desejável da imagem, porém reconstruí-lo por meio de diferenças com a imagem original, nesse contexto temos os operadores residuais. Um operador residual r é definido como a diferença (*pixel a pixel*) entre dois operadores, digamos ψ e ϕ , aplicados em uma dada imagem $f \in \mathcal{F}(\mathcal{D})$, isto é, $\forall p \in \mathcal{D}, [r(f)](p) = [\psi(f)](p) - [\phi(f)](p)$. Nesse caso, os operadores são chamados de *primitivas*, e os resíduos de um *pixel* p são classificados em três tipos: *resíduo positivo*, se $[\psi(f)](p) > [\phi(f)](p)$; *resíduo negativo*, se $[\psi(f)](p) < [\phi(f)](p)$; e *resíduo nulo* caso contrário.

Operadores residuais têm sido estendidos para extraer resíduos de famílias de primitivas. Alguns deles são definidos sobre uma única família de primitivas $\{\psi_i : i \in \mathcal{I}_{\preccurlyeq}\}$ indexadas por um conjunto $\mathcal{I}_{\preccurlyeq} = \{1, 2, \dots, \mathcal{I}_{MAX}\}$, tal que $\psi_i \preccurlyeq \psi_{i+1}$, onde \preccurlyeq indica uma relação de ordem parcial sobre a família de primitivas. Assim, para uma dada imagem $f \in \mathcal{F}(\mathcal{D})$ um operador residual desta família é definido como o supremo dos resíduos consecutivos extraídos sobre ela, isto é:

$$\forall p \in \mathcal{D}, [\theta(f)](p) = \sup_{i \in \mathcal{I}_{\preccurlyeq}} \{[r_i(f)](p) : [\psi_i(f)](p) - [\psi_{i+1}(f)](p)\}. \quad (5.3)$$

Baseado nessa forma de extrair resíduos, Beucher (2007) propôs o operador residual *ultimate opening* (e por dualidade, *ultimate closing*), onde as primitivas são aberturas por EEs de tamanhos crescentes $\{\gamma_{\mathcal{B}_i} : i \in \mathcal{I}_{\geq}\}$ (respectivamente, fechamentos por EEs de tamanhos decrescentes $\{\varphi_{\mathcal{B}_i} : i \in \mathcal{I}_{\leq}\}$). Mais tarde, Marcotegui, Hernández e Retornaz (2011) propuseram os operadores residuais *ultimate attribute opening* (UAO) e por dualidade *ultimate attribute closing* (UAC), onde as primitivas são respectivamente: aberturas por atributos $\{\gamma_i^k : i \in \mathcal{I}_{\geq}\}$ e fechamentos por atributos $\{\varphi_i^k : i \in \mathcal{I}_{\leq}\}$.

Sob outra perspectiva, Alves e Hashimoto (2014) propuseram os *ultimate grain filters* (UGF) para extrair valores residuais de famílias de *grain filters* $\{\phi_i : i \in \mathcal{I}\}$ indexadas pelo conjunto $\mathcal{I} = \{1, 2, \dots, \mathcal{I}_{MAX}\}$. Esses operadores produzem resíduos positivos e negativos, enquanto os operadores definidos por Marcotegui, Hernández e Retornaz (2011) e Beucher (2007) não lidam com essa simultaneidade, isto é, ou produzem resíduos positivos ou negativos. Embora existam algumas diferenças, os operadores expostos possuem uma propriedade em comum: as famílias de primitivas que foram usadas para produzi-los é composta por *levelings* consecutivos. Assim, esses operadores pertencem a uma classe denominada de últimos *levelings*.

Definição 5.1 (Últimos *levelings* (ALVES; HASHIMOTO; MARCOTEGUI, 2017)). Um operador *último leveling positivo* \mathcal{R}_θ^+ (respectivamente, negativo \mathcal{R}_θ^-) é o supremo dos resíduos positivos r_i^+ (respectivamente, negativos r_i^-) extraídos de uma família $\{\psi_i : i \in \mathcal{I}\}$

indexada pelo conjunto de índices $\mathcal{I} = \{0, 1, \dots, \mathcal{I}_{MAX}\}$, tal que para quaisquer $i, j \in \mathcal{I}$, se $i \leq j$ então ψ_j é *leveling* de ψ_i . Assim para uma dada imagem $f \in \mathcal{F}(\mathcal{D})$, o operador *último leveling* \mathcal{R}_θ é o supremo entre \mathcal{R}_θ^+ e \mathcal{R}_θ^- , isto é, $\forall p \in \mathcal{D}$:

$$\begin{aligned} [\mathcal{R}_\theta^+(f)](p) &= \sup_{i \in \mathcal{I}} \{[r_i^+(f)](p) : [r_i^+(f)](p) = [\psi_i(f)](p) - [\psi_{i+1}(f)](p) \vee 0\}, \\ [\mathcal{R}_\theta^-(f)](p) &= \sup_{i \in \mathcal{I}} \{[r_i^-(f)](p) : [r_i^-(f)](p) = [\psi_{i+1}(f)](p) - [\psi_i(f)](p) \vee 0\}, \\ [\mathcal{R}_\theta(f)](p) &= [\mathcal{R}_\theta^+(f)](p) \vee [\mathcal{R}_\theta^-(f)](p). \end{aligned}$$

Na Figura 5.2 são apresentados exemplos de operadores únicos *levelings*, respectivamente UAO, UAC e UGF.

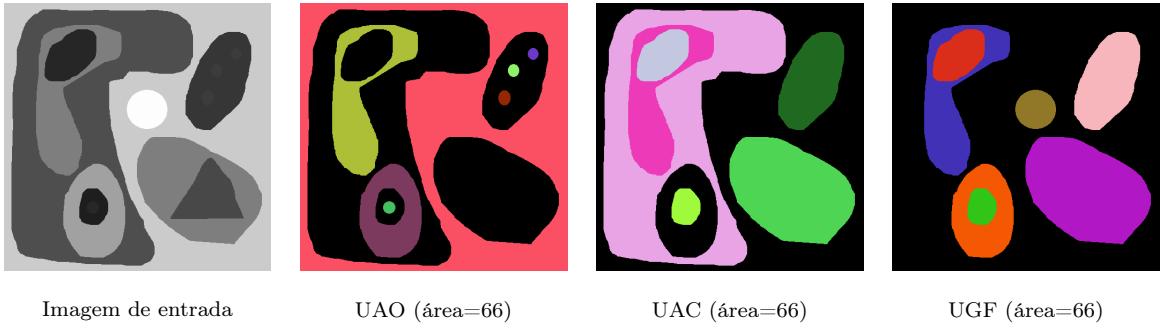


Figura 5.2: Exemplos de aplicações de operadores únicos *levelings*. As imagens foram rotuladas para facilitar a visualização dos resíduos. Da esquerda para direita, UAO e UAC (MARCOTE-GUI; HERNÁNDEZ; RETORNAZ, 2011) e UGF (ALVES; HASHIMOTO, 2014).

Conforme mostrado na seção anterior, podemos construir um espaço de escalas baseado em *levelings* por operações de poda e reconstruções em árvores morfológicas. Dessa forma, para duas árvores do espaço de escalas baseado em *levelings*, digamos \mathcal{T}_f^i e \mathcal{T}_f^{i+1} podemos utilizar os nós em \mathcal{T}_f^i que não estão presentes em \mathcal{T}_f^{i+1} para calcular os valores residuais entre $\psi_i(f)$ e $\psi_{i+1}(f)$, ou seja, basta analisar os *pixels* presentes nos nós do conjunto $\mathcal{Nr}(i) = \mathcal{T}_f^i \setminus \mathcal{T}_f^{i+1}$. Assim, o i -ésimo resíduo positivo (respectivamente, negativo) $r_{\mathcal{T}_f^i}^+(\tau)$ (respectivamente, $r_{\mathcal{T}_f^i}^-(\tau)$) pode ser obtido da seguinte maneira:

$$\forall \tau \in \mathcal{Nr}(i), r_{\mathcal{T}_f^i}^+(\tau) = \begin{cases} \text{level}(\tau) - \text{level}(\text{Pai}(\tau)), & \text{se } \text{Pai}(\tau) \notin \mathcal{Nr}(i), \\ \text{level}(\tau) - \text{level}(\text{Pai}(\tau)) \\ + r_{\mathcal{T}_f^i}^+(\text{Pai}(\tau)), & \text{caso contrário,} \end{cases} \quad (5.4)$$

então, com auxílio da recorrência acima o i -ésimo resíduo positivo (respectivamente, negativo) $r_i^+(f)$ (respectivamente, $r_i^-(f)$) é dado pela seguinte recorrência:

$$\forall p \in \mathcal{D}, [r_i^+(f)](p) = \begin{cases} r_{\mathcal{T}_f^i}^+(\mathcal{SC}(\mathcal{T}_f^i, p)), & \text{se } \mathcal{SC}(\mathcal{T}_f^i, p) \in \mathcal{Nr}(i), \\ 0, & \text{caso contrário.} \end{cases} \quad (5.5)$$

Os fatos apresentados levam a algoritmos eficientes para computar operadores últimos *levelings* (FABRIZIO; MARCOTEGUI, 2009; ALVES; HASHIMOTO, 2014).

5.3 ESTRATÉGIAS PARA FILTRAGEM DE RESÍDUOS INDESEJÁVEIS

Durante o processo de extração residual pelos operadores últimos *levelings* é comum que valores sejam extraídos de regiões indesejáveis. Por conta da natureza dos operadores últimos *levelings* esses valores muitas vezes se sobrepõe aos desejáveis. Uma maneira de minimizar este problema é o uso de estratégias de filtragem residual. Nesse sentido, queremos projetar classificadores $\Omega : \mathcal{P}(\mathcal{D}) \rightarrow \{\text{desejável}, \text{indesejável}\}$ para filtrar os resíduos. Na Figura 5.3 é apresentado um exemplo de filtragem de resíduos.

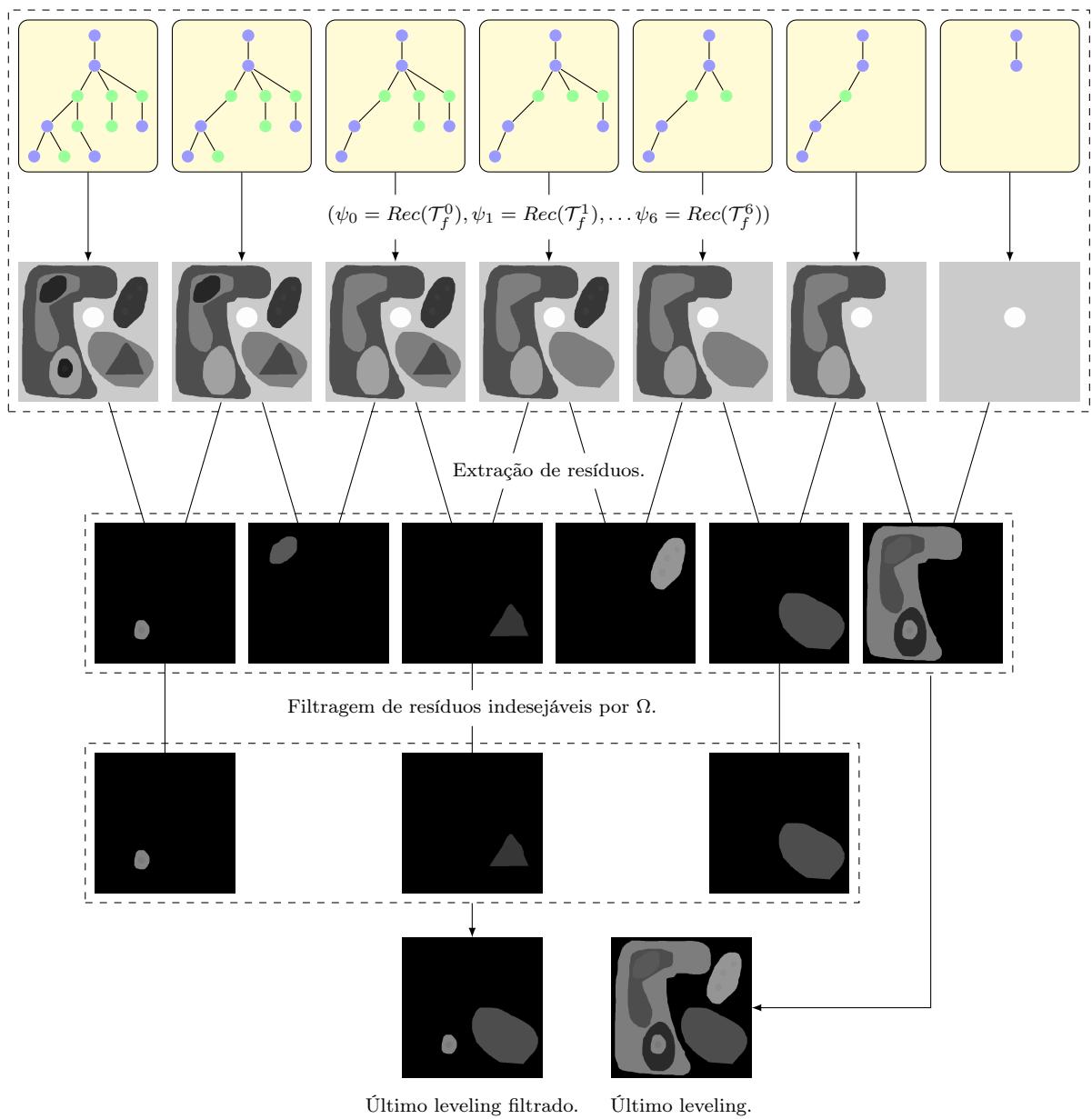


Figura 5.3: Comparação de um operador último leveling filtrado e não filtrado.

Existem diversas formas de planejar estes classificadores, contudo todas elas partem de um mesmo princípio. Para decidir se um resíduo $r_i^+(f)$ (respectivamente, $r_i^-(f)$) é filtrado ou não, é necessário apenas verificar se algum vértice $\tau \in \mathcal{N}r(i)$ é classificado por Ω como desejável. Nesse sentido, uma maneira de construir classificadores para resíduos de últimos *levelings* é através de atributos computados nos vértices $\tau \in \mathcal{N}r(i)$. Um atributo denotado por κ é visto como um mapeamento $\kappa : \mathcal{P}(\mathcal{D}) \rightarrow \mathbb{R}$. Exemplos de atributos conhecidos são: $\kappa_{\text{área}}$, κ_{largura} , κ_{altura} , $\kappa_{\text{perímetro}}$, etc. Em adição, denota-se por $\kappa(\tau)$ o valor de um atributo para um dado nó $\tau \in \mathcal{N}r(i)$. Na próxima seção é apresentada uma revisão da literatura sobre funções de energia. Em vista dessa consolidada abordagem são mostradas novas estratégias para filtragem de resíduos dos últimos *levelings* baseadas em funções de energia.

5.4 ESTRATÉGIAS BASEADAS EM FUNÇÕES DE ENERGIA

O problema de obter uma segmentação ótima da imagem é um assunto que vem sendo estudado pela comunidade científica há muitos anos (FU; MUI, 1981; SKARBEK; KOSCHAN, 1994; FREIXENET et al., 2002; PENG; ZHANG; ZHANG, 2013). Uma abordagem que é muito consolidada e ainda estudada é a do uso de funções de energia para segmentar imagens. Nesse sentido, o trabalho seminal de Mumford e Shah (1989) contribuiu com a consolidação das funções de energia para segmentação de imagens. A próxima seção apresenta uma revisão da literatura do assunto, começando pelo trabalho de Mumford e Shah (1989), seguindo pelos trabalhos de Ballester et al. (2007), Xu, Géraud e Najman (2013) e terminando no trabalho de Xu, Géraud e Najman (2016).

5.4.1 O FUNCIONAL DE MUMFORD-SHAH

De acordo com Mumford e Shah (1989) uma imagem f pode ser modelada como uma função seccionalmente suave, isto é, uma função restrita a uma decomposição de \mathcal{D} , ou seja, uma partição. Nesse sentido, a segmentação de uma imagem é vista como a busca de uma partição ótima de f . Um *funcional* pode ser visto como um mapeamento de um conjunto de funções X em um único escalar em \mathbb{R} , isto é, $X \rightarrow \mathbb{R}$ (KREYSZIG, 2007). No caso do funcional de Mumford-Shah esse mapeamento depende do conjunto de imagens e partições sobre \mathcal{D} , isto é, $E : \mathcal{F}(\mathcal{D}) \times \mathbb{P}(\mathcal{D}) \rightarrow \mathbb{R}$. Essa abordagem pode ser vista como uma medida de avaliação de uma partição, ou seja, o funcional de Mumford-Shah avalia uma partição (ou uma segmentação) dando a ela um valor escalar real, e essa é a indução por trás do funcional de Mumford-Shah. Assim, seja $\partial P = \{\partial R_1, \dots, \partial R_n\}$ do conjunto dos contornos de cada região $R_i \in P$ (veja um exemplo na Figura 5.4), o funcional de

Mumford-Shah é definido como:

$$E(f, P) = \sigma^2 \times \int \int_P (\tilde{f} - f)^2 dx dy + \int \int_{P - \partial P} \|\nabla \tilde{f}\| dx dy + \nu \times |\partial P|, \quad (5.6)$$

onde σ^2 e ν são parâmetros, o primeiro termo expressa que \tilde{f} aproxima f , o segundo termo expressa que \tilde{f} e f não variam muito nas regiões $R \in P$ e o terceiro termo penaliza o funcional através dos contornos ∂P .

Uma abordagem interessante é obtida restringindo E para funções constantes definidas por partes, isto é, \tilde{f} é constante em cada região $R \in P$ denotado por $\mu(R)$, isto é, a média da região. Nesse caso, o segundo termo é eliminado da equação (isto é, $\nabla \tilde{f} = 0$) e assim temos que:

$$E(f, P) = \sigma^2 \times \int \int_P (\mu(R) - f)^2 dx dy + \nu \times |\partial P|, \quad (5.7)$$

onde $\mu(R) = \frac{1}{|R|} \int \int f dx dy$ é a média da região R e $|R|$ é a área de R .

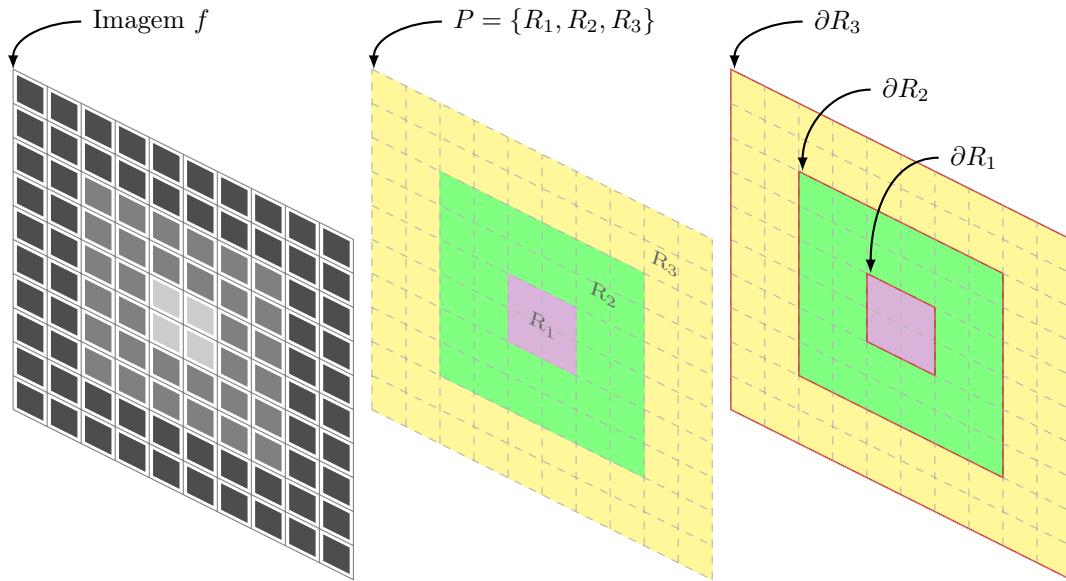


Figura 5.4: Exemplo do conjunto dos contornos ∂P .

