Interactive 3D Segmentation Repair with Image-Foresting Transform, Supervoxels and Seed Robustness

Anderson Carlos Moreira Tavares

Thesis presented
to the
Institute of Mathematics and Statistics
of
University of São Paulo
to
the title
of
Ph.D. in Computer Science

Program: Ph.D. in Computer Science

Supervisor: Prof. Dr. Paulo Vechiatto de Miranda

During the development of this work the author has received financial support from CAPES

São Paulo, August 2017

Interactive 3D Segmentation Repair with Image-Foresting Transform, Supervoxels and Seed Robustness

This thesis version contains corrections and changes suggested by the Judging Committee during the defense of the original version of the work, held at June 2nd of 2017. A copy of the original and correction versions are available at the Institute of Mathematics and Statistics of the University of Sao Paulo.

Judging Committee:

- Prof. Dr. Paulo André Vechiatto de Miranda (supervisor) - IME-USP
- Prof. Dr. Roberto Hirata Junior - IME-USP
- Prof. Dr. Alexandre Xavier Falcão - IC-UNICAMP
- Prof. Dr. Fábio Augusto Menocci Cappabianco - ICT-UNIFESP
- Prof. Dr. João Paulo Papa - FC-UNESP

Acknowledgments

I would first like to thank my mother Luiza and my bride Valeska for the attention, support and motivation to move on and doing this research. With enthusiastic and encouraging chats, they make my life so vibrant and full of expectations. They are my mind and my heart.

I would also like to thank my thesis advisor Prof. Dr. Paulo Vechiatto de Miranda of the Institute of Mathematics and Statistics at University of Sao Paulo. Prof. Miranda is a gentle and hardworking person, specially at the difficult moments. He was very engaged in our research, doing a lot of effort to make the best for us and for the academy. He is also a great developer, specially when using Emacs with its complex shortcuts.

I must express my profound gratitude for the experts who were involved in the validation survey for this research project: Prof. Dr. Roberto Hirata Junior, Prof. Dr. Alexandre Xavier Falcão, Prof. Dr. Fábio Augusto Menocci Cappabianco and João Paulo Papa.

Thanks also to Prof. Dr. Carlos Hitoshi Morimoto for opening this opportunity for my doctoral application and for the tips on my study, career and personality, and to Prof. Dr. Marcel P. Jackowsky for trusting me in many projects we have developed. His medical imaging lectures opened many new doors in my mind.

Thanks to my fellow doctoral and master students for the friendship and support. Creativity, effort, focus and perseverance dominated these brilliant people and environment. I really also enjoyed the funny and relaxing days like volleyball.

A very special gratitude goes out to all down at CNPq (305381/2012-1, 308985/2015-0, 486083/2013-6, FINEP 1266/13), FAPESP grant # 2011/50761-2, CAPES, NAP eScience - PRP - IME - USP for helping and providing the funding and infrastructure for the work. It is no surprise it is one of the best academic institutions of the world. Thanks also to Dr. J. K. Udupa (MIPG-UPENN) for the images.

Last but not least, thank you reader from the bottom of my heart for investing your time and attention in this work. I hope this text meets your expectations and can contribute to your future projects.

For Luiza and Valeska

Resumo

TAVARES, A. C. M. **Reparação Interativa de Segmentações 3D com Transformada Imagem-Floresta, Supervoxels e Robustez de Sementes**. 2017. 65 f. Tese (Doutorado) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2017.

Segmentação de imagem consiste no seu particionamento em regiões, tal como para isolar os pixels pertencentes a objetos de interesse em uma imagem, sendo uma etapa importante para visão computacional, processamento de imagens médicas e outras aplicações. Muitas vezes a segmentação automática gera resultados com imperfeições. O usuário pode corrigi-las editando-a manualmente, interativamente ou simplesmente descartar o resultado e gerar outro automaticamente. Métodos interativos combinam os benefícios dos métodos manuais e automáticos, reduzindo o esforço do usuário e utilizando seu conhecimento de alto nível. Nos métodos baseados em sementes, para continuar ou reparar uma segmentação prévia (presegmentação), evitando o usuário começar do zero, é necessário resolver o Problema da Segmentação Interativa Reversa (RISP), ou seja, estimar automaticamente as sementes que o gerariam. Para isso, este trabalho particiona o objeto da segmentação em núcleos. Em um núcleo, duas sementes separadamente produzem o mesmo resultado, tornando uma delas redundante. Com isso, apenas uma semente por núcleo é necessária. Núcleos contidos nos resultados de outros núcleos são redundantes e também podem ser descartados, reduzindo ainda mais o conjunto de sementes, um processo denominado Análise de Redundância. Um conjunto mínimo de sementes para a presegmentação é gerado e o problema da reparação interativa pode então ser resolvido através da adição de novas sementes ou remoção. Dentro do arcabouço da Transformada Imagem-Floresta (IFT), novos métodos como Oriented Image-Foresting Transform (OIFT) e Oriented Relative Fuzzy Connectedness (ORFC) foram desenvolvidos. Todavia, não há algoritmos para calcular o núcleo destes métodos. Este trabalho desenvolve tais algoritmos, com prova de corretude. Os núcleos também nos fornecem uma indicação do grau de robustez dos métodos sobre o posicionamento das sementes. Por isso, um método híbrido do GraphCut com o núcleo do ORFC, bem como um Coeficiente de Robustez (RC), foram desenvolvidos. Neste trabalho também foi desenvolvida outra solução para reparar segmentações, a qual é baseada em IFT-SLIC, originalmente utilizada para gerar supervoxels. Resultados experimentais analisam, comparam e demonstram o potencial destas soluções.

Palavras-chave: segmentação baseada em grafos, transformada imagem-floresta, robustez de sementes, supervoxels.

Abstract

TAVARES, A. C. M. **Interactive 3D Segmentation Repair with Image-Foresting Transform, Supervoxels and Seed Robustness**. 2017. 65 f. Tese (Doutorado) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2017.

Image segmentation consists on its partition into relevant regions, such as to isolate the pixels belonging to desired objects in the image domain, which is an important step for computer vision, medical image processing, and other applications. Many times automatic segmentation generates results with imperfections. The user can correct them by editing manually, interactively or can simply discard the segmentation and try to automatically generate another result by a different method. Interactive methods combine benefits from manual and automatic ones, reducing user effort and using its high-level knowledge. In seed-based methods, to continue or repair a prior segmentation (presegmentation), avoiding the user to start from scratch, it is necessary to solve the Reverse Interactive Segmentation Problem (RISP), that is, how to automatically estimate the seeds that would generate it. In order to achieve this goal, we first divide the segmented object into its composing cores. Inside a core, two seeds separately always produce the same result, making one redundant. With this, only one seed per core is required. Cores leading to segmentations which are contained in the result of other cores are redundant and can also be discarded, further reducing the seed set, a process called Redundancy Analysis. A minimal set of seeds for presegmentation is generated and the problem of interactive repair can be solved by adding new seeds or removing seeds. Within the framework of the Image-Foresting Transform (IFT), new methods such as Oriented Image-Foresting Transform (OIFT) and Oriented Relative Fuzzy Connectedness (ORFC) were developed. However, there were no known algorithms for computing the core of these methods. This work develops such algorithms, with proof of correctness. The cores also give an indication of the degree of robustness of the methods on the positioning of the seeds. Therefore, a hybrid method that combines GraphCut and the ORFC cores, as well as the Robustness Coefficient (RC), have been developed. In this work, we present another developed solution to repair segmentations, which is based on IFT-SLIC, originally used to generate supervoxels. Experimental results analyze, compare and demonstrate the potential of these solutions.

Keywords: graph-based segmentation, image-foresting transform, seed robustness, supervoxels.