Um caso especial do funcional de Mumford-Shah é obtido quando multiplica-se os dois lados da Equação 5.7 por σ^{-2} , isto é:

$$\begin{aligned} \sigma^{-2} \times E(f, P) &= [\sigma^2 \times \int \int_P (\mu(R) - f)^2 dx dy + \nu \times |\partial P|] \times \sigma^{-2} \\ \sigma^{-2} \times E(f, P) &= \int \int_P (\mu(R) - f)^2 dx dy + \sigma^{-2} \times \nu \times |\partial P|. \end{aligned}$$

Assim, chama-se $E_0 = \sigma^{-2} \times E(f, \partial P)$ e $\nu_0 = \sigma^{-2} \times \nu$. Logo temos:

$$E_0(f, P) = \int \int_P (\mu(R) - f)^2 dx dy + \nu_0 \times |\partial P|, \quad (5.8)$$

É possível mostrar que E_0 é o limite funcional de E , quando fixamos um conjunto ∂P e σ tende a 0, isso faz com que E passe a depender apenas do parâmetro ν , e então dessa forma o funcional Mumford-Shah pode ser reescrito como:

$$E(f, P) = \int \int_P (\mu(R) - f)^2 dx dy + \nu \times |\partial P|. \quad (5.9)$$

Intuitivamente a Equação 5.9 apresenta uma abordagem simplificada, porém interessante para segmentar imagens. Veja que, o segundo termo da Equação 5.9 penaliza o funcional utilizando os contornos e um parâmetro de escala ν enquanto o primeiro termo busca ao mesmo tempo uma região que seja homogênea, isto é, uma partição ótima (que tenha energia mínima) deve ter regiões homogêneas com poucos contornos. Muito dos trabalhos encontrados na literatura, inclusive os que são explorados neste capítulo, mostram abordagens que se baseiam em minimizar a Equação 5.9. Na próxima seção são apresentados alguns trabalhos relacionados com a abordagem de obter a segmentação de uma imagem utilizando funções de energia, e a partir da representação de uma imagem em árvores morfológicas são mostrados algoritmos para minimizar o funcional de Mumford-Shah dada pela Equação 5.9.

5.4.2 MINIMIZANDO FUNÇÕES DE ENERGIA A PARTIR DE ÁRVORES MORFOLÓGICAS

Conforme já foi introduzido, o trabalho seminal de [Mumford e Shah \(1989\)](#) contribuiu com uma nova abordagem para segmentar imagens a partir de funções de energia. Nosso foco de interesse estão nos trabalhos derivados do trabalho de [Mumford e Shah \(1989\)](#) que envolveram a utilização de árvores morfológicas para representar imagens e então a busca pela minimização do funcional. É importante lembrar que, essa não é a única abordagem para minimizar o funcional de Mumford-Shah, porém no nosso entendimento, a literatura oferece algoritmos eficientes para trabalhar com imagens representadas em árvores. Mais especificamente, estamos interessados na versão discreta da Equação 5.9:

$$E_\nu(f, P) = \sum_{R \in P} (D(f, R) + \nu \times C(R)), \quad (5.10)$$

sendo os termos D e C :

$$D(f, R) = \sum_{p \in R} (\mu(R) - f(p))^2 \text{ e } C(R) = |\partial R|, \quad (5.11)$$

onde $\mu(R) = \frac{vol(R)}{|R|}$ é a média de f em cada região $R \in P$, ∂R é o conjunto dos pixels do contorno de R e $vol(R) = \sum_{p \in R} f(p)$ é o volume da região R .

Seguindo essa ideia, vamos iniciar revisando a abordagem proposta por [Ballester et al. \(2007\)](#) para minimizar a versão discreta do funcional de Mumford-Shah subordinada

a árvore de formas e então computar uma partição ótima.

5.4.2.1 O funcional variacional

As árvores morfológicas constituem estruturas de dados eficientes e não redundantes para representar imagens. O funcional de Mumford-Shah é escrito sobre uma partição. As árvores morfológicas representam hierarquicamente os componentes conexos dos conjuntos de níveis extraídos de uma imagem. De fato, uma árvore morfológica \mathcal{T} representa uma partição associada da imagem a partir dos CNPs, isto é, $P_{\mathcal{T}} = \{\hat{\tau} : \tau \in \mathcal{T}\}$. Dessa forma, o funcional de Mumford-Shah pode ser denotada por $E_{\nu}(f, P_{\mathcal{T}})$ e então a energia é subordinada a uma árvore morfológica \mathcal{T} .

Dada uma árvore \mathcal{T} uma versão simplificada de \mathcal{T} pode ser obtida removendo alguns nós de \mathcal{T} e atualizando a relação de parentesco. Dessa forma, o funcional de Mumford-Shah pode ser minimizado a partir da computação de uma sequência de partições associadas $(P_{\mathcal{T}_0}, P_{\mathcal{T}_1}, \dots, P_{\mathcal{T}_n})$, tal que $(\mathcal{T}_0, \mathcal{T}_1, \dots, \mathcal{T}_n)$ são árvores que geraram as partições, onde \mathcal{T}_{i+1} é uma versão simplificada de \mathcal{T}_i para $0 \leq i < n$ e $E_{\nu}(f, P_{\mathcal{T}_n}) \leq E_{\nu}(f, P_{\mathcal{T}_{n-1}}) \leq E_{\nu}(f, P_{\mathcal{T}_{n-2}}) \leq \dots \leq E_{\nu}(f, P_{\mathcal{T}_0})$. Mais especificamente, estamos interessados em encontrar a versão simplificada \mathcal{T}^* de \mathcal{T} que tem energia mínima, isto é:

$$\mathcal{T}^* = \arg \min \{E_{\nu}(f, P_{\mathcal{T}_{i+1}}) : \mathcal{T}_{i+1} \text{ é uma árvore simplificada de } \mathcal{T}_i\}. \quad (5.12)$$

A remoção de um dado nó $\tau \in \mathcal{T}_i$ pode ser computada como a união entre duas regiões da partição associada na árvore \mathcal{T}_i , o que gera um novo nó na árvore simplificada \mathcal{T}_{i+1} , isto é, $\mathcal{T}_{i+1} = Remove(\mathcal{T}_i, \tau)$ sendo que $\tau' \in \mathcal{T}_{i+1}$, tal que $\hat{\tau}' = \hat{\tau}_{Pai(\tau)} \cup \hat{\tau}$, onde $\hat{\tau}_{Pai(\tau)}$ é a região dada pelo pai de τ em \mathcal{T}_i . Os filhos do nó removido em \mathcal{T}_i terão a relação de parentesco atualizada em \mathcal{T}_{i+1} , isto é, o nó $\tau' \in \mathcal{T}_{i+1}$ é pai dos irmãos e dos filhos do nó removido $\tau \in \mathcal{T}_i$ (note que, isso acontece se τ não é um nó folha e se ele tem irmãos). Um exemplo que ilustra esta abordagem pode ser visto na Figura 5.5.

Para decidir a respeito da remoção de um dado nó $\tau \in \mathcal{T}$, Ballester et al. (2007) propõe o uso de uma medida baseada no funcional de Mumford-Shah chamada de *funcional variacional* (veja também Koepfler, Lopez e Morel (1994) e Morel e Solimini (1995)). Essa medida é definida como a diferença de energia de uma árvore \mathcal{T} com e sem um dado nó τ , e é dada por:

$$\Delta E(\mathcal{T}, \tau) = E_{\nu}(f, P_{\mathcal{T}}) - E_{\nu}(f, P_{\mathcal{T} \setminus \{\tau\}}). \quad (5.13)$$

Note que, somente as regiões $\hat{\tau} \in P_{\mathcal{T}}$, $\hat{\tau}' = \hat{\tau}_{Pai(\tau)} \in P_{\mathcal{T}}$ e $\hat{\tau}'' = \hat{\tau} \cup \hat{\tau}_{Pai(\tau)} \in P_{\mathcal{T} \setminus \{\tau\}}$ estão na diferença simétrica entre $P_{\mathcal{T}}$ e $P_{\mathcal{T} \setminus \{\tau\}}$, então podemos reescrever a Equação 5.13 da seguinte forma:

$$\begin{aligned}\Delta E(\mathcal{T}, \tau) &= E_\nu(f, P_{\mathcal{T}}) - E_\nu(f, P_{\mathcal{T} \setminus \{\tau\}}) \\ &= [D(\hat{\tau}) + \nu \times C(\hat{\tau})] + [D(\hat{\tau}') + \nu \times C(\hat{\tau}')] - [D(\hat{\tau}'') + \nu \times C(\hat{\tau}'')].\end{aligned}\quad (5.14)$$

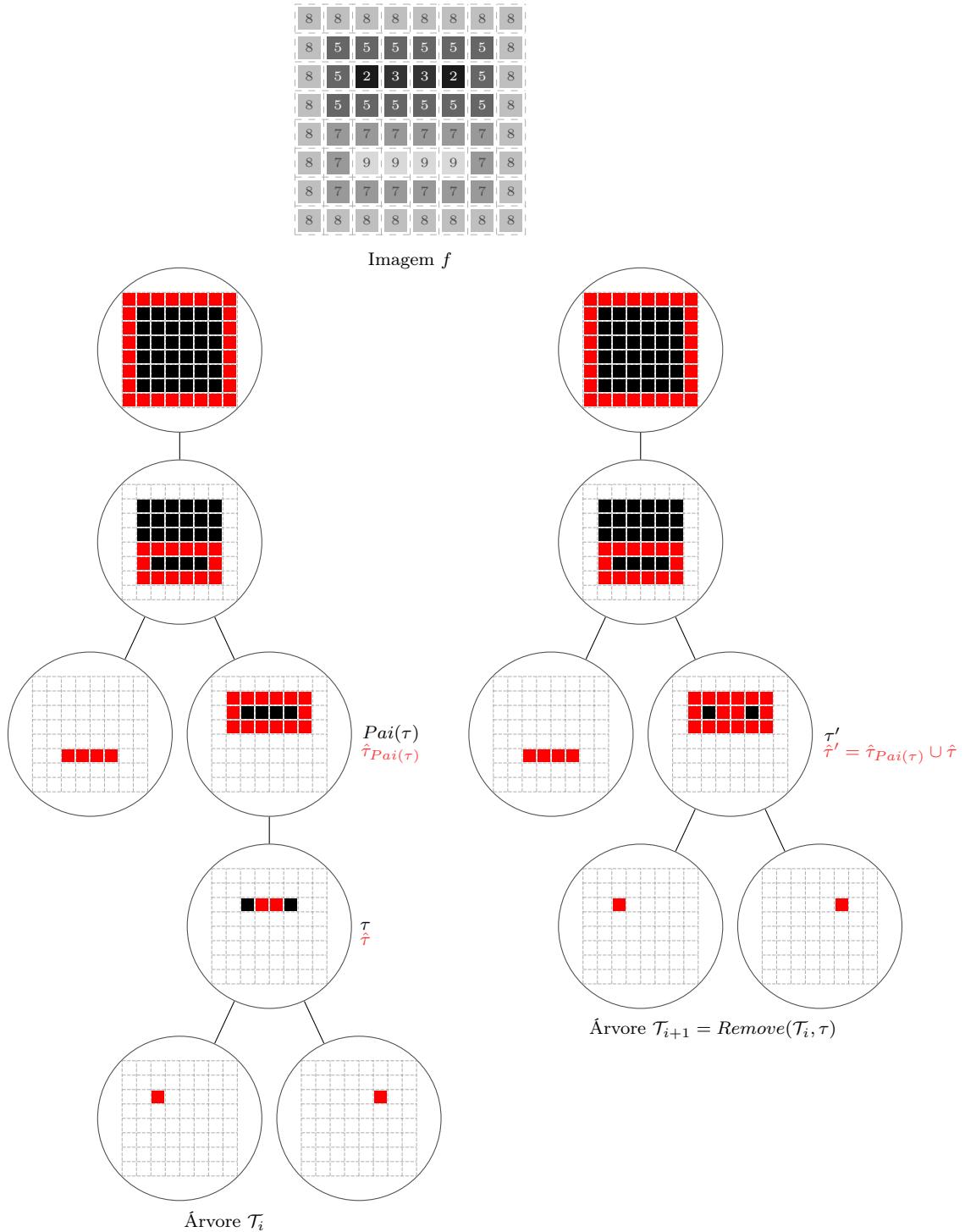


Figura 5.5: Exemplo de árvore simplificada T_{i+1} computada a partir de T_i . A árvore utilizada no exemplo é uma árvore de formas.

Com algumas propriedades algébricas e recorrendo as conhecidas: propriedades da esperança¹ e da variância^{2,3} podemos mostrar que a Equação 5.14 pode ser simplificada.

Proposição 5.1. *Se o conjunto $\hat{\tau}$ é uma região de uma partição associada, isto é, $\hat{\tau} \in \mathcal{P}_{\mathcal{T}}$ então $D(\hat{\tau}) = vol_Q(\hat{\tau}) - \frac{[vol(\hat{\tau})]^2}{|\hat{\tau}|}$, onde $vol_Q(\hat{\tau}) = \sum_{p \in \hat{\tau}} f(p)^2$.*

Prova: Primeiramente, considere a variância de $\hat{\tau}$, isto é:

$$Var(\hat{\tau}) = \frac{1}{|\hat{\tau}|} D(\hat{\tau}) = \frac{1}{|\hat{\tau}|} \sum_{p \in \hat{\tau}} (E(\hat{\tau}) - f(p))^2.$$

Claramente, podemos passar $|\hat{\tau}|$ para o outro lado e portanto:

$$|\hat{\tau}| \times Var(\hat{\tau}) = \sum_{p \in \hat{\tau}} (E(\hat{\tau}) - f(p))^2.$$

Logo, pela definição alternativa da variância temos que:

$$\begin{aligned} |\hat{\tau}| \times Var(\hat{\tau}) &= |\hat{\tau}| \times \left[E_Q(\hat{\tau}) - [E(\hat{\tau})]^2 \right] \\ &\iff |\hat{\tau}| \times \left[\frac{\sum_{p \in \hat{\tau}} f(p)^2}{|\hat{\tau}|} - \frac{\left(\sum_{p \in \hat{\tau}} f(p) \right)^2}{|\hat{\tau}|^2} \right] \\ &\iff \sum_{p \in \hat{\tau}} f(p)^2 - \frac{\left(\sum_{p \in \hat{\tau}} f(p) \right)^2}{|\hat{\tau}|} \\ &\iff vol_Q(\hat{\tau}) - \frac{[vol(\hat{\tau})]^2}{|\hat{\tau}|}. \end{aligned}$$

Assim, concluímos que $D(\hat{\tau}) = vol_Q(\hat{\tau}) - \frac{[vol(\hat{\tau})]^2}{|\hat{\tau}|}$. \square

Agora, com ajuda da Proposição 5.1, podemos reescrever a Equação 5.14 na sua versão simplificada:

$$\Delta E(\mathcal{T}, \tau) = [D(\hat{\tau}) + \nu \times C(\hat{\tau})] + [D(\hat{\tau}') + \nu \times C(\hat{\tau}')] - [D(\hat{\tau}'') + \nu \times C(\hat{\tau}'')]$$

¹ $E(X) = \frac{1}{|X|} \sum_{x \in X} f(x)$ e $E_Q(X) = \frac{1}{|X|} \sum_{x \in X} f(x)^2$,

² $Var(X) = \frac{1}{|X|} \sum_{x \in X} (E(x) - f(x))^2$,

³ $Var(X) = E_Q(X) - [E(X)]^2$, para $X \subseteq \mathcal{D}$.

$$\begin{aligned}
&= D(\hat{\tau}) + D(\hat{\tau}') - D(\hat{\tau}'') + \nu \times (C(\hat{\tau}) + C(\hat{\tau}') - C(\hat{\tau}'')) \\
&= \sum_{p \in \hat{\tau}} (\mu(\hat{\tau}) - f(p))^2 + \sum_{p \in \hat{\tau}'} (\mu(\hat{\tau}') - f(p))^2 - \sum_{p \in \hat{\tau}''} (\mu(\hat{\tau}'') - f(p))^2 \\
&\quad + \nu \times (C(\hat{\tau}) + C(\hat{\tau}') - C(\hat{\tau}'')) \\
&= \left(|\hat{\tau}| \times \left[\frac{\sum_{p \in \hat{\tau}} f(p)^2}{|\hat{\tau}|} - \frac{\left(\sum_{p \in \hat{\tau}} f(p) \right)^2}{|\hat{\tau}|^2} \right] \right) + \left(|\hat{\tau}'| \times \left[\frac{\sum_{p \in \hat{\tau}'} f(p)^2}{|\hat{\tau}'|} - \frac{\left(\sum_{p \in \hat{\tau}'} f(p) \right)^2}{|\hat{\tau}'|^2} \right] \right) \\
&\quad - \left(|\hat{\tau}''| \times \left[\frac{\sum_{p \in \hat{\tau}''} f(p)^2}{|\hat{\tau}''|} - \frac{\left(\sum_{p \in \hat{\tau}''} f(p) \right)^2}{|\hat{\tau}''|^2} \right] \right) + \nu \times (C(\hat{\tau}) + C(\hat{\tau}') - C(\hat{\tau}'')) \\
&= \sum_{p \in \hat{\tau}} f(p)^2 - \frac{\left(\sum_{p \in \hat{\tau}} f(p) \right)^2}{|\hat{\tau}|} + \sum_{p \in \hat{\tau}'} f(p)^2 - \frac{\left(\sum_{p \in \hat{\tau}'} f(p) \right)^2}{|\hat{\tau}'|} - \sum_{p \in \hat{\tau}} f(p)^2 - \sum_{p \in \hat{\tau}'} f(p)^2 \\
&\quad + \frac{\left(\sum_{p \in \hat{\tau}''} f(p) \right)^2}{|\hat{\tau}''|} + \nu \times (C(\hat{\tau}) + C(\hat{\tau}') - C(\hat{\tau}'')) \quad (\hat{\tau} \cap \hat{\tau}' = \emptyset) \\
&= vol_Q(\hat{\tau}) - \frac{[vol(\hat{\tau})]^2}{|\hat{\tau}|} + vol_Q(\hat{\tau}') - \frac{[vol(\hat{\tau}')]^2}{|\hat{\tau}'|} - vol_Q(\hat{\tau}) - vol_Q(\hat{\tau}') + \frac{[vol(\hat{\tau}'')]^2}{|\hat{\tau}''|} \\
&\quad + \nu \times (|\partial \hat{\tau}| + |\partial \hat{\tau}'| - |\partial \hat{\tau}''|) \\
&= \frac{[vol(\hat{\tau}'')]^2}{|\hat{\tau}''|} - \frac{[vol(\hat{\tau})]^2}{|\hat{\tau}|} - \frac{[vol(\hat{\tau}')]^2}{|\hat{\tau}'|} + \nu \times (|\partial \hat{\tau}| + |\partial \hat{\tau}'| - |\partial \hat{\tau}''|). \quad (5.15)
\end{aligned}$$

Se para um dado nó $\tau \in \mathcal{T}$ o valor do funcional variacional dada pela Equação 5.15 for estritamente positivo, então isso indica que a remoção de τ decresce (respectivamente, aumenta) o valor da energia (respectivamente, funcional variacional) e por consequência ele deve ser removido. Assim, com uma abordagem gulosa podemos obter uma solução ótima local da Equação 5.12 por meio da seguinte recorrência:

$$\mathcal{T}_{i+1} = \begin{cases} Remove(\mathcal{T}_i, \arg \min_{\tau \in \mathcal{T}_i} \Delta E(\mathcal{T}_i, \tau)), & \text{se } \Delta E(\mathcal{T}_i, \tau) > 0 \\ \mathcal{T}_i, & \text{caso contrário,} \end{cases} \quad (5.16)$$

quando $\mathcal{T}_{i+1} = \mathcal{T}_i$ temos que \mathcal{T}_i é uma árvore ótima local \mathcal{T}^* . Abordagens gulosas são conhecidas na literatura por prover soluções ótimas locais em problemas onde as soluções ótimas globais tem alto custo computacional. Nesse sentido, [Ballester et al. \(2007\)](#) propõe implementar a Recorrência 5.16 com o auxílio de uma estrutura de ordenação chamada *Fila de Prioridade*. A ideia central do algoritmo é inserir os nós nessa fila e então consumi-

los de acordo com a prioridade da fila, sendo que a maior prioridade corresponde ao maior valor de ΔE . O algoritmo termina quando não houver mais remoções de nós que decrescem a energia. Note que, cada remoção de um nó implica na atualização de parentesco de seus filhos e irmãos, assim a ordem de remoção dos nós tem impacto no resultado obtido (veja a Figura 5.6). Explorando essa lacuna, Xu, Géraud e Najman (2013) propõe uma abordagem heurística para explorar os nós. Vamos revisar essa abordagem com mais detalhes na próxima seção.