Contents

List of Acronyms

DC *Distance Cut*. 2

DIFT *Differential Image Foresting Transform*. 32

FC *Fuzzy Connectedness*. 2

FN *False Negatives*. 13, 14

FP *False Positives*. 13

GC *Graph Cut*. 2, 5, 27, 28

IFT *Image-Foresting Transform*. 2, 3, 5, 6, 15, 16, 19, 31

MI *Monotonically Incremental*. 10

MR *Magnetic Ressonance*. xvi, 7, 28, 29, 34, 36

OIFT *Oriented Image Foresting Transform*. 6, 17, 21, 22, 24–26, 39

OPSF *Optimum-Path Spanning Forest*. 10

OPSFP *Optimum-Path Spanning Forest Problem*. 10, 12, 15

ORFC *Oriented Relative Fuzzy Connectedness*. 6, 17, 18, 21, 27, 39

RAG *Region Adjacency Graph*. 24, 25, 37, 38

RC *Robustness Coefficient*. 26, 34–36

RFC *Relative Fuzzy Connectedness*. 18

RISP *Reverse Interactive Segmentation Problem*. 31, 40

RW *Random Walk*. 2, 4

TN *True Negatives*. 13

TP *True Positives*. 13

WS *Watershed*. 2

List of Symbols

$A(\mathcal{S}_o, \mathcal{S}_b)$	Segmentation algorithm. 12
C_{opt}	Optimal Connectivity map. xvi, 10, 37, 38
$D(\mathcal{O}, \mathcal{G})$	Dice coefficient. 13
$DCC_G(s)$	Directed Connected Component. 9
E	Set of arcs. 8, 11
G^T	Graph Transpose. 8
G	Graph. 8, 11, 12
H	Handicap (initialization) function. 10
$I(s)$	Image intensity. 7
I	Digital image. 7
L	Label map. 12
Pr	Predecessor map and spanning forest. 8
Q	Priority Queue. 15–17
R^{Pr}	Set of forest roots. 8
$SCC_G(s)$	Strongly Connected Component. 9
V	Set of nodes. 8–14
$\Pi(G)$	Set of paths of a graph G. 8, 9
Π_t	Set of paths from any node to t. 8
$\Pi_{\mathcal{S}\rightsquigarrow t}$	Set of paths from node $s \in \mathcal{S}$ to t. 8
$\Pi_{s\rightsquigarrow t}$	Set of paths from node s to t. 8
Π	Set of paths of a implicit graph. 8
α	Orientation factor. 11
δ	Non-oriented similarity factor. 11
$\equiv$	Equivalence relation. 21, 23–25
$\langle s, t \rangle$	Arc from node s to t. 8
$\langle t \rangle$	Trivial path. 8
$\mathcal{C}$	A cut. 12, 23, 27
$\mathcal{G}$	Groundtruth. xiii, 13, 14
$\mathcal{I}$	Image domain. 7, 11
$\mathcal{L}$	Set of distinct class labels. 12
$\mathcal{N}$	Core. xv, xvi, xix, 21–24, 26, 28, 35–38
$\mathcal{O}$	Segmentation object. xiii, 12, 13, 15, 18, 27, 31
$\mathcal{P}_L$	Partition induced by a label map. 12
$\mathcal{S}_b$	Set of background (external) seeds. xv, xvi, xix, 12, 15, 17, 18, 21–24, 26–28, 31, 35–37
$\mathcal{S}_o$	Set of object (internal) seeds. xv, xix, 12, 15, 17, 18, 21–24, 26–28, 31, 35–37
$\mathcal{S}$	Set of seeds. 8, 12, 14–16, 19, 22, 27, 28, 31

$\mathcal{X}(\mathcal{S}_o, \mathcal{S}_b)$	Space of objects restricted to seeds. 12, 18
$\mathcal{X}_\infty^\downarrow(\mathcal{S}_o, \mathcal{S}_b)$	Space of objects of minimum ∞-norm energy. 18
$\mathcal{X}$	Space of objects. 12, 18, 21
ω	Weight function. 8, 11, 12, 15, 17, 18, 23–27, 37
π_t^*	Optimum path. 10
$\pi_s \cdot \langle s, t \rangle$	Concatenation between path and arc. 8
$\pi_s \cdot \pi_{s \rightsquigarrow t}$	Concatenation between paths. 8
π_t	Path from any node to t. 8, 10
$\pi_{\mathcal{S} \rightsquigarrow t}$	Path from node $s \in \mathcal{S}$ to t. 8
$\pi_{s \rightsquigarrow t}$	Path from node s to t. 8, 9
π_t^{Pr}	Path from forest root to node t. 8
$\overset{\text{oift}}{\equiv}$	Equivalence relation based on OIFT. 24
$\varepsilon_\infty^\downarrow$	Minimum ∞-norm energy. 18, 23
ε_∞	A ∞-norm energy function. 12, 17, 18, 28
ε_q	A q-norm energy function. 12
ε	Energy function. 12
$\wedge$	Logical conjunction in propositional logic. xvi, 23, 24, 37
c	Total number of classes. 12
$f^{\nearrow}$	OIFT connectivity function. 17, 22, 23
f_D	IFT-SLIC connectivity function. 19, 31
$f_{\min}^{\leftarrow}$	Minimum weight connectivity function for reversed arcs. 18, 23–25
$f_{\min}$	Minimum weight connectivity function. 10, 16, 17
f_{euc}	Euclidean distance connectivity function. 10
f_{sum}	Weight sum connectivity function. 10
f_w	Last weight connectivity function. 10
f	Connectivity function. 10, 15
l_b	Background label. 12
l_o	Object label. 12
$s\mathcal{A}t$	Adjacency relation between nodes s and t. 11

List of Figures

List of Tables

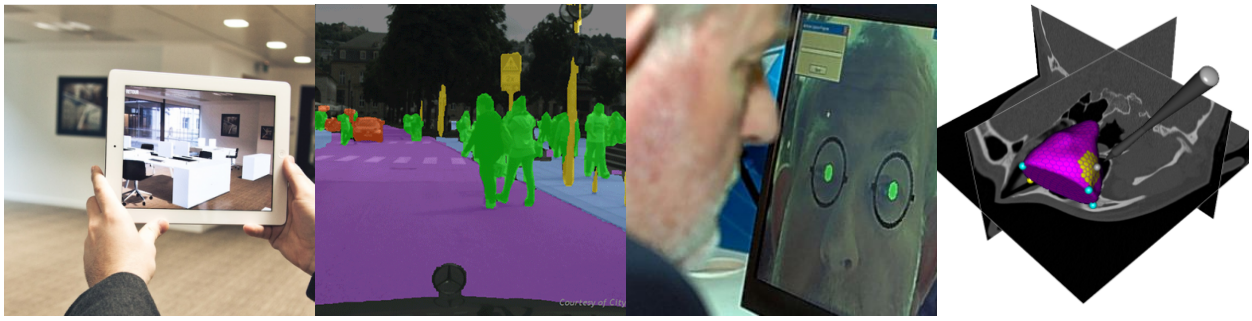
List of Algorithms

Chapter 1

Introduction

1.1 Motivation

Image segmentation is one of the most fundamental and challenging problems in image processing and computer vision [1, 2]. It consists on partitioning a digital image into regions, to simplify a posterior analysis, visualization, object representation and other tasks. It is an important step for several applications, like computer vision, medical image processing, and object recognition/tracking (Figure 1.1). In the case of binary segmentation, such as to separate an object from a background, the result of a segmentation can also be called a *mask*.



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

Figure 1.1: *Some applications of image segmentation: a) Augmented/Virtual/Mixed Reality, b) Object recognition/tracking, c) Face recognition and d) Medical Imaging.*

In a manual segmentation process, the mask can be obtained by directly labeling all voxels. Interactive methods explore high-level visual expertise but requiring minimal user effort. Automatic segmentation eliminates the need for user interaction. A simple diagram is illustrated in Figure 1.2.

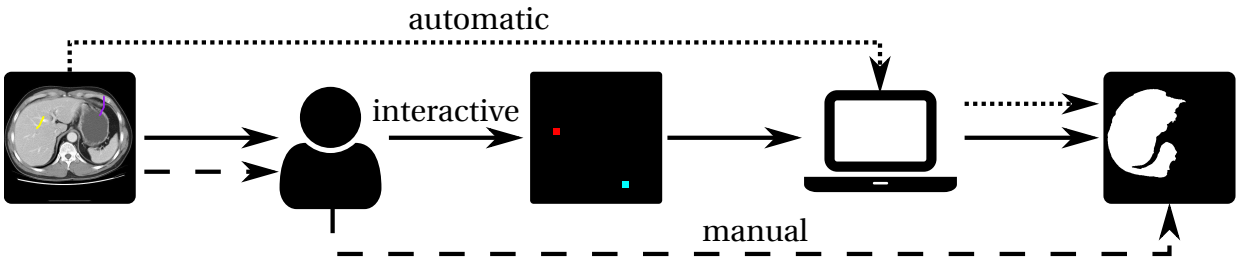


Figure 1.2: *Segmentation classification by user interaction: the user can manually label all voxels, a small set (or define parametric surfaces) in a interactive way, or use automatic process.*

Medical image segmentation assists the medical practitioner in the diagnosis of clinical pa-

tient status, performing delineation of organs, detecting abnormalities and deformities [3]. Due to the variability of medical images generated by different modalities [4], with the presence of poorly defined structures, non-standard intensity distribution, field inhomogeneity, noise, partial volume and interplay among these factors [1], automatic segmentation methods often generate results with imperfections.

A common problem in medical image analysis is how to repair an automatic segmentation, as obtained by FreeSurfer [5], SPM2 [6] and CLASP[1]. The professional can simply discard the initial segmentation (*presegmentation*) and start another one using different parameter values in the same automatic method, try other automatic methods, or can edit the presegmentation. A manual editing requires a great user effort. Besides that, the results vary among professionals and sessions of a same specialist. Editing through interactive methods combines the benefits of automatic (e.g., reduction of user effort) and manual approaches (e.g., adding the knowledge of a specialist).

Another related problem is how to continue a presegmentation obtained from interactive method. Usually this task is performed for constructing *groundtruths*, generated by successive refinements, which can be executed in different moments, or sessions. It is necessary to keep the history of user actions between consecutive sessions. However, popular formats do not store this data. For both described problems, the history of presegmentation should be estimated, which is a reverse segmentation problem, as depicted in Figure 1.3.

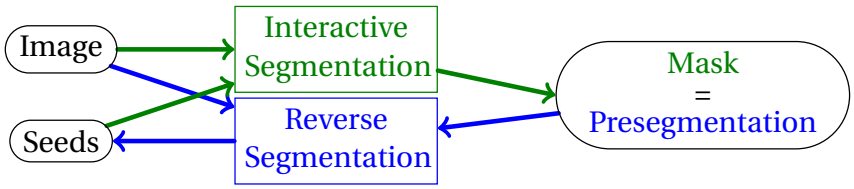


Figure 1.3: *Segmentation Editing Process: Given an image and its presegmentation, the seeds are estimated by a reverse segmentation method (blue section), so that the user can continue or correct the segmentation (green section).*

Although the segmentation problem has motivated the development of a variety of works, the problem of editing segmentations has not caught much attention. Parametric surfaces [7–10], energy minimization [11], low level editing [8–10], region-based segmentation [12, 13], edge-based ones [14–16], graphs [7, 11–13, 15] or Human-Computer Interface [7, 17] have been applied. These methods differ in user effort degree, complexity, running time, flexibility and robustness of algorithm. A complete analysis of these aspects still are demanded in comparative assessments. Empirical evaluations still are often conducted, with high degree of subjectivity.

Many image segmentation methods are modeled as graph partition problems, where a graph represents an image. In *seed*-based methods, the user defines restrictions by marking nodes as seeds, that is, by adding labels *a priori*. The algorithm finds a partition which satisfies these restrictions while optimizes a local or global expression. The labels are propagated to the remaining nodes of the graph, defining a possible segmentation. Seeds may be added and removed to perform corrections for intermediate results. *Watershed* (WS) [18, 19], *Graph Cut* (GC) [20, 21], *Fuzzy Connectedness* (FC) [22–24], *Random Walk* (RW) [25], *Distance Cut* (DC) [26], *Image-Foresting Transform* (IFT) [27] and *GrowCut* [28] are popular graph-based methods.

IFT is a popular framework of methods of graph partitioning, being an extension of optimal-path search algorithm from Dijkstra to different connectivity functions and multiple sources [27], which solves the problem of finding an optimum-path forest (OPF) by dynamic programming [29]. Its linearithmic complexity (linear depending on the priority queue) makes it an attractive solution for image segmentation. IFT can perform segmentation of multiple classes in just one step. The differential version [30] allows successive refinements in the result by adding and removing seeds, with sub-linear complexity. In the context of image segmentation as a cut in

[1] https://www.mcgill.ca/bic/

the graph, besides the traditional formulation with undirected graphs [27, 29], versions with digraphs [31–33] were also developed, investigated and applied.

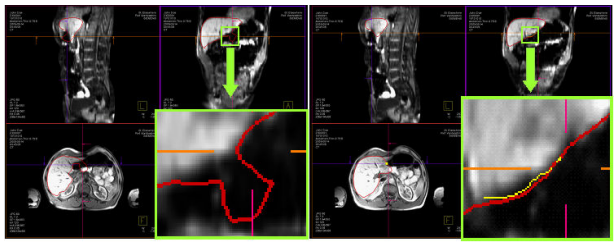
The algorithm receives a graph and a seed set, and returns a label map. To edit the segmentation, the user modifies the seed set and rerun the method, in a differential way or not. Without previous seed set, the user should redo all previous segmentation. This work estimates a minimal set of seeds, from the presegmentation by IFT. This solution may also be used for correcting results obtained from automatic segmentations. A smaller set of estimated seeds allows user modifications to exert a bigger influence to the algorithm. In the opposite way, when the seed set equals the set of image voxels, the method degenerates towards a manual editing, undesirable for an interactive segmentation. The present work develops and analyzes two approaches for interactive repair of segmentations:

- **IFT-SLIC** [34, 35]: Alexandre et al. [34] developed a supervoxel generation method which initializes a seed set equally spaced like a grid, and moves them appropriately according to intensity and shape constraints, by regulating supervoxels compactness and boundary adherence. The natural returning of seeds allows the user to edit the presegmentation by IFT;

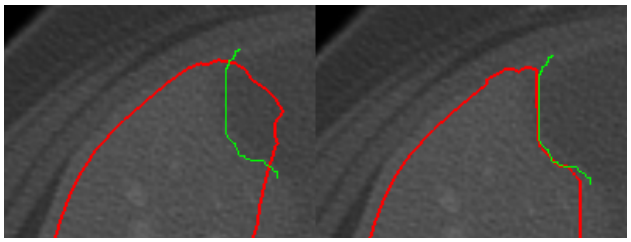
- **Theoretical robustness analysis** [12, 13, 36, 37]: if two seeds separately produce the same result, then they are equivalent, and just one is enough for defining a region. By finding equivalence regions (subsets of equivalent seeds), called *cores*, and representing each one by just one seed, a compact seed set can be built. Besides that, cores with resulting segmentations contained in results generated by other cores can be discarded, further reducing the seed set. The cores can also determine the degree of robustness to seeds placement, which can serve as a metric for comparison among methods.

1.2 Related Methods

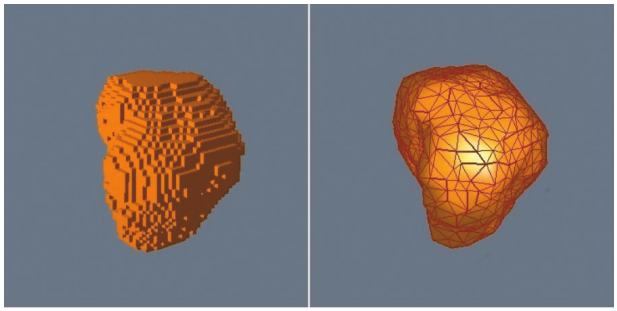
In spite of the vast literature on segmentation, only a few works have dealt with the editing issue, usually considering qualitative and highly subjective empirical evaluations [38]. Figure 1.4 illustrates the methods described in this section.