5.4.2.2 Ordem de remoção dos nós

Conforme explorado na seção anterior, Ballester et al. (2007) apresentam uma abordagem gulosa para implementar a Recorrência 5.16 baseada em uma fila de prioridade. Por um lado essa abordagem explora os nós seguindo a ordem da fila de prioridade, e além disso ocorrem atualizações na fila e re-computações na árvore toda vez que um nó é removido. Por outro lado, Xu, Géraud e Najman (2013) propõe uma abordagem heurística para explorar os nós com o intuito de eliminar o custo das atualizações das prioridades. Essa heurística envolve o uso de um atributo para decidir a ordem em que os nós são explorados, e consequentemente removidos. Mais especificamente, um atributo de significância (alguns deles são revisados em Xu (2013)). Assim, o atributo de significância utilizado foi a magnitude do gradiente ao longo do contorno. Para um dado nó $\tau \in \mathcal{T}$, a magnitude do gradiente ao longo do contorno é dada por:

$$\mathcal{G}_\nabla(\tau) = \frac{S_\nabla(\tau)}{|\partial\tau|}, \quad (5.17)$$

onde $S_\nabla(\tau) = \sum_{e \in \partial\tau} (\nabla g(e))$ é a soma do gradiente dos contornos de τ , e é um pixel que faz parte do conjunto de contornos $\partial\tau$ e ∇g é a imagem do gradiente de f . Na prática, para obter ∇g utilizamos um dos detectores de bordas conhecidos no estado da arte, como por exemplo: o detector de bordas de *Sobel* ou de *Prewitt* (uma *survey* a respeito de detectores de bordas pode ser encontrada em Bhardwaj e Mittal (2012)). Assim, a abordagem de Xu, Géraud e Najman (2013) funciona da seguinte forma: exploram-se os nós em ordem crescente de acordo com o Atributo 5.17, então os nós são removidos pela Recorrência 5.16, da mesma forma que a abordagem de Ballester et al. (2007) as remoções dos nós implicam na atualização de parentesco de seus filhos e irmãos. O algoritmo termina quando não há mais nós a serem explorados.

Em comparação a abordagem de Ballester et al. (2007), na abordagem de Xu, Géraud e Najman (2013) não há o custo das atualizações das prioridades como ocorre com a fila de prioridade, além disso conforme relatado por Xu, Géraud e Najman (2013) na maioria dos casos a heurística proposta converge mais rapidamente, pois remove mais nós por passo do algoritmo. Apesar disso, por conta da abordagem heurística e de não haver atualizações de

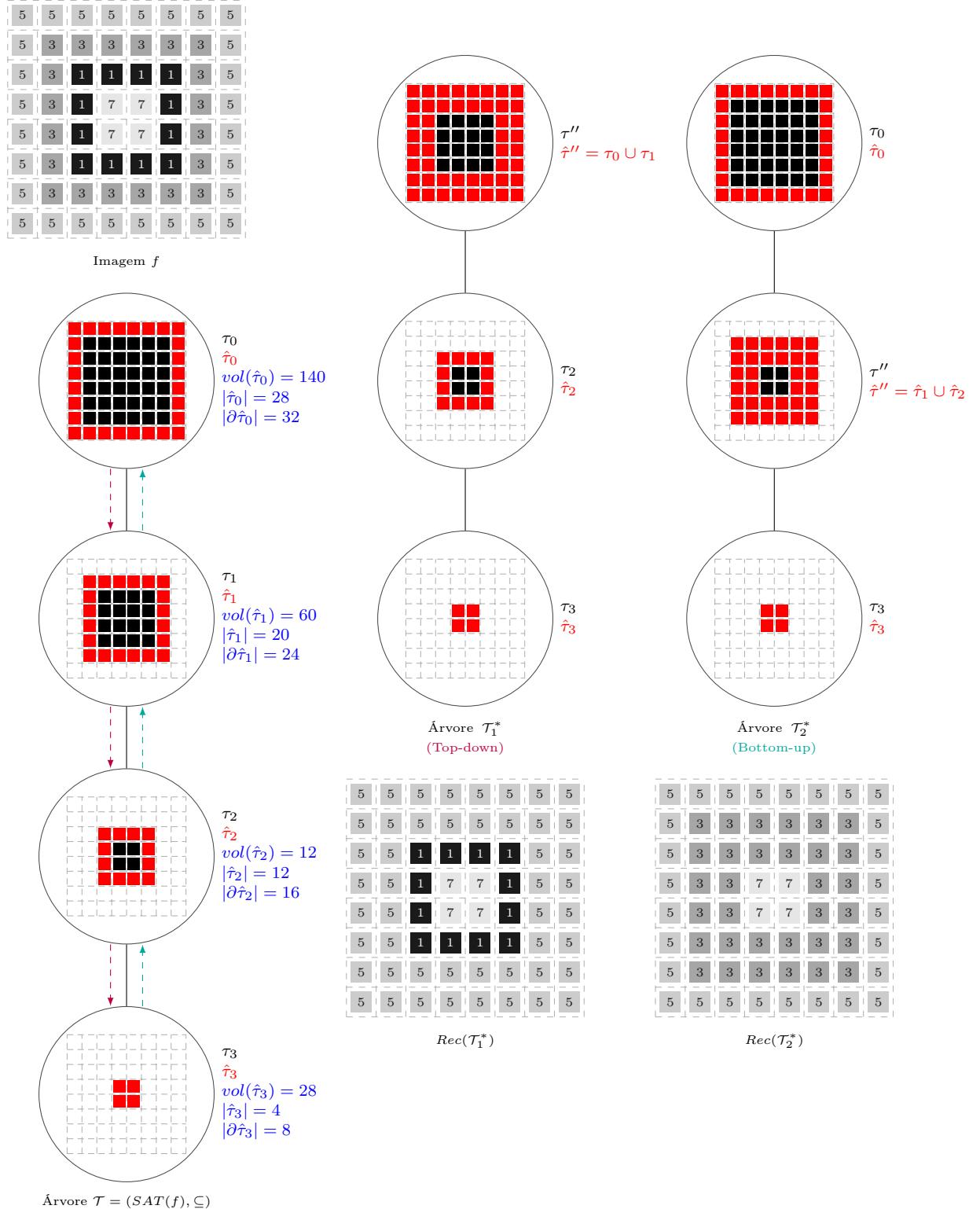


Figura 5.6: Exemplo de computação de árvores ótimas a partir de duas abordagens diferentes para explorar os nós, respectivamente Top-down e Bottom-up, com $\nu = 2$.

prioridade o algoritmo proposto por Xu, Géraud e Najman (2013) pode vir a remover nós com energia menor que outros, visto que não há uma prioridade na remoção como na fila em Ballester et al. (2007), contudo, vale ressaltar que essa abordagem produz também uma

solução ótima local e segundo os autores essa abordagem tem grande robustez a ruídos. Na Figura 5.6 é apresentado um exemplo mostrando o impacto no resultado devido a mudança na ordem em que os nós são explorados (*Top-down* e *Bottom-up*), as árvores ótimas foram computadas pela Recorrência 5.16 com $\nu = 2$, atualizando as relações de parentesco dos filhos e dos irmãos dos nós removidos quando necessário.

Até aqui, apresentamos duas abordagens para minimizar o funcional de Mumford-Shah, ambas utilizando árvores morfológicas. Um parâmetro pouco explorado até o momento foi o parâmetro ν . Esse parâmetro é também chamado de parâmetro de escala do funcional de Mumford-Shah e através dele é possível controlar o nível de simplificação da imagem. Nos trabalhos de Ballester et al. (2007) e Xu, Géraud e Najman (2013) o parâmetro ν é fixado. Quanto maior o valor de ν , mais simplificada a imagem fica, isso porque mais nós serão removidos e por consequência mais uniões de regiões da imagem irão ocorrer. De fato, encontrar um valor ótimo de escala ν não é uma tarefa fácil, visto que muitas vezes objetos de interesse numa imagem não pertencem a uma escala específica, mas sim a diversas (KOENDERINK, 1984). Explorando essa lacuna, Xu, Géraud e Najman (2016) propõe computar um atributo baseado no funcional variacional, assim eliminando a necessidade de fixar o parâmetro de escala ν . Essa abordagem é revisada na próxima seção.

5.4.2.3 O atributo funcional

Uma forma de construir filtros conexos é utilizando atributos baseados em regiões de imagens. Como já foi introduzido, árvores morfológicas podem representar eficientemente regiões de imagens por meio da hierarquia de níveis de cinza. A literatura apresenta diversos atributos que podem ser eficientemente calculados nas árvores morfológicas, tais como: área, perímetro, altura, largura entre outros. Em adição, vale relembrar que, esses atributos podem ser usados para construir estratégias para filtragem de resíduos dos últimos *levelings*. Nesse sentido, Xu, Géraud e Najman (2016) propõe computar um atributo baseado no funcional variacional de Ballester et al. (2007). Vale ressaltar que, o funcional variacional pode ser utilizada como um atributo, porém, pela sua definição o parâmetro de escala ν tem que ser fixado, é possível também variar a escala durante a computação do algoritmo, contudo não há garantia que o princípio da causalidade seja respeitado. Esse princípio é um dos mais fundamentais a respeito de abordagens multiescalas (ele já foi mencionado no Capítulo 3) e foi apresentado por Koenderink (1984). Em respeito ao funcional variacional, pode ocorrer que em uma dada escala ν_1 tenham nós removidos que seriam preservados numa escala ν_2 quando $\nu_2 > \nu_1$ e isso não é válido pelo princípio da causalidade. Na Figura 5.7 é apresentado um exemplo de computação de árvores ótimas pela Recorrência 5.16 utilizando duas escalas distintas, respectivamente ν_1 e ν_2 , sendo $\nu_2 > \nu_1$, note que os nós τ_2 e τ_3 foram removidos na escala ν_1 , mas não na escala ν_2 .

e isso invalida o princípio da causalidade, vale ressaltar que, a árvore foi explorada em abordagem *Top-down*.

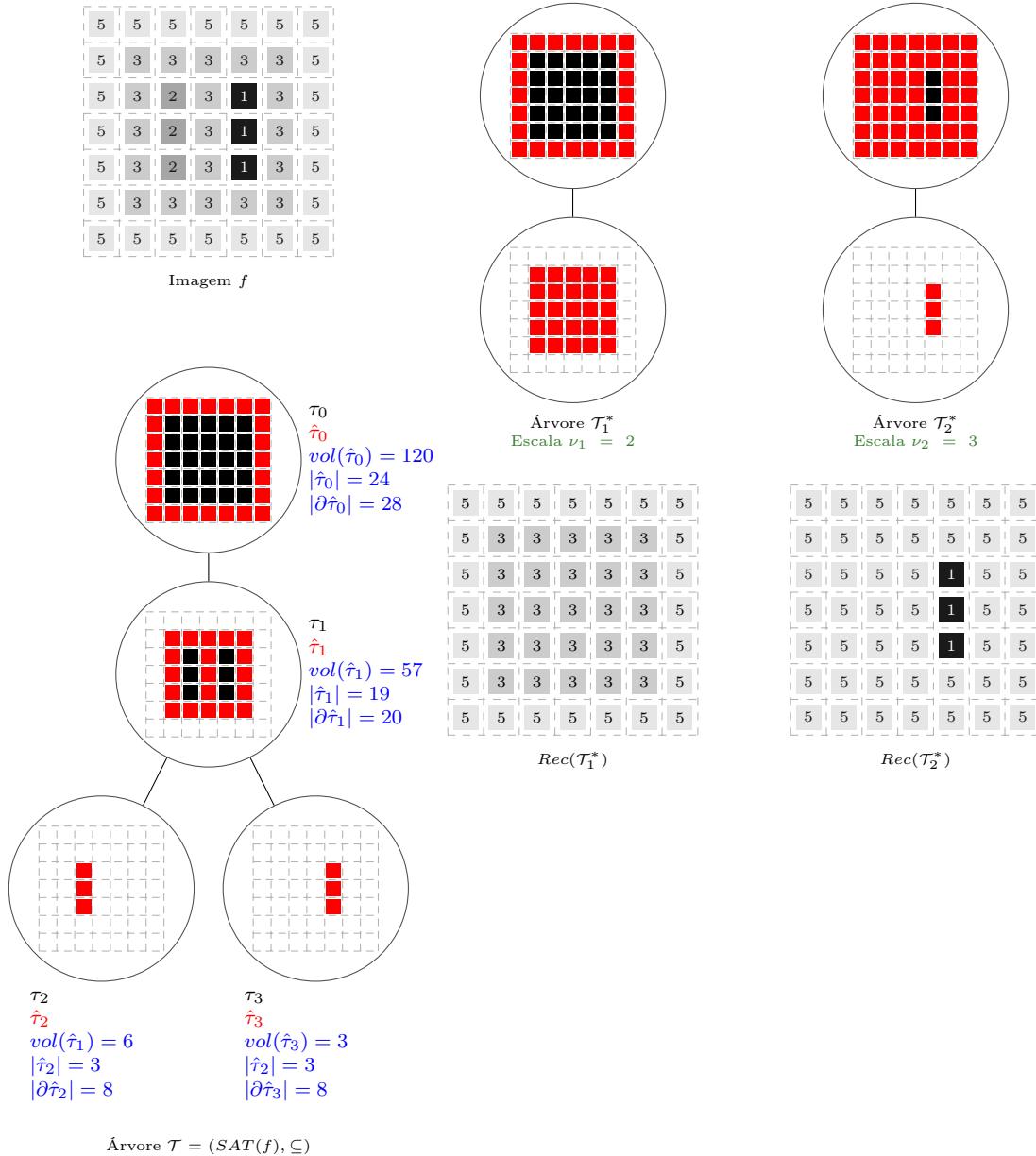


Figura 5.7: Exemplo que não valida o princípio da causalidade, sendo $\nu_2 > \nu_1$.

Assim, Xu, Géraud e Najman (2016) propõe não mais fixar o parâmetro de escala ν , dando origem ao atributo funcional. Baseado no funcional variacional dado pela Equação 5.15, para um dado nó $\tau \in \mathcal{T}$ o *atributo funcional* é definido da seguinte forma:

$$\begin{aligned}
\kappa_{ms}(\tau) &= \frac{-\Delta E_\nu(\mathcal{T}, \tau)}{C(\hat{\tau}) + C(\hat{\tau}') - C(\hat{\tau}'')} + \nu \\
&= \frac{-\left[\frac{[vol(\hat{\tau}'')]^2}{|\hat{\tau}''|} - \frac{[vol(\hat{\tau})]^2}{|\hat{\tau}|} - \frac{[vol(\hat{\tau}')]^2}{|\hat{\tau}'|}\right]}{C(\hat{\tau}) + C(\hat{\tau}') - C(\hat{\tau}'')} - \frac{\nu \times (C(\hat{\tau}) + C(\hat{\tau}') - C(\hat{\tau}''))}{C(\hat{\tau}) + C(\hat{\tau}') - C(\hat{\tau}'')} + \nu \\
&= -\left[\frac{[vol(\hat{\tau}'')]^2}{|\hat{\tau}''|} - \frac{[vol(\hat{\tau})]^2}{|\hat{\tau}|} - \frac{[vol(\hat{\tau}')]^2}{|\hat{\tau}'|}\right] \times \frac{1}{C(\hat{\tau}) + C(\hat{\tau}') - C(\hat{\tau}'')} \\
&= \frac{[vol(\hat{\tau})]^2}{|\hat{\tau}|} + \frac{[vol(\hat{\tau}')]^2}{|\hat{\tau}'|} - \frac{[vol(\hat{\tau}'')]^2}{|\hat{\tau}''|} \times \frac{1}{|\partial\hat{\tau}| + |\partial\hat{\tau}'| - |\partial\hat{\tau}''|}. \tag{5.18}
\end{aligned}$$

O atributo funcional mede a persistência de um nó ser removido em escalas mais grosseiras, quanto maior o valor de κ_{ms} para um dado nó $\tau \in \mathcal{T}$, mais difícil é de remover esse nó, isto equivale a dizer que em mais escalas grosseiras o nó está presente. Assim, o atributo funcional é computado nas árvores morfológicas da seguinte forma: A árvore é percorrida e para todos os nós é calculado um valor inicial de κ_{ms} , depois os nós são ordenados de acordo com o Atributo 5.17, então um a um os nós são removidos e são atualizadas as relações de parentesco dos seus filhos e irmãos, o valor de κ_{ms} do nó removido é atualizado toda vez que ele é maior que o inicialmente computado. O algoritmo termina após explorar todos os nós. Note que, da mesma forma que a abordagem anterior, a heurística para explorar a árvore pode vir a remover nós com menor valor de energia que outros, pois não há prioridade na remoção.

5.4.3 ESTRATÉGIAS

Apresentou-se nas seções anteriores uma revisão literária sobre o funcional de Mumford-Shah. A partir das abordagens de Ballester et al. (2007) e Xu, Géraud e Najman (2016) podemos definir duas novas estratégias para filtragem de resíduos dos últimos *levelings*. Assim, para um dado nó $\tau \in \mathcal{T}$ e dois parâmetros de *threshold* $\varepsilon^{min}, \varepsilon^{max} \in \mathbb{R}$, definem-se duas novas estratégia para filtragem de resíduos dos últimos *levelings*, a primeira a partir do funcional variacional ΔE denotado por $\kappa_{\Delta E}$:

$$\Omega_{\kappa_{\Delta E}}(\tau) = \begin{cases} \text{desejável,} & \text{se } \varepsilon^{max} > \kappa_{\Delta E}(\tau) > \varepsilon^{min}, \\ \text{indesejável,} & \text{caso contrário,} \end{cases} \tag{5.19}$$

e a segunda a partir do atributo funcional κ_{ms} :

$$\Omega_{\kappa_{ms}}(\tau) = \begin{cases} \text{desejável,} & \text{se } \varepsilon^{max} > \kappa_{ms}(\tau) > \varepsilon^{min}, \\ \text{indesejável,} & \text{caso contrário.} \end{cases} \tag{5.20}$$

Vale ressaltar que, em alguns casos o valor $\varepsilon^{max} = +\infty$, cabendo apenas selecionar o valor ε^{min} , dessa forma denota-se ε^{min} simplesmente por ε . Ambas as estratégia foram utilizadas pelo autor com sucesso para classificar resíduos de últimos *levelings* em duas aplicações: reconhecimento de plantas (atributo funcional) e segmentação de vasos sanguíneos de imagens de retina (funcional variacional e atributo funcional) os resultados são apresentados no Capítulo 6 com detalhes.