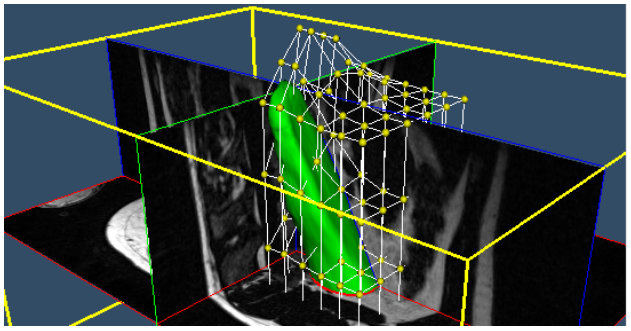
(a) *El-Zehiri et al. [14]*



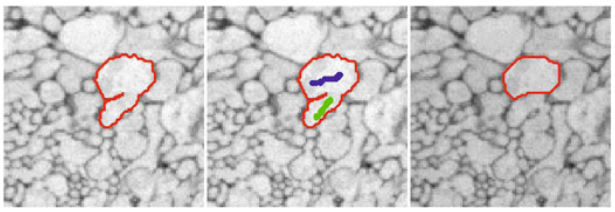
(b) *Harrison et al. [15]*



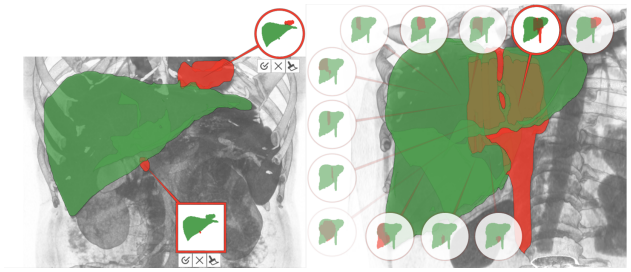
(c) *Jackowski et al. [8]*



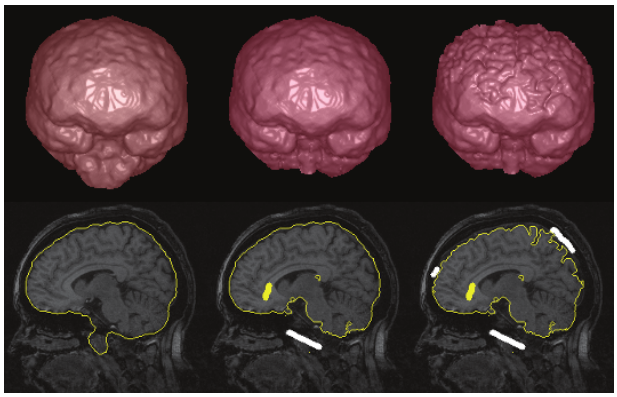
(d) *Valenzuela et al. [10]*



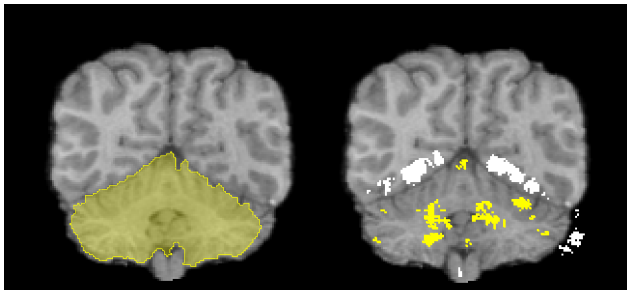
(e) *Yang et al. [11]*



(f) *Karimov et al. [17]*



(g) *Miranda et al. [12,13]*



(h) *Spina et al. [39]*

Figure 1.4: *Methods for the segmentation repair problem: a-b) Edge based RW, c-d) Parametric surfaces, e) Graph Cut, f) Human-Computer Interface, g) Seed robustness of Image-Foresting Transform.*

El-Zehiri et al. [14] and Grady and Funka-Lea [40] apply RW [25] to correct presegmented images, optimized with downsampling, which loses important and high frequency information, like small objects, negatively affecting the result. Harrison et al. [15] join discriminative classification and energy minimization with RW for contour-based correction, using GPU training. It inherits disadvantages from contour-based segmentations, like sensitivity to seed placing, lack

of texture and region information. It depends on the training set size to propagate the labels to other slices, affecting its accuracy.

Jackowski et al. [8] approximate a digital volume representing the segmented object by a Rational of Gaussians (RaG) parametric surface, allowing the user to change the surface by its control points. Advantages are compression for fast transmission, sub-voxel correction, and inclusion of graphical effects without a voxelized appearance. But editing non-compact objects by control points is not trivial. Valenzuela et al. [10] use Bézier-based surfaces. The user can modify the curve in one slice, and it propagates to the rest in 3D.

Yang and Choe [11] use GC, with energy function composed by presegmentation and new user inputs. It assumes that the presegmentation is almost correct, restricting the user active field. It inherits GC disadvantages. The graph weights are based on the Euclidean distance, not effective for non-compact objects, like veins and arteries. To remove parts of the presegmentation, the user must always unnecessarily place background and foreground seeds. Moreover, its conducted evaluation does not include a user effort analysis. Karimov et al. [17] develop a software that suggests correction candidates, based on the extraction of region skeletons, which should be similar to ground truth, and histogram similarity analysis. Complex images can affect the number of candidates.

Miranda et al. [12, 13] proposed an editing solution based on the IFT with an experimental analysis in MR-T1 three-dimensional images. Contrary to previous methods, it can be applied to multidimensional images and to objects with arbitrary shapes, with low running time and without any special hardware support. It first solves the reverse segmentation problem, with strong theoretical background, reducing the required number of seeds by employing a conservative force [13]. The corrections can then be performed in sub-linear time by differential IFT (DIFT) [30]. It is restricted to the max-arc path-cost function over an undirected graph derived from a gradient image, which is usually not the best option to deal with blurred transitions.

Spina et al. [39] proposed a solution with robot users [41], which simulate user interaction by placing brush strokes automatically to iteratively perform the segmentation task resulting in the given presegmentation. It can correct any existing delineation result [39]. However, it considered a robot user tailored to IFT-based segmentation, since the end goal was to learn the spatial distribution of seeds added to reproduce ground truth training masks, in order to output a statistical seed model of an object of interest to aid in its interactive segmentation. Hence, they were more interested in consistent seed positioning than high accuracy for editing.

Our proposed methods are also based on the IFT framework, but they are designed to circumvent the main problems of [13], such as its high number of seeds and non-uniform seed distribution, in order to give more freedom to the user to perform corrections, and using better path-cost functions. To achieve that, it is necessary to understand concepts described in Chapter 2.

1.3 Goals and Contributions

The present work has as main goals the development of methods to solve the problem of interactively editing segmentations obtained by automatic or semi-automatic techniques, in the absence of the user's action history. From this main objective, several contributions were made:

- Application and investigation of IFT-SLIC [34] in obtaining the seed set for interactive segmentation repair. Features such as compactness (which gives a more regular shape) and boundary adherence are benefits of using this method. The format of partitions, compared to other methods, is more regular [35];
- Theoretical analysis of the *Oriented Relative Fuzzy Connectedness* (ORFC) robustness and development of the algorithm that computes the ORFC cores, with proof of correctness [36];
- Theoretical analysis of the *Oriented Image Foresting Transform* (OIFT) robustness and development of the algorithm that computes the OIFT cores, with proof of correctness [37];
- Development of the hybrid method $ORFC_{Core} + GC$, and comparative evaluation with other methods [36];
- Core redundancy analysis and algorithm development [36,37];
- The proposal of new segmentation repair algorithms, for OIFT and ORFC, based on their core computation and redundancy analysis;
- Definition of robustness index of methods [37].

Throughout the text, these different contributions are described and detailed.

1.4 Structure of the work

Chapter 2 presents the basic concepts regarding digital images and the theory of graphs needed to understand IFT and the solutions developed. Chapter 3 describes the IFT framework, including its derived methods OIFT, ORFC and IFT-SLIC. Chapter 4 is about the robustness analysis of the OIFT and ORFC methods, with their theorems and proofs. Chapter 6 describes the development of interactive segmentation repair methods using the three approaches mentioned above, as well as their experimental results. The Chapter 5 shows a hybrid method which improves *Graph Cut* by using the cores of ORFC, and vice-versa, with experimental evaluations comparing it with other methods from the literature. Chapter 7 concludes the work and proposes extensions and future work.

Chapter 2

Background

Two fundamental concepts are treated throughout the text and defined in this chapter: digital images (Section 2.1) and graphs (Section 2.2). Although the application domain is image segmentation, as the methods described in this text deal with graphs, any other domain (e.g., social and biological networks), whose problem could be modeled by graphs, can benefit from these methods. This section also describes how to convert a digital image to a graph, concepts about segmentation, and evaluation of segmentation methods, important elements to comprehend solutions of the segmentation editing problem.