5.4.4 ALGORITMOS PARA COMPUTAÇÃO DOS ATRIBUTOS DE ENERGIA

Para computar os atributos de energia, são necessários computar antes alguns outros atributos. Esses atributos são: perímetro externo (contornos), área e volume. Em adição, no caso do atributo funcional κ_{ms} é necessário computar a imagem do gradiente por causa do atributo de ordenação \mathcal{G}_∇ , vale ressaltar que, no caso da árvore de formas a imagem do gradiente deve ser computada sobre a imagem f_S da Grid Khalimsky. Atributos como a área e o volume podem ser computados eficientemente, pois podem ser obtidos percorrendo os nós da árvore uma vez. Essa implementação é apresentada no Algoritmo 7 onde cada nó da árvore é visitado apenas uma vez, assim esse algoritmo tem complexidade de tempo na ordem $\mathcal{O}(|\mathcal{T}|)$. Vale ressaltar que, geralmente nos trabalhos apresentados na literatura atributos como área e volume são computados de forma acumulada (veja Xu et al. (2015)), porém, no caso dos atributos de energia a área e o volume dizem respeito apenas aos nós compactos. Para computar o atributo de perímetro externo denotado por $\kappa_{perímetro}$ nós utilizamos o algoritmo de Chain Code apresentado na página 223 do livro Burger e Burge (2008) onde também computa-se a soma dos gradientes dos contornos S_∇ .

Algoritmo 7: Algoritmo para computação de atributos de área e volume para árvores morfológicas.

\mathcal{T}	uma árvore morfológica	τ, τ'	nós da árvore \mathcal{T}
---------------	------------------------	---------------	-----------------------------

```

função pre-processamento( $\mathcal{T}$ )
  para todo  $\tau \in \mathcal{T}$  faça
     $\kappa_{area}(\hat{\tau}) \leftarrow |\hat{\tau}|$ 
     $\kappa_{vol}(\hat{\tau}) \leftarrow level(\hat{\tau}) \times |\hat{\tau}|$ 
  fim
fim

```

O atributo baseado no funcional variacional apresentado em Ballester et al. (2007) depende de uma fila de prioridade para ser computado. Conforme apresentado no Capítulo 4 optamos por utilizar nesse caso a fila de prioridade proposta por Falcao, Udupa e Miyazawa (2000). Assim, uma implementação do algoritmo para computação do funcional variacional de Ballester et al. (2007) é apresentada no Algoritmo 8. Vale ressaltar que, em cada iteração remove-se um dado nó de Q que possui o maior valor do funcional

variacional, contudo a ordem dada pela prioridade tem um custo adicional, então no pior caso temos uma complexidade de tempo na ordem de $\mathcal{O}(|\mathcal{T}|^2)$.

Algoritmo 8: Implementação do algoritmo para computação do funcional variacional de Ballester et al. (2007).

\mathcal{T}^*	árvore ótima de \mathcal{T}	τ, τ', τ''	nós da árvore \mathcal{T}^*
Q	fila de prioridade	ν	parâmetro de escala de Mumford Shah

```

função computa-energia-variacional( $\tau, \nu$ )
     $\tau' = Pai(\tau)$ 
    retorna  $\frac{[\kappa_{vol}(\hat{\tau}) + \kappa_{vol}(\hat{\tau}')]^2}{\kappa_{area}(\hat{\tau}) + \kappa_{area}(\hat{\tau}')} - \frac{[\kappa_{vol}(\hat{\tau})]^2}{\kappa_{area}(\hat{\tau})} - \frac{[\kappa_{vol}(\hat{\tau}')]^2}{\kappa_{area}(\hat{\tau}')} + \nu \times \kappa_{perimetro}(\tau)$ 
fim
função atualiza-energia( $Q, \tau, \nu$ )
    se  $\kappa_{\Delta E}(\tau) > 0$  então
        | remove( $Q, \tau, \kappa_{\Delta E}(\tau)$ )
    fim
     $\kappa_{\Delta E}(\tau) \leftarrow$  computa-energia-variacional( $\tau, \nu$ )
    se  $\kappa_{\Delta E}(\tau) > 0$  então
        | enqueue( $Q, \tau, \kappa_{\Delta E}(\tau)$ )
    fim
fim
função computa-funcional-variacional( $\mathcal{T}, \nu$ )
     $\mathcal{T}^* = \mathcal{T}$ 
    pre-processamento( $\mathcal{T}^*$ )
    para todo  $\tau \in \{\mathcal{T}^* \setminus Raiz(\mathcal{T}^*)\}$  faça
         $\kappa_{\Delta E}(\tau) \leftarrow$  computa-energia-variacional( $\tau, \nu$ )
        se  $\kappa_{\Delta E}(\tau) > 0$  então
            | enqueue( $Q, \tau, \kappa_{\Delta E}(\tau)$ )
        fim
    fim
    enquanto  $Q$  não está vazia faça
         $\tau \leftarrow$  dequeue ( $Q$ )
         $\tau' \leftarrow Pai(\tau)$ 
         $\mathcal{T}^* = Remove(\mathcal{T}^*, \tau)$ 
         $\kappa_{area}(\hat{\tau}') \leftarrow \kappa_{area}(\hat{\tau}') + \kappa_{area}(\hat{\tau})$ 
         $\kappa_{vol}(\hat{\tau}') \leftarrow \kappa_{vol}(\hat{\tau}') + \kappa_{vol}(\hat{\tau})$ 
        se  $\tau' \neq Raiz(\mathcal{T}^*)$  então
            | atualiza-energia( $Q, \tau', \nu$ )
        fim
        para todo  $\tau'' \in Filhos(\tau')$  faça
            | atualiza-energia( $Q, \tau'', \nu$ )
        fim
    fim
    retorna ( $\kappa_{\Delta E}, \mathcal{T}^*$ )
fim

```

Para computar o atributo funcional apresentado por Xu, Géraud e Najman (2016)

é necessário percorrer os nós pela ordem dada pelo atributo G_∇ . Mais uma vez, a fila de prioridade apresentada por [Falcao, Udupa e Miyazawa \(2000\)](#) se torna a opção mais eficiente. Uma implementação do algoritmo de [Xu, Géraud e Najman \(2016\)](#) é apresentada no Algoritmo 9. Vale ressaltar que, esse algoritmo percorre a árvore apenas uma vez e a estrutura da árvore é completamente consumida, assim a complexidade de tempo do algoritmo no pior caso é $\mathcal{O}(|\mathcal{T}|)$.

Algoritmo 9: Implementação do algoritmo para computação do atributo funcional de [Xu, Géraud e Najman \(2016\)](#).

\mathcal{T}^*	árvore ótima de \mathcal{T}	τ, τ', τ''	nós da árvore \mathcal{T}^*
Q	fila de prioridade	mínimo	prioridade da fila

```

função computa-energia-funcional( $\tau$ )
     $\tau' = Pai(\tau)$ 
    retorna  $\frac{[\kappa_{vol}(\hat{\tau})]^2}{\kappa_{area}(\hat{\tau})} + \frac{[\kappa_{vol}(\hat{\tau}')]^2}{\kappa_{area}(\hat{\tau}')} - \frac{[\kappa_{vol}(\hat{\tau}) + \kappa_{vol}(\hat{\tau}')]^2}{\kappa_{area}(\hat{\tau}) + \kappa_{area}(\hat{\tau}')} \times \frac{1}{\kappa_{perímetro}(\tau)}$ 
fim

função computa-atributo-funcional( $\mathcal{T}$ )
    pre-processamento( $\mathcal{T}$ )
     $upper \leftarrow -\infty$ 
    para todo  $\tau \in \{\mathcal{T} \setminus Raiz(\mathcal{T})\}$  faça
         $G_\nabla(\tau) \leftarrow \frac{S_\nabla(\tau)}{\kappa_{perímetro}(\tau)}$ 
         $\kappa_{ms}(\tau) \leftarrow \text{computa-energia-funcional}(\tau)$ 
         $upper \leftarrow \max\{upper, G_\nabla(\tau)\}$ 
    fim
    inicia( $Q, upper$ )
    para todo  $\tau \in \{\mathcal{T} \setminus Raiz(\mathcal{T})\}$  faça
        | enqueue( $Q, \tau, G_\nabla(\tau)$ )
    fim
    enquanto  $Q$  não está vazia faça
        |  $\tau \leftarrow \text{dequeue}(Q, \text{mínimo})$ 
        |  $\tau' \leftarrow Pai(\tau)$ 
        |  $\kappa_{ms}(\tau) \leftarrow \max\{\text{computa-energia-funcional}(\tau), \kappa_{ms}(\tau)\}$ 
        |  $\mathcal{T} = Remove(\mathcal{T}, \tau)$ 
        |  $\kappa_{area}(\hat{\tau}') \leftarrow \kappa_{area}(\hat{\tau}') + \kappa_{area}(\hat{\tau})$ 
        |  $\kappa_{vol}(\hat{\tau}') \leftarrow \kappa_{vol}(\hat{\tau}') + \kappa_{vol}(\hat{\tau})$ 
    fim
    retorna  $\kappa_{ms}$ 
fim

```

RESULTADOS E EXPERIMENTOS

6.1 DETECÇÃO DE PLANTAS

Reconhecimento de plantas é uma tarefa muito importante na área de visão computacional aplicada a agricultura (SMEDT; BILLIAUWS; GOEDEMÉ, 2011). Existem diversas motivações para estudos de visão computacional aplicada nessa área, pois plantas são um importante bem de consumo na sociedade e estão sujeitas a uma série de problemas patológicos (BARBEDO, 2013). Contudo, no contexto de visão computacional a tarefa de detecção de plantas é considerada difícil, pois plantas possuem uma morfologia complexa (mudam de cor e de forma durante o ciclo de crescimento), além do que as condições do ambiente influenciam no resultado (MINERVINI et al., 2016). Na Figura 6.1 é apresentado um exemplo que destaca as mudanças morfológicas que plantas da espécie *Arabidopsis* passaram durante algumas semanas de aquisição.

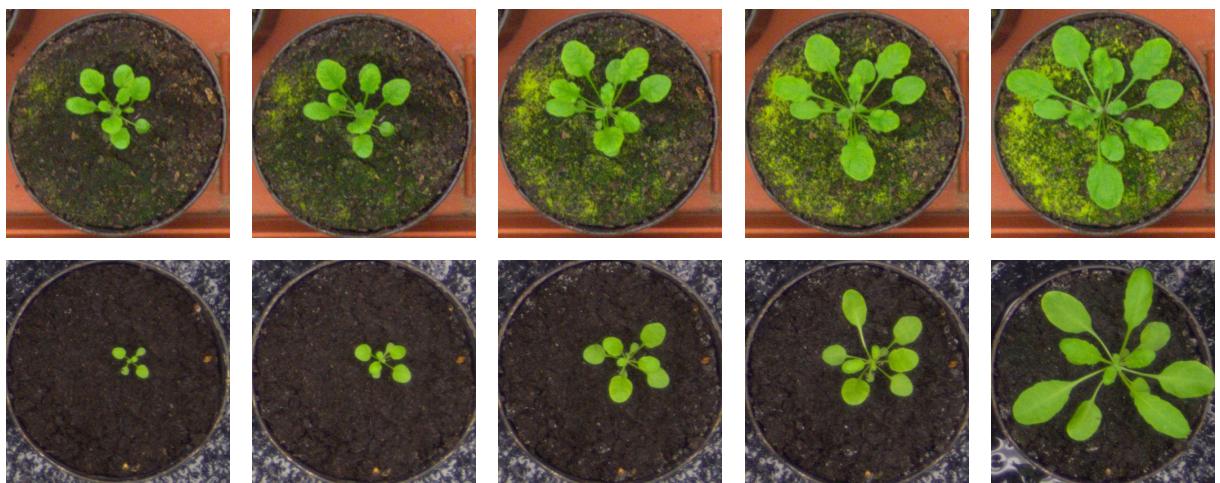


Figura 6.1: Evolução de plantas da espécie *Arabidopsis* analisadas durante algumas semanas. Note as diferenças encontradas na morfologia da planta durante seu ciclo de crescimento no período de aquisição das imagens.

Fonte : Minervini et al. (2016) (Adaptado pelo autor).

Uma importante base de imagens de plantas disponibilizada na literatura é a *Leaf Phenotyping dataset* (MINERVINI et al., 2016). Ela é composta de duas espécies de plantas: *Arabidopsis* e *Tobacco*. Além do mais, a base oferece imagens e dados anotados por especialistas para tarefas de diversas áreas de visão computacional e *machine learning*, tais como: detecção e segmentação de plantas, detecção e segmentação de folhas, detecção de contornos e reconhecimento de agentes mutantes. Nesse contexto, aplicamos nossa abordagem em uma tarefa específica de visão computacional provida pela base chamada de *bounding box detection* que é explicada na próxima seção.

6.1.1 BOUNDING BOX DETECTION

O conceito de *bounding box detection* é sem dúvida um dos mais simples e eficazes na detecção de objetos sendo utilizado como uma etapa de pré-processamento em muitas abordagens de visão computacional (LEMPITSKY et al., 2009). Em poucas palavras, o *bounding box* é o menor retângulo que contém um objeto de interesse na imagem. Detectar *bounding boxes* em uma imagem em geral costuma reduzir drasticamente o espaço de busca para uma técnica de segmentação, pois com a região do objeto detectada a tarefa de segmentação tende a ser menos onerosa. Na Figura 6.2 é apresentado um exemplo de imagem da base de imagens de plantas *Leaf Phenotyping dataset* com os *bounding boxes* detectados por nossa abordagem.



Figura 6.2: Exemplo de uma imagem da base de imagens de plantas *Leaf Phenotyping dataset* com os *bounding boxes* detectados por nossa abordagem.

Fonte : Minervini et al. (2016) (Adaptado pelo autor).

6.1.2 ABORDAGEM PROPOSTA

Algumas abordagens propostas para detecção de plantas encontradas na literatura utilizam o canal *green* f_g de uma dada imagem multibanda f_{rgb} , pois é o canal que costuma apresentar maior contraste nas regiões de plantas (BARBEDO, 2013; GIUFFRIDA; MINERVINI; TSAFTARIS, 2015). Na Figura 6.3 é apresentado um exemplo que mostra a diferença de contraste entre os três canais (RGB) extraídos de uma imagem de planta.

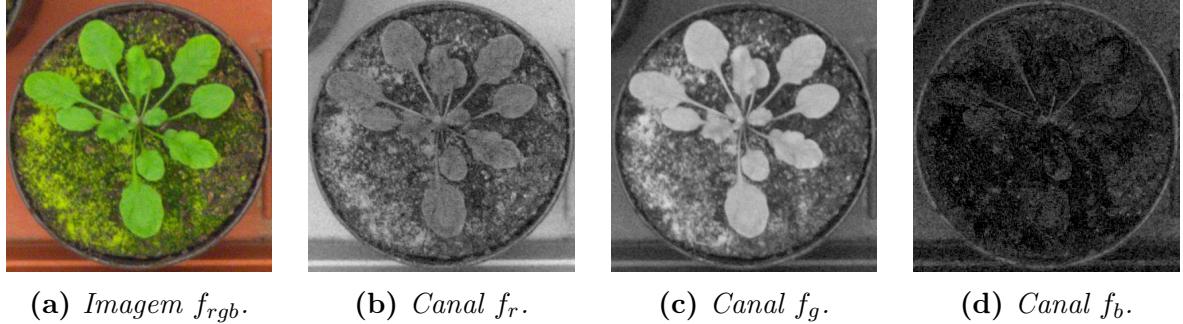


Figura 6.3: Exemplo ilustrando o contraste dos canais de uma imagem de planta f_{rgb} , note que o canal f_g apresenta maior contraste nas regiões da planta.

Fonte : Minervini et al. (2016) (Adaptado pelo autor).

Com a informação conhecida de maior contraste do canal *green*, optamos por escolher a *max-tree* (as regiões de plantas vão se concentrar próximas as folhas da árvore) e então o operador último *leveling* UAO, vale ressaltar que, outros operadores últimos *levelings* poderiam ter sido explorados. Assim, descrevem-se os passos da nossa abordagem:

1. Extrai-se o canal f_g da imagem de entrada f_{rgb} e então computa-se uma árvore morfológica $\mathcal{T}_{f_g} = (\mathcal{U}(f_g), \subseteq)$.
2. Na árvore \mathcal{T}_{f_g} é computado ou, o atributo funcional κ_{ms} ou o atributo MSER κ_{mser} . Para obter melhores resultados optou-se por construir uma estratégia obtida pela combinação de ou, o atributo funcional ou o atributo MSER com um atributo de área e um classificador knn utilizando as bandas RGB da imagem.
3. Com a estratégia de filtragem combinada ou, $\Omega^1 = \Omega_{\kappa_{ms}}$ e $\Omega_{\kappa_{área}}$ e Ω_{knn} ou $\Omega^2 = \Omega_{\kappa_{mser}}$ e $\Omega_{\kappa_{área}}$ e Ω_{knn} e a árvore \mathcal{T}_{f_g} são extraídos os resíduos pela computação do operador último *leveling* UAO e uma imagem de saída binarizada f_{ule} é gerada.
4. Algumas regiões de plantas extraídas pelo operador UAO podem estar incompletas e isso pode vir a gerar falsos *bounding boxes*.
5. Para tratar o problema de regiões incompletas primeiramente computa-se da imagem f_{ule} uma árvore $\mathcal{T}_{f_{ule}} = (\mathcal{U}(f_{ule}), \subseteq)$.
6. Para juntar as regiões incompletas, exploram-se os nós da árvore $\mathcal{T}_{f_{ule}}$ e aplica-se uma técnica de pós-processamento para junta-los, isto é, para quaisquer dois nós $\tau_1, \tau_2 \in \mathcal{T}_{f_{ule}}$, se $d(\text{centro}(\tau_1), \text{centro}(\tau_2)) < r_{join}$ então é feita uma operação de junção de τ_1 e τ_2 , onde d é a distância euclidiana, $r_{join} \in \mathbb{N}$ é um valor de *threshold* e *centro* é uma função que computa o centro de um nó.
7. Finalmente com as regiões devidamente definidas computam-se os *bounding boxes targets*.

Na Figura 6.4 é apresentado por meio de um diagrama (numerado em vermelho) os passos da nossa abordagem aplicada a uma parte de uma imagem de plantas da base *Leaf Phenotyping dataset*. Para facilitar a visualização os *bounding boxes* foram desenhados na imagem de entrada. Além disso, para cada imagem processada é fornecido um arquivo *csv*¹ contendo os *bounding boxes targets* para computar algumas métricas de avaliação que são detalhadas na próxima seção. Na Figura 6.5 é apresentado um exemplo que compara dois resultados da nossa abordagem (caso bom e ruim).