2.1 Digital Image

A *digital image* is a mapping $I : \mathcal{I} \rightarrow \mathcal{V}$, which assigns a vector $I(s) \in \mathcal{V}$ for a *space element* (*spel*) $s = (s_1, \ldots, s_n) \in \mathcal{I}$, where $\mathcal{I} \subset \mathbb{Z}^n$ is the image domain (space of coordinates) and $\mathcal{V} \subset \mathbb{Z}^m$ the space of characteristics (or intensities). $\mathcal{I}(K)$ returns the domain of a specific image K, as well as $\mathcal{V}(K)$ returns its values. If $n = 2$, s is denoted *picture element* (*pixel*). If $n = 3$, then s is denoted *volume element* (*voxel*), and the image is *three-dimensional* (3D). Generally speaking, if $n > 1$, the image is *multidimensional*. If $m > 1$, the image is *multichannel* or *multiband*. Medical images generated by MR devices are examples of multidimensional images. Colored images are examples of multichannel ones (Figure 2.1).

Figure 2.1: *Digital Image: (a) 3D MR image of 1 channel, (b) 2D colored image with 3 channels (RGB) and its (c) space of characteristics*

2.2 Graph

A *weighted graph* is a tuple $G = \langle V, E, \omega \rangle$, where V (or explicitly $V(G)$) is a set of *nodes*, $E \subseteq V \times V$ (or explicitly $E(G)$) is a set of *arcs* and $\omega : E \to \mathbb{R}$ assigns a *weight* $\omega(\langle s,t \rangle)$ for each arc $\langle s,t \rangle \in E$, with $s,t \in V$. If $\langle s,t \rangle \in E$, then t is *adjacent* to s. In an *oriented* graph, also called *digraph*, $\langle s,t \rangle$ and $\langle t,s \rangle$ are distinct (they are ordered pairs), as opposed to a *undirected* graph. The *transpose* graph $G^T = \langle V, E^T, \omega^T \rangle$ of G is the unique digraph where $E^T = \{\langle t,s \rangle : \langle s,t \rangle \in E\}$ and $\omega^T(\langle s,t \rangle) = \omega(\langle t,s \rangle)$. In a *symmetric* graph, $E = E^T$, that is, $\langle s,t \rangle \in E \Leftrightarrow \langle t,s \rangle \in E$. Figure 2.2 shows examples of graphs.

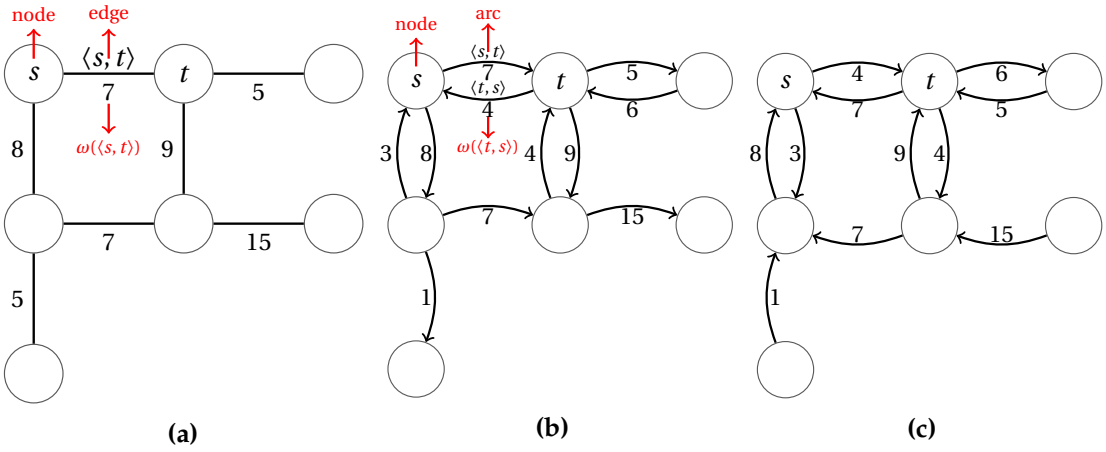


Figure 2.2: *Weighted graph: (a) non-oriented, (b) digraph and (c) transpose of (b).*

2.2.1 Path, Predecessor Map, Forest and Root

A *path* $\pi_{s \rightsquigarrow t} = \langle s = t_1, t_2, \ldots, t_n = t \rangle$ is a sequence of adjacent nodes, where s represents the *origin* and t the *terminus*. $\Pi_{s \rightsquigarrow t}$ is the set of all paths in G from s to t, $\Pi_t = \bigcup_{s \in V} \Pi_{s \rightsquigarrow t}$ is the set of all paths π_t with terminus t and $\Pi = \bigcup_{t \in V} \Pi_t$. Let $\Pi(G)$ be all possible paths in G. Consider also $\pi_{\mathcal{S} \rightsquigarrow t} \in \Pi_{\mathcal{S} \rightsquigarrow t} = \{\pi_{s \rightsquigarrow t} : s \in \mathcal{S}\}$, for any $\mathcal{S} \subseteq V$. A path is *trivial* when $\pi_t = \langle t \rangle$. A path $\pi_t = \pi_s \cdot \langle s,t \rangle$ represents an extension of a *prefix* π_s by an arc $\langle s,t \rangle$ and $\pi_t = \pi_s \cdot \pi_{s \rightsquigarrow t}$ the extension by another path. A *predecessor map* is a function $Pr : V \to V \cup \{nil\}$ where $\forall t \in V, Pr(t) = s$ if $\langle s,t \rangle \in E$, otherwise $Pr(t) = nil$. A *spanning forest* is a predecessor map without cycles. The *roots* of the forest are nodes $R^{Pr} = \{r \in V : Pr(r) = nil\}$. Let π_t^{Pr} be defined recursively as $\langle t \rangle$ if $t \in R^{Pr}$, or $\pi_s^{Pr} \cdot \langle s,t \rangle$ otherwise. Figure 2.3 illustrates these concepts.

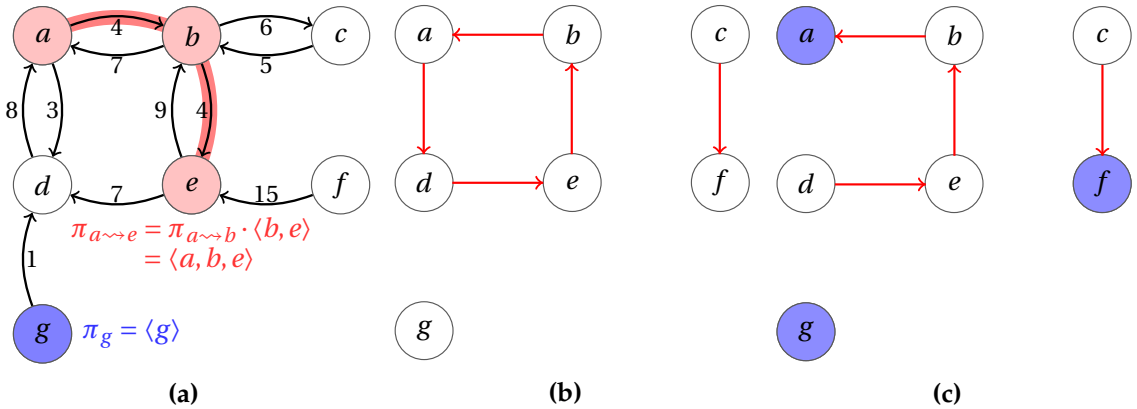


Figure 2.3: *(a) Trivial (π_g) and extended ($\pi_{a \rightsquigarrow e}$) paths, (b) predecessor map (red arrows indicating the predecessors), with cycle $\langle d, a, b, e \rangle$ and (c) spanning forest Pr, roots $R^{Pr} = \{a, f, g\}$ (blue circles) and paths π^{Pr} (red arrows).*

2.2.2 Component

Let $DCC_G(s) = \{t \in V : \exists \pi_{s \rightsquigarrow t} \in \Pi(G)\}$ be the *Directed Connected Component* of base $s \in V$, the set of all successors of s, and $SCC_G(s) = \{t \in V : \exists \{\pi_{s \rightsquigarrow t}, \pi_{t \rightsquigarrow s}\} \subseteq \Pi(G)\}$ the *Strongly Connected Component* of s, the set of pairwise nodes connected by paths. DCC and SCC may be related as: $SCC_G(s) = \{t \in V : s \in DCC_G(t) \text{ and } t \in DCC_G(s)\}$. A known and efficient (linear complexity) algorithm to find SCCs of a graph was developed by Tarjan [42], which visits neighbors of the current node (in a depth-first search way, which requires a stack) to find cycles (updating each *lowlink* to be the lowest *index* of the component) for each node, its neighbors are visited until a cycle is found it visits neighbors. Algorithm 1 shows Tarjan' SCC method and Figure 2.4 illustrates the steps.

Algorithm 1: Tarjan' Strongly Connected Components

```
1  i ← 0 and S ← empty array
2  For Each s ∈ V(G) Do
3      If s.index = nil Then strongconnect(s)
4  Function strongconnect(s)
5      s.index ← i, s.lowlink ← i and i ← i + 1
6      S.push(s) and s.onStack ← true
7      For Each ⟨s, t⟩ ∈ E(G) Do
8          If t.index = nil Then strongconnect(t) and s.lowlink ← min(s.lowlink, t.lowlink)
9          Else If t.onStack Then s.lowlink ← min(s.lowlink, t.index)
10     If s.lowlink = s.index Then
11         component ← new SCC
12         Do
13             t ← S.pop() and t.onStack = false
14             component.push(t)
15         While t ≠ s
16         store or just output component
```

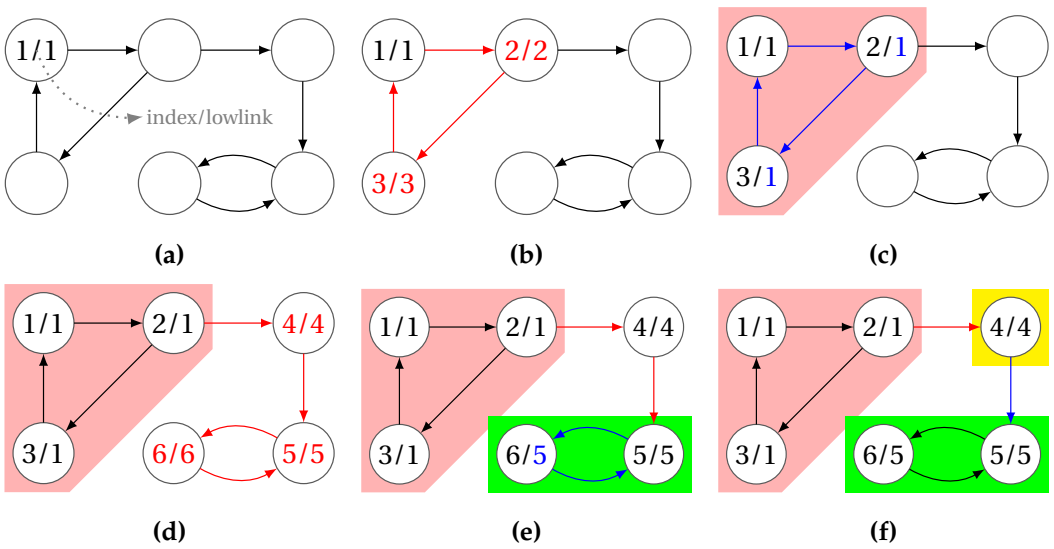


Figure 2.4: *Example of finding Strongly Connected Components by Tarjan algorithm: (a) for each non-processed node s, (b) we recursively apply strongconnect(s) (red arrows) by propagating it to its neighbors and setting their index and lowlink variables (red numbers). (c) If a cycle is found, we reverse the flow (blue arrows), updating lowlink until we find starting node (index = lowlink), creating a SCC. (d) As we continue the process, (e) we see that not all parents are included in the next new SCC. f) The final result.*

2.2.3 Connectivity Function and Optimum Path

A *connectivity function* $f : \Pi \to \mathbb{R}$ assigns a value for a path $\pi \in \Pi$. A path π_t is *optimum* if $f(\pi_t) \geq f(\pi'_t), \forall \pi'_t \in \Pi_t$, for a *maximization* (*primal*) problem. For a *minimization* (dual) one, $f(\pi_t) \leq f(\pi'_t), \forall \pi'_t \in \Pi_t$. Equality means there may be more than one optimum path. An optimum path to t is denoted as π^*_t. This generates a *connectivity map* $C_{opt} : V \to \mathbb{R}$ as $C_{opt}(t) = f(\pi^*_t)$. Figure 2.5 illustrates these concepts.

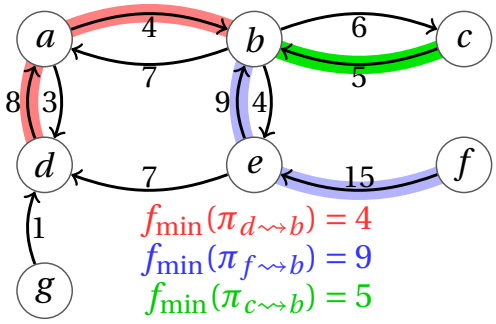


Figure 2.5: *Example of connectivity function (minimum arc value) for 3 different paths π_b. To compute π^*_b and $f(\pi^*_b)$, all paths should be calculated, but at least it will prefer $\pi_{f \rightsquigarrow b}$.*

An optimum path $\langle t_1, \ldots, t_n \rangle$ composed only by optimum prefixes $\langle t_1, \ldots, t_j \rangle$ $(1 \leq j \leq n)$ is a *complete-optimum prefix path*, or just *prefix-complete*. In a general case, an optimum path composed by optimum *subpaths* $\langle t_i, \ldots, t_j \rangle$ $(1 \leq i \leq j \leq n)$ is a *complete-optimum path*, or just *complete*. An *Optimum-Path Spanning Forest Problem* (OPSFP), consists on finding a *Optimum-Path Spanning Forest* (OPSF), a spanning forest Pr, so that π^{Pr}_t are optimum paths, for all $t \in V$, according to a connectivity function f. In an OPSF, $C_{opt}(t) = f(\pi^{Pr}_t), \forall t \in V$, and π^{Pr}_t is complete (and prefix-complete). A path connectivity may be based on its subpaths. This work uses functions defined recursively, as shown in Table 2.1, where $H(s)$ is a *handicap* function for initialization of trivial paths.

Table 2.1: *Connectivity functions commonly used for paths*

function	**parameter**	
	$\pi_s = \langle s \rangle$	$\pi_{s \rightsquigarrow t} = \pi_{s \rightsquigarrow r} \cdot \langle r, t \rangle$
$f_{\min}$	$H(s)$	$\min\{f_{\min}(\pi_{s \rightsquigarrow r}), \omega(\langle r, t \rangle)\}$
f_{sum}		$f_{sum}(\pi_{s \rightsquigarrow r}) + \omega(\langle r, t \rangle)$
f_{euc}		$d_{euc}(s,t) = \sqrt{\sum_{i=1}^{n}(s_i - t_i)^2}$
f_w		$\omega(\langle r, t \rangle)$

It can be checked that, for f_{max} or f_{sum}, at least one OPSF can be generated from any graph with non-negative weights. This also happens for any other *Monotonically Incremental* (MI) functions, which satisfies

$$\begin{aligned} f(\langle t \rangle) &= H(t), \\ f(\pi \cdot \langle s, t \rangle) &= f(\pi) \odot \langle s, t \rangle \end{aligned} \tag{2.1}$$

where $\odot : \mathbb{R} \times E \to \mathbb{R}$ is a binary operation that satisfies the conditions:

- **M1**: $x' \geq x \implies x' \odot \langle s, t \rangle \geq x \odot \langle s, t \rangle$,
- **M2**: $x \odot \langle s, t \rangle \geq x$.

Finding an OPSF is not restricted to MI functions. There are a more general class of functions called *smooth* functions [27] which generate at least one OPSF.

2.3 Image as a Graph

To get a graph from an image, it is necessary to establish a relation between *spels*. An *adjacency relation* $\mathcal{A}$ is a binary relation in $\mathcal{I}$. Let $E(s) = \{t \in \mathcal{I} : s\mathcal{A}t\}$ and $E = \bigcup_{s\in\mathcal{I}}\{\langle s,t\rangle : t \in E(s)\}$. Commonly used relations are those defined by metric distances. For example, set $s\mathcal{A}t$ as true if the euclidean distance $d_{\text{euc}}(s,t) = \sqrt{\sum_{i=1}^{n}(s_i - t_i)^2} \leq \rho$, where ρ is a constant ($\rho = 1$ for neighboring-4 *grid graph* and $\rho = \sqrt{2}$ for neighboring-8 *king graph*), as Figure 2.6.