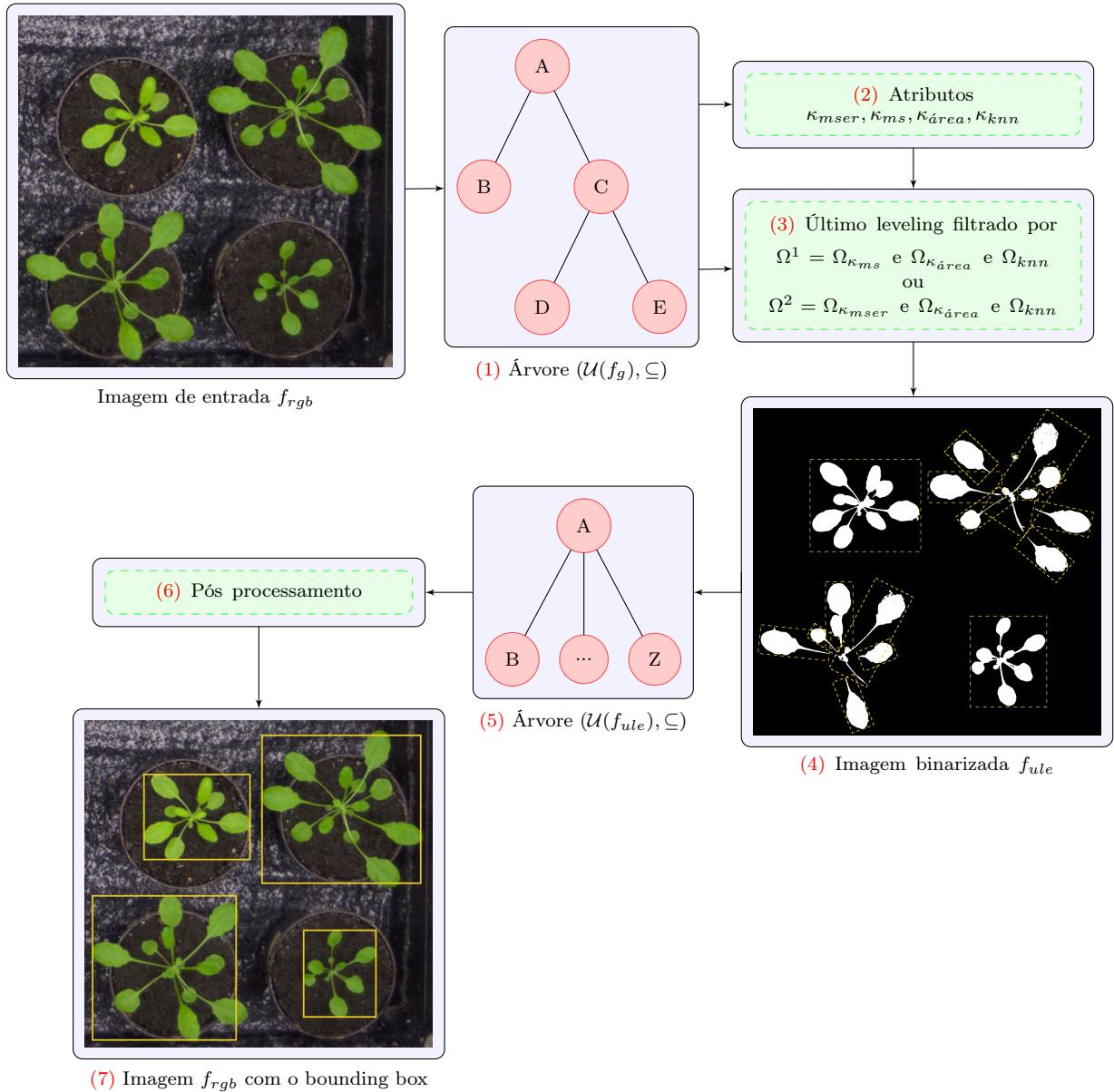


Figura 6.4: Representação da abordagem. A imagem f_g é a imagem do canal green e a imagem f_{ule} é a imagem de saída gerada pelo operador último leveling baseado nas estratégias combinadas $\Omega^1 = \Omega_{\kappa_{ms}} \text{ e } \Omega_{\kappa_{área}} \text{ e } \Omega_{knn}$ ou $\Omega^2 = \Omega_{\kappa_{mser}} \text{ e } \Omega_{\kappa_{área}} \text{ e } \Omega_{knn}$.

¹Csv é um formato de arquivo de texto regulamentado pelo RFC 4180.

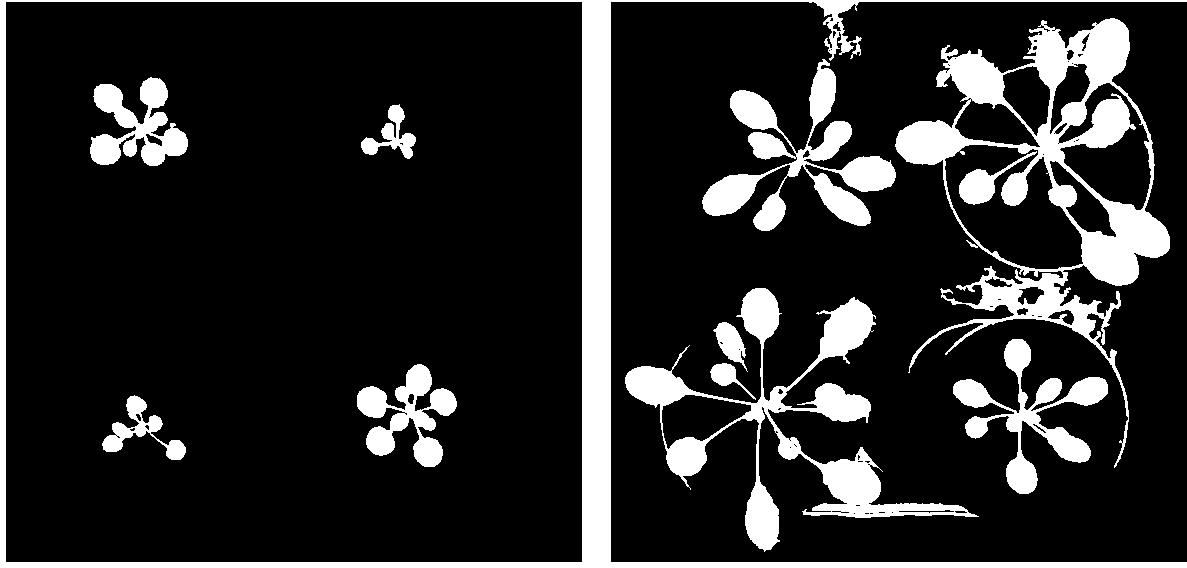
(a) *Bom resultado.*(b) *Resultado ruim.*

Figura 6.5: Comparação entre duas imagens de saída da nossa abordagem com o atributo κ_{ms} . O caso (a) apresenta um bom resultado enquanto o caso (b) mostra um caso em que a abordagem não deu bons resultados. A imagem que apresentou resultado ruim é da sub-base Ara2013-Rpi, que é a sub-base que contém imagens de qualidade inferior em comparação às demais.

6.1.3 RESULTADOS

A base de imagens *Leaf Phenotyping dataset* é composta por diversas imagens de plantas das espécies *Arabidopsis* e *Tobacco* distribuídas entre tarefas de visão computacional e de *machine learning*. Assim, quanto a tarefa de *bounding box detection* a base de imagens de plantas *Leaf Phenotyping dataset* é subdividida em três sub-bases: Ara2012, Ara2013-Canon e Ara2013-Rpi, totalizando 70 imagens com resoluções entre 1647x1158 e 3108x2324. Para cada imagem é também fornecido um documento *csv* contendo os *bounding boxes* anotados pelos especialistas.

Para avaliar o desempenho da nossa abordagem escolhemos utilizar algumas métricas práticas por Minervini et al. (2016) que inclusive fornece uma solução em *Matlab*² com todas as rotinas implementadas. A primeira delas é uma medida de acurácia chamada *Symmetric Best Dice* (SBD). Um *bounding box* é uma grade bidimensional $B_{wh} = \{0, 1, \dots, w\} \times \{0, 1, \dots, h\}$. Denota-se por $B_{wh}(p)$ um *bounding box* transladado por um *pixel* $p \in \mathcal{D}$, isto é, $B_{wh}(p) = \{0 + p_x, 1 + p_x, \dots, w + p_x\} \times \{0 + p_y, 1 + p_y, \dots, h + p_y\}$. O conjunto de todos os *bounding boxes* sobre \mathcal{D} é denotado por $\mathbb{B}(\mathcal{D}) = \{B_{wh} : B_{wh} \text{ é definido sobre } \mathcal{D}\}$. Dessa forma, sendo $\mathcal{B}_1, \mathcal{B}_2 \subseteq \mathbb{B}(\mathcal{D})$ dois subcon-

²Matlab é um *software* comercial de alta performance voltado para cálculo numérico.

juntos de *bounding boxes*, define-se a métrica *BD* da seguinte forma:

$$BD(\mathcal{B}_1, \mathcal{B}_2) = \frac{1}{|\mathcal{B}_1|} \sum_{B_i \in \mathcal{B}_1} \max_{B_j \in \mathcal{B}_2} \left\{ \frac{2 \times |B_i \cap B_j|}{|B_i| + |B_j|} \right\}. \quad (6.1)$$

Considere $\mathcal{B}_r \subseteq \mathbb{B}(\mathcal{D})$ como o conjunto dos *bounding boxes* dos resultados conhecidos (anotados pelos especialistas) de uma imagem f , e $\mathcal{B}_t \subseteq \mathbb{B}(\mathcal{D})$ o conjunto dos *bounding boxes targets* computados de f , assim define-se a métrica *SBD* da seguinte forma:

$$SBD(\mathcal{B}_r, \mathcal{B}_t) = \min\{BD(\mathcal{B}_r, \mathcal{B}_t), BD(\mathcal{B}_t, \mathcal{B}_r)\}. \quad (6.2)$$

Uma outra medida utilizada foi a *Difference in Count* (*DiC*) que identifica o número correto de *bounding boxes targets* que deveriam ser computados em comparação aos anotados pelos especialistas. Dessa forma, para os conjuntos \mathcal{B}_r e \mathcal{B}_t computados de uma imagem f , define-se a medida *DiC* da seguinte forma:

$$DiC = |\mathcal{B}_r| - |\mathcal{B}_t|. \quad (6.3)$$

Adicionalmente, calculou-se também o valor absoluto de *DiC*, isto é, $|DiC|$. É importante ressaltar que, as métricas foram computadas para todas as 70 imagens das bases Ara2012, Ara2013-Canon e Ara2013-Rpi. Assim, a partir de um conjunto de parâmetros obteve-se os valores de *thresholds* ótimos, sendo eles: $\varepsilon_{kms} = 190$ (atributo funcional), $\varepsilon_{mser} = 11$ (atributo MSER), $\varepsilon_{área} = 50$ (atributo de área) e $\varepsilon_{knn} = 100$ (atributo baseado no classificador *knn*). Para o parâmetro r_{join} obteve-se melhores resultados utilizando valores diferentes para as sub-bases: $r_{join} = 250$ (Ara2012 e Ara2013-Canon) e $r_{join} = 100$ (Ara2013-Rpi). Na Tabela 6.1 são apresentados os resultados para as medidas *SBD*, *DiC* e $|DiC|$ para os melhores parâmetros obtidos.

Tabela 6.1: Resultados obtidos na detecção de *bounding box* em imagens de plantas com a abordagem proposta neste trabalho. Em azul são os resultados do κ_{ms} e em vermelho os resultados do κ_{mser} .

Bases	<i>SBD</i> [%]	<i>DiC</i>	$ DiC $
Ara2012	92.0 (± 2.8)	0.20 (± 0.4)	0.20 (± 0.4)
	91.9 (± 4.1)	0.20 (± 0.4)	0.20 (± 0.4)
Ara2013-Canon	87.5 (± 2.7)	-0.10 (± 0.5)	0.30 (± 0.4)
	89.4 (± 1.9)	-0.10 (± 0.3)	0.10 (± 0.3)
Ara2013-Rpi	80.3 (± 3.9)	0.10 (± 0.8)	0.40 (± 0.6)
	83.2 (± 3.7)	0.20 (± 0.6)	0.40 (± 0.6)
Todas	85.8 (± 5.7)	0.00 (± 0.6)	0.30 (± 0.5)
	87.6 (± 4.8)	0.10 (± 0.5)	0.20 (± 0.5)

Para obter resultados comparativos nós computamos um atributo de filtragem para os últimos *levelings* baseado em uma técnica chamada Regiões Extremais Maximamente Estáveis (MSER) que foi introduzida por Matas et al. (2004). Os resultados apresentados na Tabela 6.1 revelam que o MSER obteve índices melhores a respeito da medida SBD, contudo são necessários mais experimentos, principalmente em imagens coloridas. É importante ressaltar que, na literatura não foram encontrados resultados para serem comparados no que diz respeito a tarefa *bounding box detection*. Contudo, o objetivo principal dessa aplicação é demonstrar a robustez da nova estratégia para filtrar resíduos dos últimos *levelings* baseada no atributo funcional κ_{ms} . Assim, na Figura 6.6 é apresentada uma curva com vários valores de *threshold* para a estratégia $\Omega_{\kappa_{ms}}$, é possível notar um ganho significativo de aproximadamente 4% na acurácia da nossa abordagem. Além do mais, o comportamento da curva indica que o atributo é robusto no sentido que o resultado obtido com $\varepsilon_{\kappa_{ms}} = 190$ é um máximo local ótimo.

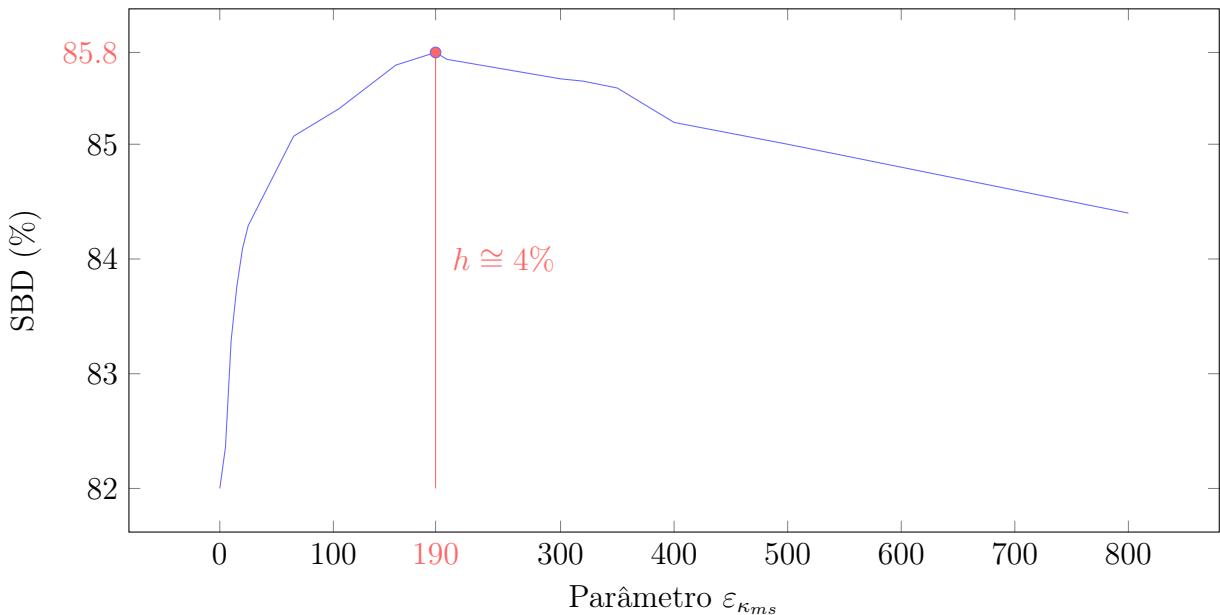


Figura 6.6: Resultados obtidos pela estratégia de energia $\Omega_{\kappa_{ms}}$ através do conjunto de parâmetros. Note na linha vermelha o ganho de aproximadamente 4% obtido com a abordagem.

6.1.4 CONCLUSÃO

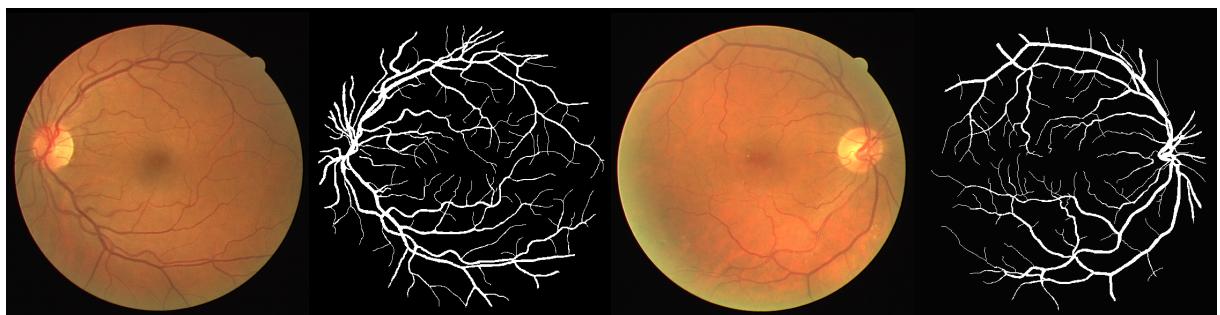
Os resultados apresentados na aplicação de detecção de plantas mostram que a filtragem residual dos últimos *levelings* baseada no atributo funcional é robusta. Cada imagem levou em média 12 segundos para ser processada em um computador com processador *i7* e 16GB de memória RAM. Em trabalhos futuros, recomenda-se explorar as seguintes áreas: computação de últimos *levelings* e do atributo funcional κ_{ms} em imagens coloridas; estudar outros atributos de energia como os *snakes* (XU; GéRAUD; NAJMAN, 2012).

6.2 ANÁLISE DE IMAGENS DE RETINA

Um dos maiores problemas de saúde pública mundial está relacionado as patologias oculares que ocorrem principalmente nas pessoas mais idosas (SRINIDHI; APARNA; RAJAN, 2017). As patologias são variadas, tais como: glaucoma, retinopatia diabética e degeneração macular, que podem levar em casos mais graves a completa cegueira (FELIPE-RIVERON; GARCIA-GUIMERAS, 2006). Nesse sentido, a segmentação de vasos sanguíneos é um indicador importante para prevenir um grande número dessas patologias, pois deles podem ser extraídas algumas medidas de diagnóstico, tais como: tamanhos dos vasos, cor, reflexividade, sinuosidade e outros (HOOVER; KOUZNETSOVA; GOLDBAUM, 2000; STAAL et al., 2004).

6.2.1 SEGMENTAÇÃO DE VASOS SANGUÍNEOS

A segmentação manual de vasos de retina é uma tarefa muito onerosa devido as variações de tamanho dos vasos além de ruídos na imagem, e por consequência o especialista está sujeito a erros durante o processo de delineação das regiões (FRAZ et al., 2012). Na Figura 6.7 são apresentados exemplos de imagens de fundos de olhos e seus respectivos vasos sanguíneos segmentados.



(a) Imagem de fundo do olho. (b) Vasos sanguíneos segmentados. (c) Imagem de fundo do olho. (d) Vasos sanguíneos segmentados.

Figura 6.7: Exemplos de imagens de fundos de olhos e seus respectivos vasos sanguíneos segmentados.

Fonte : Base DRIVE (Adaptado pelo autor).

O problema de segmentação automática de imagens de vasos sanguíneos de retina vem sendo estudado na literatura há bastante tempo e diversas abordagens têm surgido (veja as surveys: Fraz et al. (2012) e Srinidhi, Aparna e Rajan (2017)). Essas abordagens têm sido testadas em bancos de imagens públicos. Assim, nas próximas seções são apresentados os passos e resultados das abordagens para filtragem dos últimos *levelings* baseadas nas funções de energia revisadas no Capítulo 5 aplicadas na tarefa de segmentação automática de vasos sanguíneos em imagens de retina.

6.2.2 ABORDAGEM PROPOSTA

Grande parte dos métodos apresentados na literatura utilizam apenas o canal f_g da imagem da retina, pois é o que apresenta maior contraste dos vasos sanguíneos em relação aos demais canais (ZANA; KLEIN, 2001; STAAL et al., 2004; YANG; HUANG; RAO, 2008; NGUYEN et al., 2013; ZHANG et al., 2015; LISKOWSKI; KRAWIEC, 2016). Na Figura 6.8 é apresentado um exemplo de uma imagem da retina que demonstra a diferença de contraste entre os canais (RGB) extraídos da imagem.