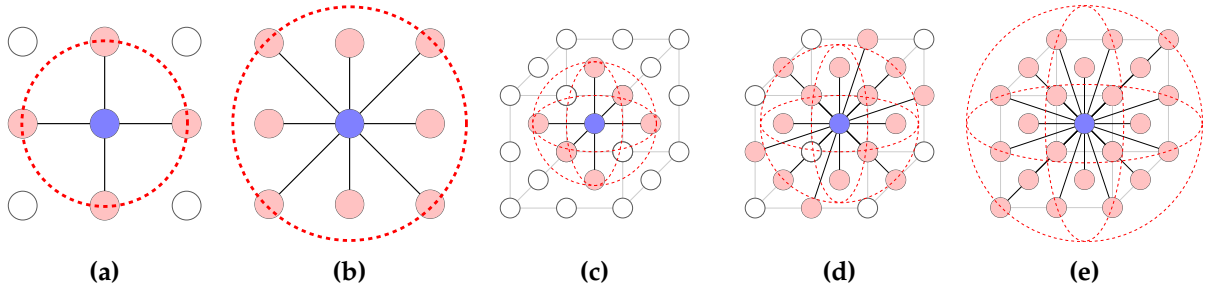


Figure 2.6: *Adjacency relation based on euclidean distance: (a) neighboring-4 ($\rho = 1$) and (b) neighboring-8 ($\rho = \sqrt{2}$). In 3D: (c) neighboring-6 ($\rho = 1$), (d) neighboring-18 ($\rho = \sqrt{2}$) and (e) neighboring-26 ($\rho = \sqrt{3}$).*

After choosing the adjacency relation, the image could be represented by a weighted digraph $G = \langle V, E, \omega\rangle$, where $V = \mathcal{I}$ and $E \subseteq V^2$, based on adjacency relation. The weight $\omega(\langle s,t\rangle)$ may be based on $I(s)$ and $I(t)$ as follows:

$$\omega(\langle s,t\rangle) = \begin{cases} \delta(s,t) \times (1-\alpha) & \text{if } I(s) > I(t) \\ \delta(s,t) \times (1+\alpha) & \text{if } I(s) < I(t) \\ \delta(s,t) & \text{otherwise} \end{cases} \tag{2.2}$$

where $\alpha \in [-1,1]$ is an orientation factor and $\delta(s,t) = \delta(t,s)$ is a measure of non-oriented similarity (i.e., $\delta(s,t) = K - \|I(s) - I(t)\|$, where K is the maximum intensity variation) [43,44]. If $\alpha = 0$, then $\omega(\langle s,t\rangle) = \omega(\langle t,s\rangle)$ (a non-oriented graph). Figure 2.7 shows an example. The methods in this work can be applied to any graph, not only those obtained from images.

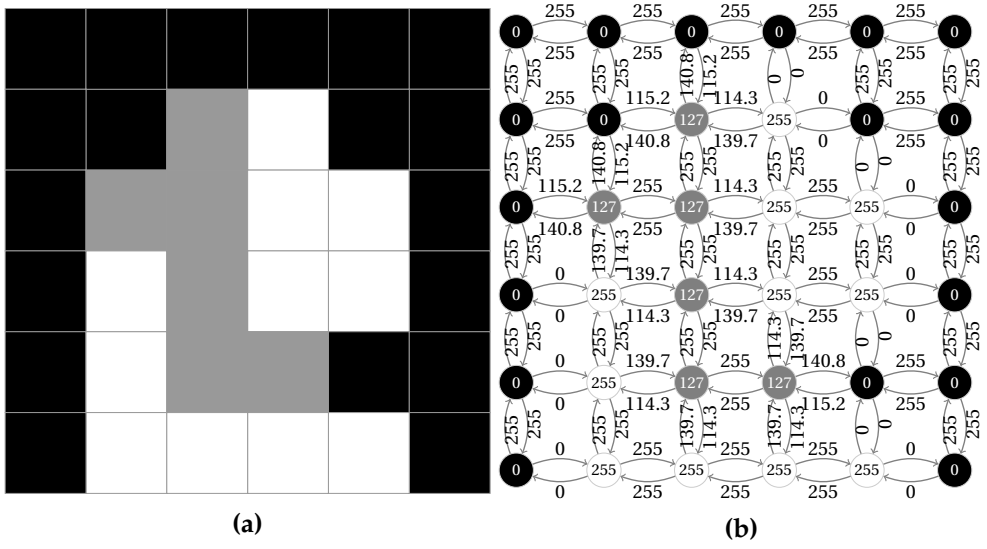


Figure 2.7: *Example of image converted to a graph: (a) 8-bit image with 3 gray levels (0, 127 and 255), (b) graph with oriented weight and $\alpha = 0.1$, with low values in high contrast transitions (e.g., black and white nodes) and discerning transitions in opposed directions (e.g.,gray/white nodes).*

2.4 Segmentation: Binary Object, Seeds, Algorithm and Energy

Given c *classes*, a *label map* $L : V \to \mathcal{L}$ ($\mathcal{L} = \{l_1, \ldots, l_c\}$) defines a *partition* $\mathcal{P}_L = \{P_1, \ldots, P_c\}$, where $\bigcup_{i=1}^{c} P_i = V$ and P_i are *regions* where $L(t) = l_i, \forall t \in P_i$. A binary partition $\{\mathcal{O}, V \setminus \mathcal{O}\}$ can be represented by the *binary segmented object* $\mathcal{O}$, with $\mathcal{L} = \{l_o, l_b\}$ (in general $l_b = 0$ and $l_o = 1$), also called *mask*. Let $\mathcal{X}$ be the space of all possible objects. A *seed-based* segmentation uses *seeds* $\mathcal{S} = \mathcal{S}_o \cup \mathcal{S}_b \subseteq V$, where $\mathcal{S}_o$ and $\mathcal{S}_b$ are *internal* ($\mathcal{S}_o \subseteq \mathcal{O}$) and *external* ($\mathcal{S}_b \subseteq V \setminus \mathcal{O}$) ones, respectively. They reduce $\mathcal{X}$ to $\mathcal{X}(\mathcal{S}_o, \mathcal{S}_b) = \{\mathcal{O} \in \mathcal{X} : \mathcal{S}_o \subseteq \mathcal{O} \subseteq V \setminus \mathcal{S}_b\}$. A *segmentation algorithm* $A(\mathcal{S}_o, \mathcal{S}_b) \in \mathcal{X}(\mathcal{S}_o, \mathcal{S}_b)$ divides G into $\mathcal{O}$ and $V \setminus \mathcal{O}$. Primal seed-based OPSFP require trivial connectivity values for seeds to be higher (lower for dual ones) than other nodes, so that root set $R^{Pr} \subseteq \mathcal{S}$.

A *cut* is defined as $\mathcal{C}(\mathcal{O}) = \{\langle s,t \rangle \in E : s \in \mathcal{O} \text{ and } t \notin \mathcal{O}\}$. An *energy* $\varepsilon : \mathcal{X} \to \mathbb{R}$ can be assigned to an object (and its cut), so that we can restrict a set of solutions to those which minimize it. A class of energies often used is *q-norm*, where $\varepsilon_q(\mathcal{O}) = (\sum_{\langle s,t \rangle \in \mathcal{C}(\mathcal{O})} \omega(\langle s,t \rangle)^q)^{\frac{1}{q}}$. If $q = \infty$, then $\varepsilon_\infty(\mathcal{O}) = \max_{\langle s,t \rangle \in \mathcal{C}(\mathcal{O})} \omega(\langle s,t \rangle)$. A ε_q*-minimizer* returns $\mathcal{O}$ with the lowest $\varepsilon_q(\mathcal{O})$. Figure 2.8 illustrates these concepts.