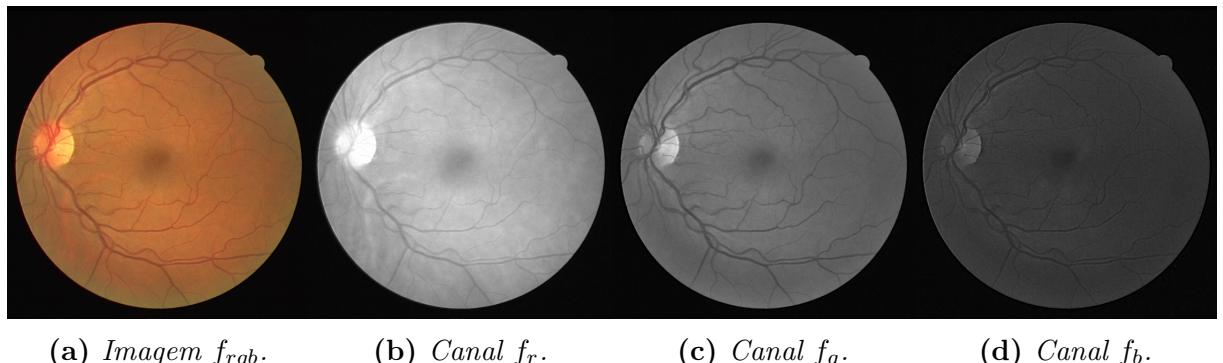


Figura 6.8: Exemplo ilustrando a diferença de contraste entre os canais de uma imagem de retina f_{rgb} , note a ênfase de contraste nas regiões de vasos sanguíneos da banda f_g .

Fonte : Base DRIVE (Adaptado pelo autor).

Conforme já introduzimos anteriormente, muitos problemas de visão computacional são resolvidos a partir de etapas. No que diz respeito ao pré-processamento de imagens de retina, a maioria das técnicas buscam aumentar o contraste dos vasos sanguíneos para depois então obter uma segmentação, pois diversas regiões do canal f_g não são de vasos sanguíneos. Nesse sentido, uma abordagem muito utilizada é baseada no uso de operadores morfológicos (ZANA; KLEIN, 2001; YANG; HUANG; RAO, 2008; HASSAN et al., 2015; ALVES et al., 2016).

Grande parte dos operadores morfológicos podem ser definidos com base em um conjunto $\mathcal{B} \subseteq \mathbb{Z}^2$ chamado EE. Esses operadores computam o valor de um *pixel* $p \in \mathcal{D}$ utilizando os valores de *pixels* vizinhos de p . A relação de adjacência definida por \mathcal{B} é dada da seguinte forma: p é vizinho de q segundo \mathcal{B} se, e somente se, p e $q \in \mathcal{D}$ e $q \in (\mathcal{B} + p) = \{b + p : b \in \mathcal{B}\}$. Assim, denota-se por $\mathcal{A}_{\mathcal{B}}$ uma relação de adjacência dada por um EE \mathcal{B} .

Dado o exposto, vamos definir alguns operadores morfológicos necessários para a etapa de pré-processamento. Os primeiros e também os principais são: a *erosão* e a *dilatação* (SERRA, 1983). De acordo com Soille (2003) podemos definir as operações de erosão e dilatação, denotadas por $\varepsilon_{\mathcal{B}}$ e $\delta_{\mathcal{B}}$, para uma dada imagem f e um EE \mathcal{B} , como os filtros

do mínimo e do máximo, isto é:

$$\forall p \in \mathcal{D}, [\varepsilon_{\mathcal{B}}(f)](p) = \min_{q \in \mathcal{A}_{\mathcal{B}}(p)} f(q) \text{ e } [\delta_{\mathcal{B}}(f)](p) = \max_{q \in \mathcal{A}_{\mathcal{B}}(p)} f(q). \quad (6.4)$$

Alguns operadores morfológicos podem ser obtidos por combinações de outros operadores. Nesse sentido, utilizando um mesmo EE, a erosão e a dilatação podem ser combinadas para construir outros dois operadores: *abertura* e *fechamento*. Assim, definem-se os operadores abertura e fechamento, denotados por $\gamma_{\mathcal{B}}$ e $\varphi_{\mathcal{B}}$, para uma dada imagem f e um EE \mathcal{B} como:

$$\gamma_{\mathcal{B}}(f) = \varepsilon_{\mathcal{B}}(\delta_{\mathcal{B}}(f)) \text{ e } \varphi_{\mathcal{B}}(f) = \delta_{\mathcal{B}}(\varepsilon_{\mathcal{B}}(f)). \quad (6.5)$$

Outros operadores morfológicos podem ser obtidos por combinações de uma imagem chamada de marcadora, um EE e uma outra imagem chamada de máscara. Um exemplo deste tipo de operador é a dilatação geodésica. A dilatação geodésica, denotada por $\delta_{\mathcal{B}}^1(f, g)$ de uma imagem marcadora f em respeito de uma imagem máscara g e um EE \mathcal{B} , sendo $f \leq g$ e $\mathcal{D}_f = \mathcal{D}_g$ é definida da seguinte forma:

$$\delta_{\mathcal{B}}^1(f, g) = \delta_{\mathcal{B}}(f) \wedge g. \quad (6.6)$$

Iterando o operador dilatação geodésica n vezes até obter idempotência, isto é, $\delta_{\mathcal{B}}^n(f, g) = \delta_{\mathcal{B}}(\delta_{\mathcal{B}}^n(f, g)) \wedge g$ é obtida uma operação chamada *reconstrução morfológica*. Em especial, essa reconstrução morfológica é chamada de *reconstrução por dilatação*. Assim, seja g uma imagem máscara, f uma imagem marcadora e \mathcal{B} um EE, onde $f \leq g$ e $\mathcal{D}_f = \mathcal{D}_g$, a reconstrução por dilatação é definida como a dilatação geodésica de f em respeito a g até atingir a idempotência, isto é:

$$Rec_{\mathcal{B}}^{\delta}(f, g) = \delta_{\mathcal{B}}^n(f, g), \quad (6.7)$$

onde, $\delta_{\mathcal{B}}^n(f, g) = \delta_{\mathcal{B}}(\delta_{\mathcal{B}}^n(f, g)) \wedge g$.

Dadas as definições, a abordagem de pré-processamento foi construída a partir do trabalho de [Zana e Klein \(2001\)](#). A ideia central é baseada em uma filtragem morfológica feita em mais de uma direção para obter uma imagem com maior quantidade de vasos sanguíneos e menos ruído. Assim, para uma dada imagem do canal verde f_g é feita uma reconstrução por dilatação da máxima abertura de uma sequência de n aberturas ($\gamma_{\mathcal{B}_{\theta_0}}(f), \gamma_{\mathcal{B}_{\theta_1}}(f), \dots, \gamma_{\mathcal{B}_{\theta_n}}(f)$) construída sobre uma sequência de EE's horizontais ($\mathcal{B}_{\theta_0}, \mathcal{B}_{\theta_1}, \dots, \mathcal{B}_{\theta_n}$) com altura 1 e largura 7 rotacionados por um ângulo $\theta_i, i = 0, \dots, n$, isto é:

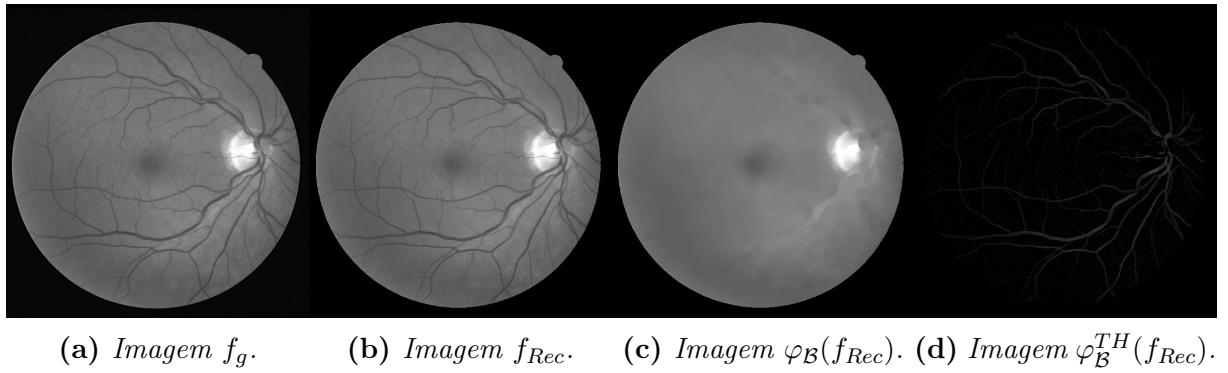
$$f_{Rec} = Rec_{\mathcal{B}}^{\delta}(f_g, \max\{\gamma_{\mathcal{B}_{\theta_0}}(f), \gamma_{\mathcal{B}_{\theta_1}}(f), \dots, \gamma_{\mathcal{B}_{\theta_n}}(f)\}). \quad (6.8)$$

Os melhores resultados foram obtidos com uma sequência de 12 ângulos, isto é, $(\theta_0 = 0^\circ, \theta_1 = \theta_0 + 30^\circ, \theta_2 = \theta_1 + 30^\circ, \dots, \theta_{11} = 330^\circ)$. Vale ressaltar que, os índices podem ser

utilizados para obter um EE rotacionado. Assim, rotaciona-se um dado elemento $b \in \mathcal{B}_i$ para um índice $i = 0, \dots, 11$, da seguinte forma:

$$(b_x \times \cos\left(\frac{2\pi \times i}{12}\right) - b_y \times \sin\left(\frac{2\pi \times i}{12}\right), b_y \times \cos\left(\frac{2\pi \times i}{12}\right) + b_x \times \sin\left(\frac{2\pi \times i}{12}\right)), \quad (6.9)$$

onde \sin e \cos são respectivamente as funções seno e cosseno. Note que, a implementação baseada em índices é mais eficiente, pois pode ser previamente computada e armazenada em um vetor. Finalmente, na imagem f_{Rec} aplica-se o operador residual fechamento *top-hat* $\varphi_{\mathcal{B}}^{TH}(f_{Rec}) = \varphi_{\mathcal{B}}(f_{Rec}) - f_{Rec}$ utilizando um EE \mathcal{B} de disco com raio 6. Na Figura 6.9 é apresentado um exemplo da aplicação da etapa de pre-processamento para imagens de retina descrita nesta seção.



(a) *Imagen* f_g . (b) *Imagen* f_{Rec} . (c) *Imagen* $\varphi_{\mathcal{B}}(f_{Rec})$. (d) *Imagen* $\varphi_{\mathcal{B}}^{TH}(f_{Rec})$.

Figura 6.9: Passos da abordagem de pre-processamento aplicada em uma imagem de retina.
Fonte : Base DRIVE (Adaptado pelo autor).

A filtragem obtida pela etapa de pré-processamento remove muitas regiões indesejáveis e preserva os vasos sanguíneos, seguindo com a nossa abordagem deve-se agora extrair e classificar os resíduos. A partir da imagem $\varphi_{\mathcal{B}}^{TH}(f_{Rec})$ computa-se uma *max-tree* e então o operador último *leveling* UAO. Vale ressaltar que, como citado na aplicação anterior outros operadores últimos *levelings* poderiam ter sido explorados. Assim, descrevem-se os passos da abordagem proposta:

1. A partir da imagem *top-hat* $\varphi_{\mathcal{B}}^{TH}(f_{Rec})$ computa-se uma árvore morfológica $\mathcal{T}_{\varphi_{\mathcal{B}}^{TH}(f_{Rec})} = (\mathcal{U}(\varphi_{\mathcal{B}}^{TH}(f_{Rec})), \subseteq)$.
2. Na árvore $\mathcal{T}_{\varphi_{\mathcal{B}}^{TH}(f_{Rec})}$ computa-se ou, o atributo variacional $\kappa_{\Delta E}$ ou o atributo funcional κ_{ms} . Em adição para obter melhores resultados, combinou-se os atributos de energia com um atributo de área.
3. Com as estratégias de filtragem combinadas ou, $\Omega^1 = \Omega_{\kappa_{\Delta E}}$ e $\Omega_{\kappa_{rea}}$ ou $\Omega^2 = \Omega_{\kappa_{ms}}$ e $\Omega_{\kappa_{rea}}$ e a árvore $\mathcal{T}_{\varphi_{\mathcal{B}}^{TH}(f_{Rec})}$ são extraídos os resíduos pela computação do operador último *leveling* UAO e uma imagem de saída binarizada f_{ule} é gerada.

Na Figura 6.10 é apresentado por meio de um diagrama os passos da nossa abordagem que foi aplicada em duas bases públicas de imagens de retinas: DRIVE (HOOVER;

KOUZNETSOVA; GOLDBAUM, 2000) e STARE (STAAL et al., 2004). Em adição, a partir da imagem de saída f_{ule} foram computadas algumas métricas de avaliação que são detalhadas na próxima seção. Na Figura 6.11 são apresentados alguns exemplos de imagens de saída da nossa abordagem.

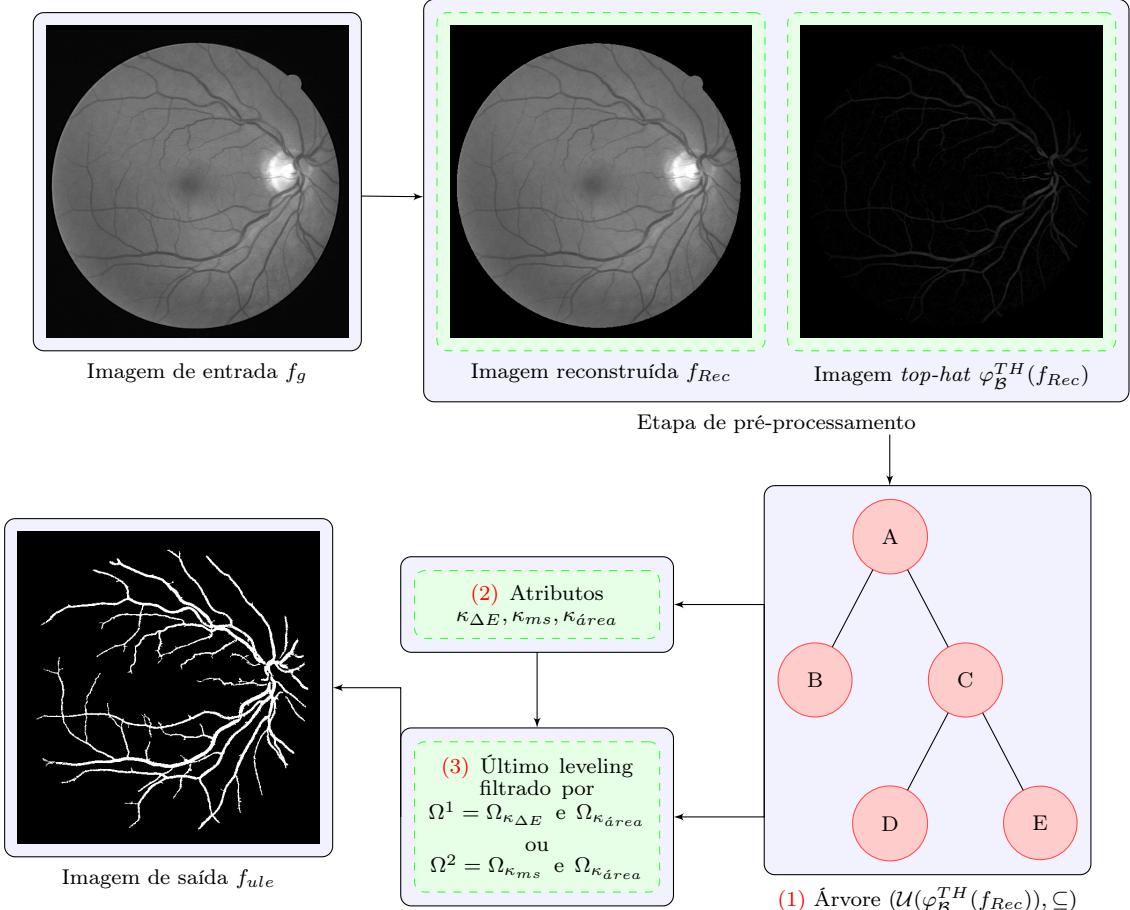


Figura 6.10: Representação da abordagem. A imagem f_g é a imagem do canal green e a imagem f_{ule} é a imagem de saída gerada pelo operador último leveling baseado nas estratégias combinadas $\Omega^1 = \Omega_{\kappa_{\Delta E}} \text{ e } \Omega_{\kappa_{área}}$ ou $\Omega^2 = \Omega_{\kappa_{ms}} \text{ e } \Omega_{\kappa_{área}}$.

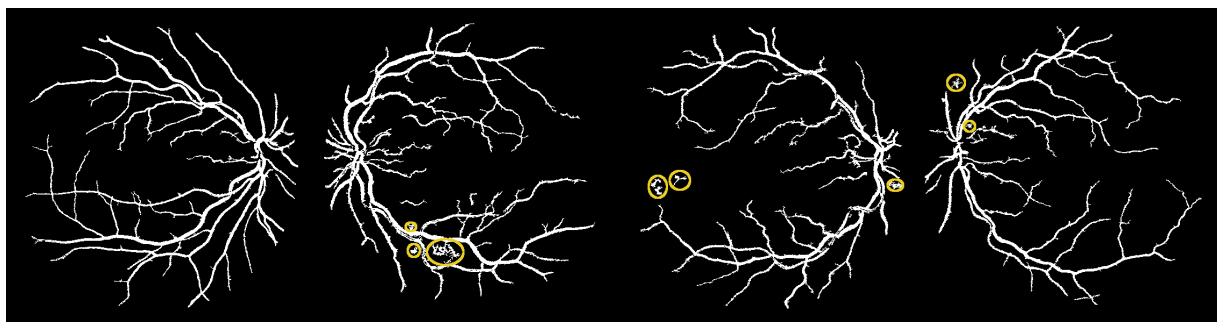


Figura 6.11: Comparação entre algumas imagens resultantes da nossa abordagem com κ_{ms} . No caso (a) os vasos foram bem segmentados e o resultado foi bom, enquanto no caso (b) são apresentadas imagens onde a saída foi ruidosa e os resultados ruins.

6.2.3 RESULTADOS

Atualmente na literatura grande parte dos trabalhos de segmentação de imagens de vasos sanguíneos são aplicados em duas bases de imagens de retina: *Digital Retinal Images for Vessel Extraction* (DRIVE) e *Structured Analysis of the Retina* (STARE) (SRINIDHI; APARNA; RAJAN, 2017). A base DRIVE é composta de 40 imagens RGB, sendo que 7 imagens possuem algum tipo de patologia. Em adição, todas foram escolhidas de forma aleatória e foram capturadas por uma câmera Canon CR5 3CCD com uma resolução de 768×584 pixels. Já o banco STARE é composto por 20 imagens RGB entre as quais 10 apresentam patologia. As imagens foram capturadas por uma câmera TopCon TRV-50 com uma resolução de 700×605 pixels. Em ambos os bancos são fornecidas imagens segmentadas manualmente por especialistas.

Para avaliar nossa abordagem, são empregadas algumas medidas de desempenho que são utilizadas por outros autores na literatura, são elas: sensibilidade, acurácia e especificidade (FRAZ et al., 2012; SRINIDHI; APARNA; RAJAN, 2017). A sensibilidade (SE) é uma medida que mede a taxa de verdadeiros positivos, já a especificidade (ES) mede a taxa de verdadeiros negativos e a acurácia (AC) mede a taxa de pixels classificados corretamente. Dessa forma, seja $\mathcal{X}_r \subseteq \mathcal{D}$ a imagem binária anotada por um especialista e $\mathcal{X}_t \subseteq \mathcal{D}$ a imagem binária resultante da nossa abordagem, assim definem-se as medidas de avaliação:

$$SE(\mathcal{X}_r, \mathcal{X}_t) = \frac{VP(\mathcal{X}_r, \mathcal{X}_t)}{VP(\mathcal{X}_r, \mathcal{X}_t) + FN(\mathcal{X}_r, \mathcal{X}_t)} \quad (6.10)$$

$$ES(\mathcal{X}_r, \mathcal{X}_t) = \frac{VN(\mathcal{X}_r, \mathcal{X}_t)}{VN(\mathcal{X}_r, \mathcal{X}_t) + FP(\mathcal{X}_r, \mathcal{X}_t)} \quad (6.11)$$

$$AC(\mathcal{X}_r, \mathcal{X}_t) = \frac{VP(\mathcal{X}_r, \mathcal{X}_t) + VN(\mathcal{X}_r, \mathcal{X}_t)}{|\mathcal{D}|} \quad (6.12)$$

onde $VP(\mathcal{X}_r, \mathcal{X}_t) = |\mathcal{X}_r \cap \mathcal{X}_t|$ é a quantidade de verdadeiros positivos, $VN(\mathcal{X}_r, \mathcal{X}_t) = |\mathcal{X}_r^c \cap \mathcal{X}_t^c|$ é a quantidade verdadeiros negativos, $FP(\mathcal{X}_r, \mathcal{X}_t) = |\mathcal{X}_r^c \cap \mathcal{X}_t|$ é a quantidade de falsos positivos e $FN(\mathcal{X}_r, \mathcal{X}_t) = |\mathcal{X}_r \cap \mathcal{X}_t^c|$ é a quantidade de falsos negativos.

Na Figura 6.12 são apresentados alguns resultados para a base DRIVE, como é possível perceber o atributo funcional apresenta imagens menos ruidosas, isto é, existem mais verdadeiros positivos e menos falsos positivos em comparação o funcional variacional. Já na Figura 6.13 nota-se um resultado oposto quando as estratégias foram aplicadas na base STARE, e de fato a acurácia na base DRIVE foi melhor no atributo funcional e na base STARE ocorreu um resultado melhor no funcional variacional. Nota-se que, apesar da abordagem de pré-processamento manter regiões de vasos sanguíneos, alguns deles que são muito finos são perdidos durante a etapa. A respeito dos *thresholds*, para a estratégia baseada no atributo funcional foram utilizados os seguintes valores: $\varepsilon_{\kappa_m} = 8$ e

$\varepsilon_{\kappa_{area}} = 100$, já na estratégia baseada no funcional variacional foram utilizados os seguintes valores: $\varepsilon_{\kappa_{\Delta E}}^{min} = -1400$, $\varepsilon_{\kappa_{\Delta E}}^{max} = 100$, $\nu = 0$ e $\varepsilon_{\kappa_{area}} = 500$.

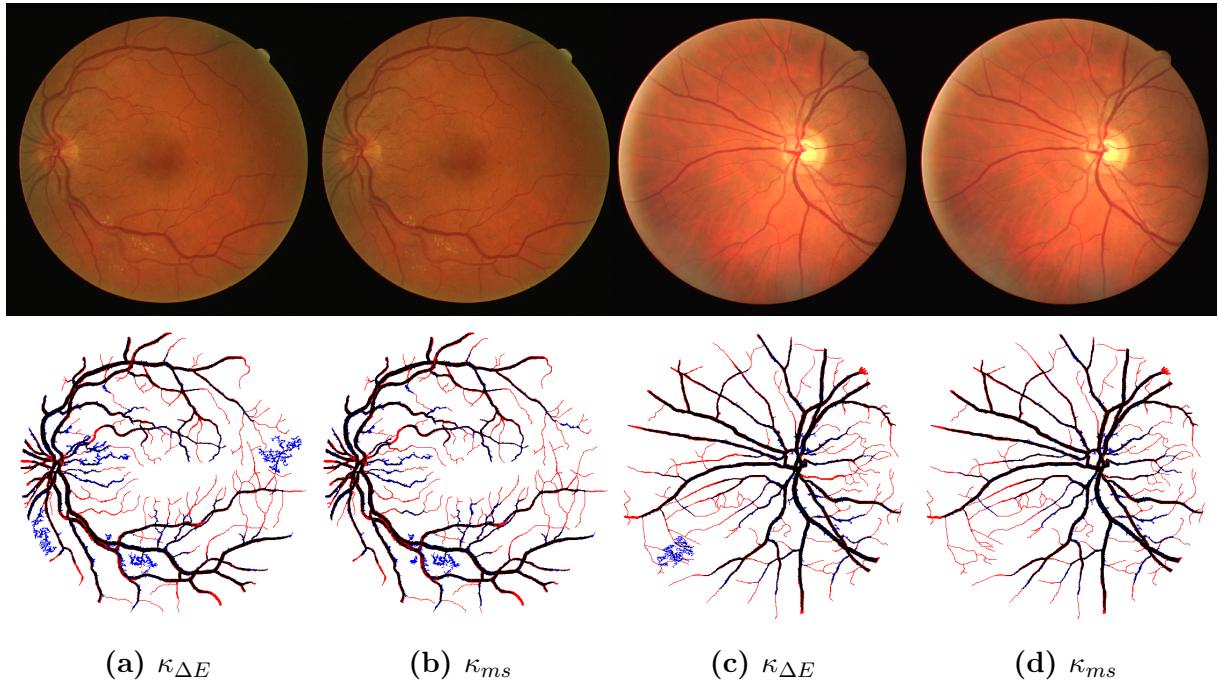


Figura 6.12: Resultados comparativos de segmentações de duas imagens da base DRIVE. Note que, as imagens segmentadas pelo atributo funcional apresentam menos ruído. Os pixels em preto são verdadeiros positivos, em branco os verdadeiros negativos, em azul os falsos positivos, e em vermelho os falsos negativos

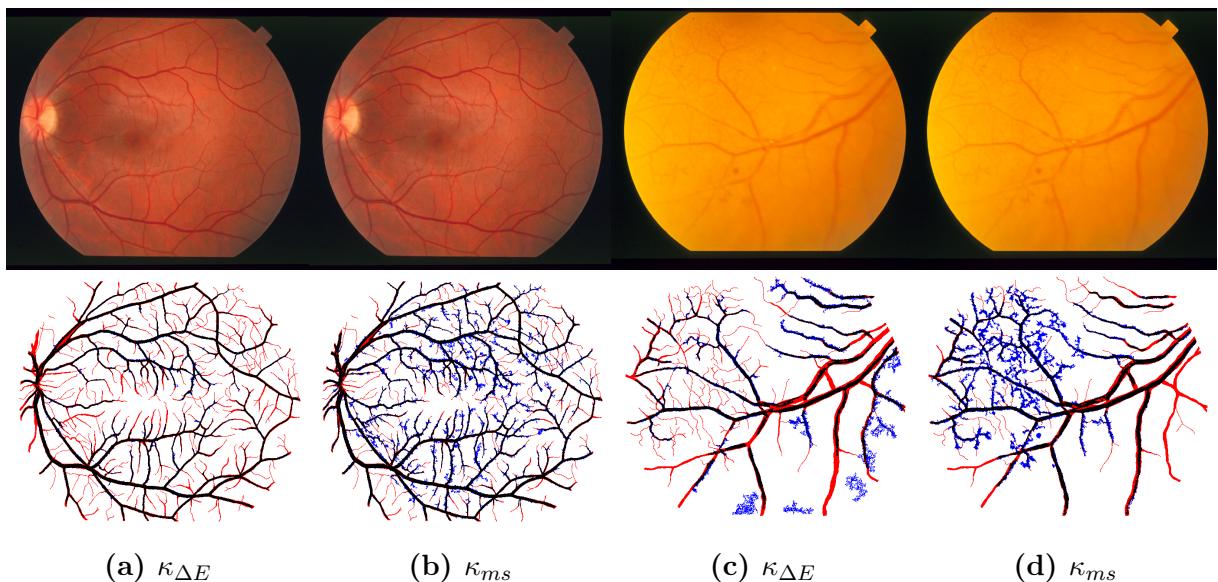


Figura 6.13: Resultados comparativos de segmentações de duas imagens da base STARE. Note que, as imagens segmentadas pelo funcional variacional apresentam menos ruído. Os pixels em preto são verdadeiros positivos, em branco os verdadeiros negativos, em azul os falsos positivos, e em vermelho os falsos negativos

Conforme introduzido, o problema de segmentação de retina é muito explorado na lite-

ratura e por consequência existe uma grande variação de métodos. Os resultados apresentados na Tabela 6.2 e 6.3 mostram que nossa abordagem apresenta resultados satisfatórios, visto que está dentre os melhores encontrados na literatura.

Tabela 6.2: Comparação de resultados de diferentes abordagens para segmentação de vasos sanguíneos em imagens da base DRIVE.

Método	Sensibilidade	Especificidade	Acurácia
Segundo o especialista	0.7761	0.9725	0.9473
Wang et al. (2015)	0.8173	0.9733	0.9767
Budai et al. (2013)	0.6440	0.9870	0.9572
Método proposto κ_{ms}	0.7004	0.9821	0.9449
Alves et al. (2016)	0.6902	0.9800	0.9429
Método proposto $\kappa_{\Delta E}$	0.6984	0.9768	0.9398
Yin, Adel e Bourennane (2013)	0.6522	0.9710	0.9267
Abdallah et al. (2015)	0.5879	-	0.9155

Tabela 6.3: Comparação de resultados de diferentes abordagens para segmentação de vasos sanguíneos em imagens da base STARE.

Método	Sensibilidade	Especificidade	Acurácia
Segundo o especialista	0.8949	0.9390	0.9354
Wang et al. (2015)	0.8104	0.9791	0.9813
Alves et al. (2016)	0.6835	0.9739	0.9460
Budai et al. (2013)	0.5800	0.9820	0.9386
Método proposto $\kappa_{\Delta E}$	0.7328	0.9578	0.9375
Método proposto κ_{ms}	0.7424	0.9546	0.9351
Mapayi, Viriri e Tapamo (2015)	0.5031	0.9567	0.9221

6.2.4 CONCLUSÃO

Esta seção apresentou duas abordagens para segmentação de vasos sanguíneos em imagens de retina. As estratégias baseadas em funções de energia obtiveram resultados satisfatórios em comparação aos métodos existentes na literatura. Vale ressaltar que, o tempo médio de processamento de uma imagem de retina dos bancos DRIVE ou STARE é em torno de 2 segundos em um computador com processador *i7* e 16GB de memória RAM. Futuramente, recomenda-se explorar outros métodos de pré-processamento como o CLAHE (GEETHARAMANI; BALASUBRAMANIAN, 2016) e o *multi line detector*

(NGUYEN et al., 2013), além de estudar outros atributos de energia como os *snakes* (XU; GéRAUD; NAJMAN, 2012).

CONCLUSÕES

Nesta dissertação foram apresentadas duas novas estratégias para filtragem de resíduos dos últimos *levelings*: o funcional variacional e o atributo funcional. Ambas as estratégias são baseadas nas chamadas funções de energia definidas e estudadas inicialmente por Mumford e Shah (1989). Os resultados apresentados no Capítulo 6 mostram dois âmbitos diferentes de problemas que as estratégias apresentadas aqui foram aplicadas e obtiveram sucesso. Dessa forma, as estratégias mostram uma opção robusta para filtrar resíduos numéricos extraídos pelos últimos *levelings*. Nesse contexto, foram apresentados neste trabalho: (ii) definições sobre: imagens, árvores morfológicas, operadores conexos, *levelings*, funções de energia e últimos *levelings*; (ii) revisão literária sobre funções de energia; (iii) algoritmos eficientes para computação de árvores morfológicas, funcional variacional e do atributo funcional e (iv) exemplos de aplicações distintas. Além disso, implementações dos algoritmos da funcional variacional e do atributo funcional em linguagem Java foram disponibilizadas como *plugins* do ImageJ.

7.1 TRABALHOS FUTUROS

Este trabalho obteve resultados e contribuições nos estudos de filtragem residual para os últimos *levelings*. Como trabalhos futuros, apontamos os seguintes assuntos para serem explorados:

- Estudar novas formas de minimizar o funcional de Mumford-Shah utilizando árvores morfológicas e técnicas meta-heurísticas, como por exemplo: Algoritmos Genéticos.
- Estudar e definir novas primitivas para computação de operadores últimos *levelings* a partir da funcional de Mumford-Shah, levando em conta o parâmetro de escala ν e estudos sobre hierarquias de partições apresentados em Guigues, Cocquerez e Men (2006), Kiran e Serra (2014) e Xu, Géraud e Najman (2016).
- Construir novas estratégias baseadas em funções de energia, estendendo o leque de opções a partir dos *snakes* de Kass, Witkin e Terzopoulos (1988) e outras funções de energias como: Geman e Geman (1984) e Leclerc (1989).
- Estender os operadores últimos *levelings* para imagens coloridas, bem como as estratégias baseadas em funções de energia obtidas nesta dissertação e as que foram relacionadas no item anterior.

REFERÊNCIAS BIBLIOGRÁFICAS

- ABDALLAH, M. B. et al. **Automatic Extraction of Blood Vessels in the Retinal Vascular Tree Using Multiscale Medialness.** *Journal of Biomedical Imaging*, Hindawi Publishing Corp., New York, NY, United States, v. 2015, p. 1:1–1:1, jan. 2015. ISSN 1687-4188. Citado na pág. 82.
- ALVES, W. A. et al. **Segmentation of Retinal Blood Vessels Based on Ultimate Elongation Opening.** In: SPRINGER. *International Conference Image Analysis and Recognition.* [S.l.], 2016. p. 727–733. Citado na pág. 76, 82.
- ALVES, W. A.; HASHIMOTO, R. F.; MARCOTEGUI, B. **Ultimate levelings.** *Computer Vision and Image Understanding*, 2017. ISSN 1077-3142. Citado na pág. 18, 21, 36, 38, 48, 50.
- ALVES, W. A. L. *Últimos levelings: conceitos, propriedades, algoritmos e aplicações em processamento e análise de imagens.* Tese (Thesis) — Universidade de São Paulo, 2013. Citado na pág. 21, 48, 49.
- ALVES, W. A. L.; HASHIMOTO, R. F. **Ultimate grain filter.** In: *2014 IEEE International Conference on Image Processing (ICIP).* [S.l.: s.n.], 2014. p. 2953–2957. ISSN 1522-4880. Citado na pág. 9, 18, 36, 38, 50, 51, 52.
- ALVES, W. A. L.; MORIMITSU, A.; HASHIMOTO, R. F. **Scale-Space Representation Based on Levelings Through Hierarchies of Level Sets", bookTitle="Mathematical Morphology and Its Applications to Signal and Image Processing: 12th International Symposium, ISMM 2015, Reykjavik, Iceland, May 27-29, 2015. Proceedings.** In: _____. Cham: Springer International Publishing, 2015. p. 265–276. ISBN 978-3-319-18720-4. Citado na pág. 48.
- BALLESTER, C. et al. **Level Lines Selection with Variational Models for Segmentation and Encoding.** *Journal of Mathematical Imaging and Vision*, v. 27, n. 1, p. 5–27, Jan 2007. ISSN 1573-7683. Citado na pág. 11, 16, 38, 53, 55, 56, 59, 60, 61, 62, 64, 65, 66.
- BARBEDO, J. G. A. **Digital image processing techniques for detecting, quantifying and classifying plant diseases.** *SpringerPlus*, v. 2, n. 1, p. 660, Dec 2013. Citado na pág. 68, 69.
- BERGER, C. et al. **Effective Component Tree Computation with Application to Pattern Recognition in Astronomical Imaging.** In: *2007 IEEE International Conference on Image Processing.* [S.l.: s.n.], 2007. v. 4, p. IV – 41–IV – 44. ISSN 1522-4880. Citado na pág. 11, 41, 42, 44, 45, 46.
- BEUCHER, S. **Numerical residues.** *Image and Vision Computing*, v. 25, n. 4, p. 405 – 415, 2007. ISSN 0262-8856. International Symposium on Mathematical Morphology 2005. Citado na pág. 50.
- BHARDWAJ, S.; MITTAL, A. **A Survey on Various Edge Detector Techniques.** *Procedia Technology*, v. 4, p. 220 – 226, 2012. ISSN 2212-0173. 2nd International Conference on Computer, Communication, Control and Information Technology(C3IT-2012) on February 25 - 26, 2012. Citado na pág. 60.

- BLUM, H.; NAGEL, R. N. **Shape description using weighted symmetric axis features.** *Pattern Recognition*, v. 10, n. 3, p. 167 – 180, 1978. ISSN 0031-3203. The Proceedings of the IEEE Computer Society Conference. *Citado na pág. 16.*
- BREEN, E. J.; JONES, R. **Attribute Openings, Thinnings, and Granulometries.** *Computer Vision and Image Understanding*, v. 64, n. 3, p. 377 – 389, 1996. ISSN 1077-3142. *Citado na pág. 36.*
- BRÜGGERMANN, R.; PATIL, G. **Ranking and Prioritization for Multi-indicator Systems: Introduction to Partial Order Applications.** Springer New York, 2011. (Environmental and Ecological Statistics). ISBN 9781441984777. Disponível em: <<https://books.google.com.br/books?id=OpUPQeD3CtgC>>. *Citado na pág. 32.*
- BUDAI, A. et al. **Robust Vessel Segmentation in Fundus Images.** v. 2013, p. 154860, 12 2013. *Citado na pág. 82.*
- BURGER, W.; BURGE, M. J. **Digital image processing: an algorithmic introduction using Java.** [S.l.]: Springer, 2008. *Citado na pág. 65.*
- CARLINET, E.; GÉRAUD, T. **A Comparative Review of Component Tree Computation Algorithms.** *IEEE Transactions on Image Processing*, v. 23, n. 9, p. 3885–3895, Sept 2014. ISSN 1057-7149. *Citado na pág. 41.*
- CASELLES, V.; MONASSE, P. **Grain Filters.** *J. Math. Imaging Vis.*, Kluwer Academic Publishers, Norwell, MA, USA, v. 17, n. 3, p. 249–270, nov 2002. ISSN 0924-9907. *Citado na pág. 38.*
- CASELLES, V.; MONASSE, P. **Geometric Description of Images As Topographic Maps.** 1st. ed. [S.l.]: Springer Publishing Company, Incorporated, 2009. ISBN 364204610X, 9783642046100. *Citado na pág. 21, 31.*
- CHENG, G.; HAN, J. **A survey on object detection in optical remote sensing images.** *{ISPRS} Journal of Photogrammetry and Remote Sensing*, v. 117, p. 11 – 28, 2016. ISSN 0924-2716. *Citado na pág. 16.*
- FABRIZIO, J.; MARCOTEGUI, B. **Fast Implementation of the Ultimate Opening.** In: _____. *Mathematical Morphology and Its Application to Signal and Image Processing: 9th International Symposium, ISMM 2009 Groningen, The Netherlands, August 24-27, 2009 Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. p. 272–281. ISBN 978-3-642-03613-2. *Citado na pág. 52.*
- FALCAO, A. X.; UDUPA, J. K.; MIYAZAWA, F. K. **An ultra-fast user-steered image segmentation paradigm: live wire on the fly.** *IEEE Transactions on Medical Imaging*, v. 19, n. 1, p. 55–62, Jan 2000. ISSN 0278-0062. *Citado na pág. 11, 43, 44, 65, 67.*
- FELIPE-RIVERON, E.; GARCIA-GUIMERAS, N. **Extraction of blood vessels in ophthalmic color images of human retinas.** *Lecture Notes in Computer Science*, Springer, v. 4225, p. 118, 2006. *Citado na pág. 75.*
- FRAZ, M. et al. **Blood Vessel Segmentation Methodologies in Retinal Images - A Survey.** *Comput. Methods Prog. Biomed.*, Elsevier North-Holland, Inc., New York, NY, USA, v. 108, n. 1, p. 407–433, out. 2012. ISSN 0169-2607. *Citado na pág. 75, 80.*

- FREEMAN, H. **On the Encoding of Arbitrary Geometric Configurations.** *IRE Transactions on Electronic Computers*, EC-10, n. 2, p. 260–268, June 1961. ISSN 0367-9950. [Citado na pág. 16.](#)
- FREIXENET, J. et al. **Yet another survey on image segmentation: Region and boundary information integration.** *Computer Vision—ECCV 2002*, Springer, p. 21–25, 2002. [Citado na pág. 53.](#)
- FU, K.-S.; MUI, J. **A survey on image segmentation.** *Pattern recognition*, Elsevier, v. 13, n. 1, p. 3–16, 1981. [Citado na pág. 53.](#)
- GEETHARAMANI, R.; BALASUBRAMANIAN, L. **Retinal blood vessel segmentation employing image processing and data mining techniques for computerized retinal image analysis.** *Biocybernetics and Biomedical Engineering*, v. 36, n. 1, p. 102 – 118, 2016. ISSN 0208-5216. [Citado na pág. 82.](#)
- GEMAN, S.; GEMAN, D. **Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images.** *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6, n. 6, p. 721–741, Nov 1984. ISSN 0162-8828. [Citado na pág. 84.](#)
- GÉRAUD, T. et al. **A Quasi-linear Algorithm to Compute the Tree of Shapes of nD Images.** In: _____. *Mathematical Morphology and Its Applications to Signal and Image Processing: 11th International Symposium, ISMM 2013, Uppsala, Sweden, May 27-29, 2013. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. p. 98–110. ISBN 978-3-642-38294-9. [Citado na pág. 11, 38, 44, 46.](#)
- GIUFFRIDA, M. V.; MINERVINI, M.; TSAFTARIS, S. **Learning to Count Leaves in Rosette Plants.** In: TSAFTARIS, H. S. S. A.; PRIDMORE, T. (Ed.). *Proceedings of the Computer Vision Problems in Plant Phenotyping (CVPPP)*. [S.l.]: BMVA Press, 2015. p. 1.1–1.13. ISBN 1-901725-55-3. [Citado na pág. 69.](#)
- GUIGUES, L.; COCQUEREZ, J. P.; MEN, H. L. **Scale-Sets Image Analysis.** *International Journal of Computer Vision*, v. 68, n. 3, p. 289–317, Jul 2006. ISSN 1573-1405. [Citado na pág. 84.](#)
- HASSAN, G. et al. **Retinal Blood Vessel Segmentation Approach Based on Mathematical Morphology.** *Procedia Computer Science*, v. 65, n. Supplement C, p. 612 – 622, 2015. ISSN 1877-0509. International Conference on Communications, management, and Information technology (ICCMIT’2015). [Citado na pág. 76.](#)
- HOOVER, A. D.; KOUZNETSOVA, V.; GOLDBAUM, M. **Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response.** *IEEE Transactions on Medical Imaging*, v. 19, n. 3, p. 203–210, March 2000. ISSN 0278-0062. [Citado na pág. 75, 79.](#)
- JONES, R. **Connected Filtering and Segmentation Using Component Trees.** *Computer Vision and Image Understanding*, v. 75, n. 3, p. 215 – 228, 1999. ISSN 1077-3142. [Citado na pág. 27, 36.](#)
- KASS, M.; WITKIN, A.; TERZOPOULOS, D. **Snakes: Active contour models.** *International Journal of Computer Vision*, v. 1, n. 4, p. 321–331, Jan 1988. [Citado na pág. 19, 84.](#)

- KIRAN, B. R.; SERRA, J. **Global-local optimizations by hierarchical cuts and climbing energies.** *Pattern Recognition*, v. 47, n. 1, p. 12 – 24, 2014. ISSN 0031-3203.
Citado na pág. 84.
- KOENDERINK, J. J. **The structure of images.** *Biological cybernetics*, Springer, v. 50, n. 5, p. 363–370, 1984. Citado na pág. 16, 62.
- KOEPFLER, G.; LOPEZ, C.; MOREL, J. M. **A Multiscale Algorithm for Image Segmentation by Variational Method.** *SIAM Journal on Numerical Analysis*, v. 31, n. 1, p. 282–299, 1994. Citado na pág. 56.
- KREYSZIG, E. **Introductory Functional Analysis with Applications.** Wiley India Pvt. Limited, 2007. (Wiley classics library). ISBN 9788126511914. Disponível em: <<https://books.google.com.br/books?id=osXw-pRsptoC>>. Citado na pág. 53.
- LECLERC, Y. G. **Constructing simple stable descriptions for image partitioning.** *International journal of computer vision*, Springer, v. 3, n. 1, p. 73–102, 1989. Citado na pág. 84.
- LEMPITSKY, V. et al. **Image segmentation with a bounding box prior.** 2009. 277-284 p. Citado na pág. 69.
- LIPSCHUTZ, S. **Topologia geral.** [S.l.]: McGraw-Hill do Brasil, 1971. Citado na pág. 21, 23.
- LISKOWSKI, P.; KRAWIEC, K. **Segmenting Retinal Blood Vessels With Deep Neural Networks.** *IEEE Transactions on Medical Imaging*, v. 35, n. 11, p. 2369–2380, Nov 2016. ISSN 0278-0062. Citado na pág. 76.
- LONCARIC, S. **A survey of shape analysis techniques.** *Pattern Recognition*, v. 31, n. 8, p. 983 – 1001, 1998. ISSN 0031-3203. Citado na pág. 16.
- MAPAYI, T.; VIRIRI, S.; TAPAMO, J.-R. **Comparative Study of Retinal Vessel Segmentation Based on Global Thresholding Techniques.** v. 2015, p. 1–15, 02 2015. Citado na pág. 82.
- MARCOTEGUI, B.; HERNÁNDEZ, J.; RETORNAZ, T. **Ultimate Opening and Gradual Transitions.** In: _____. *Mathematical Morphology and Its Applications to Image and Signal Processing: 10th International Symposium, ISMM 2011, Verbania-Intra, Italy, July 6-8, 2011. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 166–177. ISBN 978-3-642-21569-8. Citado na pág. 9, 50, 51.
- MARQUÉS, F.; VILAPLANA, V. **Face segmentation and tracking based on connected operators and partition projection.** *Pattern Recognition*, v. 35, n. 3, p. 601 – 614, 2002. ISSN 0031-3203. Image/Video Communication. Citado na pág. 27.
- MATAS, J. et al. **Robust wide-baseline stereo from maximally stable extremal regions.** *Image and Vision Computing*, v. 22, n. 10, p. 761 – 767, 2004. ISSN 0262-8856. British Machine Vision Computing 2002. Citado na pág. 20, 36, 74.
- MATAS, P. et al. **Parallel algorithm for concurrent computation of connected component tree.** In: SPRINGER. *International Conference on Advanced Concepts for Intelligent Vision Systems.* [S.l.], 2008. p. 230–241. Citado na pág. 41.

- MEYER, F. **From Connected Operators to Levelings.** In: *Proceedings of the Fourth International Symposium on Mathematical Morphology and Its Applications to Image and Signal Processing.* Norwell, MA, USA: Kluwer Academic Publishers, 1998. (ISMM '98), p. 191–198. ISBN 0-7923-5133-9. *Citado na pág. 21, 27, 28.*
- MEYER, F. **The Levelings.** In: *Proceedings of the Fourth International Symposium on Mathematical Morphology and Its Applications to Image and Signal Processing.* Norwell, MA, USA: Kluwer Academic Publishers, 1998. (ISMM '98), p. 199–206. ISBN 0-7923-5133-9. *Citado na pág. 21, 27, 28.*
- MEYER, F. **Levelings and Flat Zone Morphology.** In: *2010 20th International Conference on Pattern Recognition.* [S.l.: s.n.], 2010. p. 1570–1573. ISSN 1051-4651. *Citado na pág. 28.*
- MEYER, F.; MARAGOS, P. **Nonlinear Scale-Space Representation with Morphological Levelings.** *Journal of Visual Communication and Image Representation*, v. 11, n. 2, p. 245 – 265, 2000. ISSN 1047-3203. *Citado na pág. 28, 29, 48.*
- MINERVINI, M. et al. **Finely-grained annotated datasets for image-based plant phenotyping.** *Pattern Recognition Letters*, v. 81, p. 80 – 89, 2016. ISSN 0167-8655. *Citado na pág. 68, 69, 70, 72.*
- MONASSE, P.; GUICHARD, F. **Fast computation of a contrast-invariant image representation.** *IEEE Transactions on Image Processing*, v. 9, n. 5, p. 860–872, May 2000. ISSN 1057-7149. *Citado na pág. 38, 44.*
- MOREL, J. M.; SOLIMINI, S. **Variational Methods in Image Segmentation.** Cambridge, MA, USA: Birkhauser Boston Inc., 1995. ISBN 0-8176-3720-6. *Citado na pág. 56.*
- MOTT-SMITH, J. C. **Medial axis transformations.** *Picture Processing and Psychopictorics*, Academic Press, New York, p. 267–283, 1970. *Citado na pág. 16.*
- MUMFORD, D.; SHAH, J. **Optimal approximations by piecewise smooth functions and associated variational problems.** *Communications on pure and applied mathematics*, Wiley Online Library, v. 42, n. 5, p. 577–685, 1989. *Citado na pág. 19, 53, 55, 84.*
- NAJMAN, L.; COUPRIE, M. **Building the Component Tree in Quasi-Linear Time.** *IEEE Transactions on Image Processing*, v. 15, n. 11, p. 3531–3539, Nov 2006. ISSN 1057-7149. *Citado na pág. 41.*
- NAJMAN, L.; GÉRAUD, T. **Discrete set-valued continuity and interpolation.** In: HENDRIKS, C. L.; BORGEFORS, G.; STRAND, R. (Ed.). *Mathematical Morphology and Its Application to Signal and Image Processing – Proceedings of the 11th International Symposium on Mathematical Morphology (ISMM).* Uppsala, Sweden: Springer, 2013. (Lecture Notes in Computer Science Series, v. 7883), p. 37–48. *Citado na pág. 44.*
- NEWMAN, M. **Elements of the Topology of Plane Sets of Points.** [S.l.]: The University Press, 1939. *Citado na pág. 21, 23.*
- NGUYEN, U. T. et al. **An effective retinal blood vessel segmentation method using multi-scale line detection.** *Pattern Recognition*, v. 46, n. 3, p. 703 – 715, 2013. ISSN 0031-3203. *Citado na pág. 76, 83.*

- NISTÉR, D.; STEWÉNIUS, H. **Linear time maximally stable extremal regions.** *Computer Vision-ECCV 2008*, Springer, p. 183–196, 2008. Citado na pág. 41.
- PAVLIDIS, T.; LIOW, Y. T. **Integrating region growing and edge detection.** *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 12, n. 3, p. 225–233, Mar 1990. ISSN 0162-8828. Citado na pág. 16.
- PENG, B.; ZHANG, L.; ZHANG, D. **A survey of graph theoretical approaches to image segmentation.** v. 46, p. 1020–1038, 03 2013. Citado na pág. 53.
- PERSOON, E.; FU, K. S. **Shape Discrimination Using Fourier Descriptors.** *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8, n. 3, p. 388–397, May 1986. ISSN 0162-8828. Citado na pág. 16.
- SALEMBIER, P.; OLIVERAS, A.; GARRIDO, L. **Antiextensive connected operators for image and sequence processing.** *IEEE Transactions on Image Processing*, v. 7, n. 4, p. 555–570, Apr 1998. ISSN 1057-7149. Citado na pág. 17, 27, 36, 41.
- SALEMBIER, P.; SERRA, J. **Flat zones filtering, connected operators, and filters by reconstruction.** *IEEE Transactions on Image Processing*, v. 4, n. 8, p. 1153–1160, Aug 1995. ISSN 1057-7149. Citado na pág. 27, 28.
- SERRA, J. ***Image Analysis and Mathematical Morphology.*** Orlando, FL, USA: Academic Press, Inc., 1983. ISBN 0126372403. Citado na pág. 76.
- SKARBEK, W. ladys law; KOSCHAN, A. **Colour image segmentation a survey.** *IEEE Transactions on circuits and systems for Video Technology*, v. 14, 1994. Citado na pág. 53.
- SMEDT, F. D.; BILLIAUWS, I.; GOEDEMÉ, T. **Neural networks and low-cost optical filters for plant segmentation.** *International Journal of Computer Information Systems and Industrial Management Applications*, MIR Publishers, v. 3, p. 804–811, 2011. Citado na pág. 68.
- SOILLE, P. ***Morphological Image Analysis: Principles and Applications.*** 2. ed. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2003. ISBN 3540429883. Citado na pág. 16, 76.
- SONG, Y. **A Topdown Algorithm for Computation of Level Line Trees.** *IEEE Transactions on Image Processing*, v. 16, n. 8, p. 2107–2116, Aug 2007. ISSN 1057-7149. Citado na pág. 44.
- SRINIDHI, C. L.; APARNA, P.; RAJAN, J. **Recent Advancements in Retinal Vessel Segmentation.** *J. Med. Syst.*, Plenum Press, New York, NY, USA, v. 41, n. 4, p. 1–22, abr. 2017. ISSN 0148-5598. Citado na pág. 75, 80.
- STAAL, J. et al. **Ridge-based vessel segmentation in color images of the retina.** *IEEE Transactions on Medical Imaging*, v. 23, n. 4, p. 501–509, April 2004. ISSN 0278-0062. Citado na pág. 75, 76, 79.
- TARJAN, R. E. **Efficiency of a Good But Not Linear Set Union Algorithm.** *J. ACM*, ACM, New York, NY, USA, v. 22, n. 2, p. 215–225, abr. 1975. ISSN 0004-5411. Citado na pág. 41.

- WANG, S. et al. **Hierarchical retinal blood vessel segmentation based on feature and ensemble learning.** *Neurocomputing*, v. 149, n. Part B, p. 708 – 717, 2015. ISSN 0925-2312. Citado na pág. 82.
- WILKINSON, M. H. et al. **Concurrent Computation of Differential Morphological Profiles on Giga-Pixel Images.** In: SPRINGER. *ISMM*. [S.l.], 2011. p. 331–342. Citado na pág. 41.
- WILKINSON, M. H. F. et al. **Concurrent Computation of Attribute Filters on Shared Memory Parallel Machines.** *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 30, n. 10, p. 1800–1813, Oct 2008. ISSN 0162-8828. Citado na pág. 41.
- XU, Y. *Tree-based shape spaces : definition and applications in image processing and computer vision.* Tese (Thesis) — Université Paris-Est, dez. 2013. Citado na pág. 60.
- XU, Y. et al. **Efficient Computation of Attributes and Saliency Maps on Tree-Based Image Representations.** In: BENEDIKTSSON, J. et al. (Ed.). *Mathematical Morphology and Its Application to Signal and Image Processing – Proceedings of the 12th International Symposium on Mathematical Morphology (ISMM)*. Reykjavik, Iceland: Springer, 2015. (Lecture Notes in Computer Science Series, v. 9082), p. 693–704. Citado na pág. 65.
- XU, Y.; GÉRAUD, T.; NAJMAN, L. **Salient level lines selection using the Mumford-Shah functional.** *2013 IEEE International Conference on Image Processing*, p. 1227–1231, Sept 2013. ISSN 1522-4880. Citado na pág. 16, 38, 53, 60, 61, 62.
- XU, Y.; GÉRAUD, T.; NAJMAN, L. **Hierarchical Image Simplification and Segmentation Based on Mumford-Shah-salient Level Line Selection.** *Pattern Recogn. Lett.*, Elsevier Science Inc., New York, NY, USA, v. 83, n. P3, p. 278–286, nov. 2016. ISSN 0167-8655. Citado na pág. 11, 38, 53, 62, 63, 64, 66, 67, 84.
- XU, Y.; GÉRAUD, T.; NAJMAN, L. **Context-based energy estimator: Application to object segmentation on the tree of shapes.** In: *2012 19th IEEE International Conference on Image Processing*. [S.l.: s.n.], 2012. p. 1577–1580. ISSN 1522-4880. Citado na pág. 16, 38, 74, 83.
- XU, Y. et al. **Tree-Based Morse Regions: A Topological Approach to Local Feature Detection.** *IEEE Transactions on Image Processing*, v. 23, n. 12, p. 5612–5625, Dec 2014. ISSN 1057-7149. Citado na pág. 20.
- YANG, Y.; HUANG, S.; RAO, N. **An Automatic Hybrid Method for Retinal Blood Vessel Extraction.** *Int. J. Appl. Math. Comput. Sci.*, Versita, Warsaw, Poland, Poland, v. 18, n. 3, p. 399–407, set. 2008. ISSN 1641-876X. Citado na pág. 76.
- YIN, Y.; ADEL, M.; BOURENNANE, S. **Automatic Segmentation and Measurement of Vasculature in Retinal Fundus Images Using Probabilistic Formulation.** v. 2013, p. 260410, 12 2013. Citado na pág. 82.
- ZAHN, C. T.; ROSKIES, R. Z. **Fourier Descriptors for Plane Closed Curves.** *IEEE Transactions on Computers*, C-21, n. 3, p. 269–281, March 1972. ISSN 0018-9340. Citado na pág. 16.

- ZANA, F.; KLEIN, J. C. **Segmentation of vessel-like patterns using mathematical morphology and curvature evaluation.** *IEEE Transactions on Image Processing*, v. 10, n. 7, p. 1010–1019, Jul 2001. ISSN 1057-7149. Citado na pág. [76](#), [77](#).
- ZHANG, D.; LU, G. **Review of shape representation and description techniques.** *Pattern Recognition*, v. 37, n. 1, p. 1 – 19, 2004. ISSN 0031-3203. Citado na pág. [16](#).
- ZHANG, J. et al. **Blood Vessel Segmentation of Retinal Images Based on Neural Network.** p. 11–17, 08 2015. Citado na pág. [76](#).

ÍNDICE REMISSIVO

- Árvore de componentes, 36
 Árvore de formas, 38
 Árvores e *posets*, 31
 Árvores morfológicas, 31
 Últimos *levelings*, 50
Bounding box detection, 69
Bounding box, 72
Max-tree, 36
Min-tree, 36
Pixel, 23
Pixel do nó compacto, 37
 Adjacência, 24
 Algoritmos para computação dos atributos de energia, 65
 Algoritmos para construção de árvores morfológicas, 41
 Atributo funcional, 62
 Buracos internos, 38
 Caminho, 24
 Componente conexo, 25
 Conjunto de nível, 34
 Definições sobre árvores, 32
 Detecção de plantas, 68
 Diagrama de Hasse, 32
 Dilatação geodésica, 77
 Elemento estruturante, 76
 Espaço de escalas, 29
 Estratégias de filtragem residual, 52, 64
 Extrema regional, 26
 Família de primitivas, 50
 Filtragem de resíduos, 52
 Funções de energia, 53
 Funcional de Mumford-shah, 53
 Funcional discreta de Mumford-shah, 55
 Funcional variacional, 56
 Hierarquias de partições, 84
 ImageJ, 20
 Imagem, 23
 Imagem binária, 23
 Imagem multibanda, 23
 Menor componente, 37
 Morfologia Matemática, 16
 Operador *lower-leveling*, 28
 Operador *upper-leveling*, 28
 Operador *leveling*, 28
 Operador *ultimate attribute closing*, 50
 Operador *ultimate attribute opening*, 50
 Operador *ultimate grain filters*, 50
 Operador abertura, 77
 Operador conexo, 27
 Operador dilatação, 76
 Operador erosão, 76
 Operador fechamento, 77
 Operador residual, 50
 Partição, 26
 Partição associada, 56
 Primitivas, 50
 Reconstrução de árvores, 39
 Reconstrução por dilatação, 77
 Resíduos, 50
 Segmentação de vasos sanguíneos, 75
 Zona plana, 25