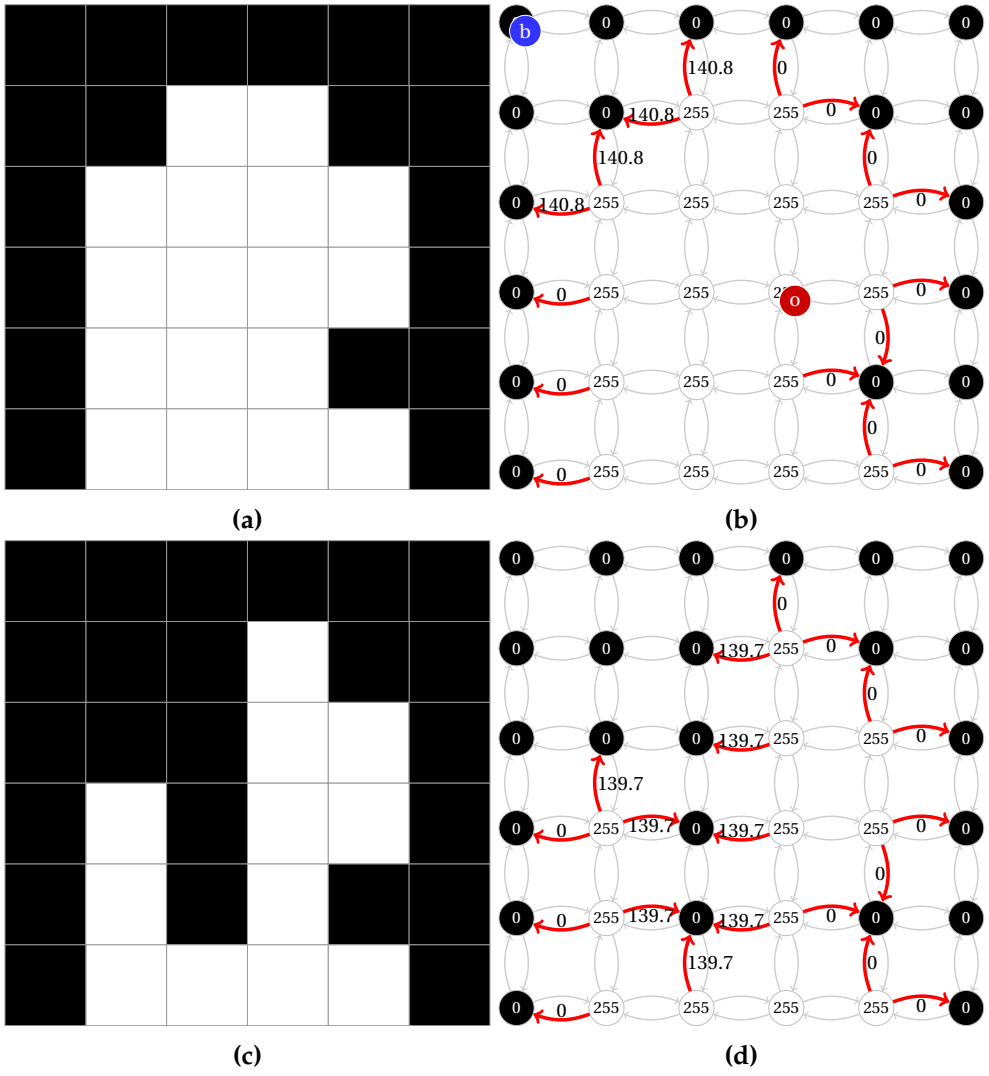


Figure 2.8: *Elements of a segmentation: (a) binary segmentation and (b) graph with binary object $\mathcal{O}_1$ (set of nodes with label 255), cut $\mathcal{C}(\mathcal{O}_1)$ highlighted (red arcs), external $\mathcal{S}_b$ (blue) and internal $\mathcal{S}_o$ (red) seeds used in algorithm $A(\mathcal{S}_o, \mathcal{S}_b)$, and energies $\varepsilon_2(\mathcal{O}_1) = 314,83$ and $\varepsilon_\infty(\mathcal{O}_1) = 140.8$, c-d) other segmentation and object $\mathcal{O}_2$ with $\varepsilon_\infty(\mathcal{O}_2) = 139.7$. An ε_∞-minimizer will prefer $\mathcal{O}_2$.*

2.5 Evaluation of Methods

The performance of a method can be defined in many ways: complexity, running time, accuracy, space consuming, etc. The *accuracy* is the degree of similarity between the gotten result $\mathcal{O}$ and desired one $\mathcal{G}$, denominated *groundtruth*. Usually it is in the range [0,1] or [-1,1], whose ends represent the worst (0 or -1) and best (1) possible accuracy.

In a binary graph segmentation (object and background), to specify the modifications between the result $\mathcal{O}$ and groundtruth $\mathcal{G}$, the *True Positives* (TP), *True Negatives* (TN), *False Positives* (FP) and *False Negatives* (FN) are defined, where TP and TN are respectively nodes labeled correctly as object and background. FP represents the background nodes in $V \setminus \mathcal{G}$, labeled as object in $\mathcal{O}$, and FN represents object nodes in $\mathcal{G}$, labeled as background in $V \setminus \mathcal{O}$. TP, TN, FP and FN compose the *confusion matrix* of Table 2.2. Equation 2.3 formulates these definitions and Figure 2.9 illustrates in a visual way.

$$TP = \mathcal{G} \cap \mathcal{O} \quad TN = V \setminus (\mathcal{G} \cup \mathcal{O}) \quad FP = \mathcal{O} \setminus \mathcal{G} \quad FN = \mathcal{G} \setminus \mathcal{O} \tag{2.3}$$

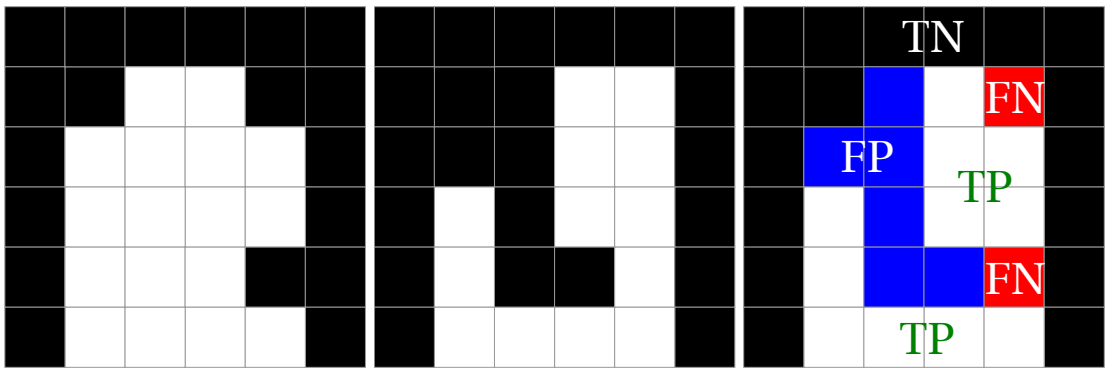


Figure 2.9: *Confusion matrix over a segmentation: (a) obtained result, (b) desired result (groundtruth), (c) regions indicating True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN).*

Table 2.2: *Confusion matrix*

		Obtained result	
		Object (positive)	**Background (negative)**
Groundtruth	**Object (positive)**	**TP** True Positive correctly labeled as object	**FN** False Negative incorrectly labeled as background
	Background (negative)	**FP** False Positive incorrectly labeled as object	**TN** True Negative correctly labeled as background

A metric used in this work for determining the accuracy of a method is *Dice coefficient* [45], as known as *F-score* or *F-measure* [46]. Given the groundtruth $\mathcal{G}$ and the result $\mathcal{O}$ over a graph, the Dice coefficient is calculated in Equation 2.4. If both regions are similar, the intersection, and consequently the numerator of the fraction, has high cardinality, coming close to the union one. When both objects are identical, the union and intersection also are, justifying the factor 2. Note that TN, and consequently the graph size, does not influence the metric.

$$D(\mathcal{O},\mathcal{G}) = \frac{2 \times |\mathcal{O} \cap \mathcal{G}|}{|\mathcal{O}| + |\mathcal{G}|} = \frac{2 \times |\mathcal{O} \cap \mathcal{G}|}{|\mathcal{O} \cup \mathcal{G}| + |\mathcal{O} \cap \mathcal{G}|} = \frac{2 \times |TP|}{2 \times |TP| + |FP| + |FN|} \tag{2.4}$$

2.5.1 Seeds in Evaluation: Robot Users and Erosion

Altering the seed set $\mathcal{S}$ can also be subject for evaluation. One approach used for evaluating interactive segmentation methods is the *robot user* [41], which consists on automatic simulation of seed inclusion process. The method runs after each modification for evaluating intermediate results and groundtruth. Wrong regions (false negative and false positive) are treated (by adding seeds) to raise accuracy, and it generates an accuracy curve. The addition may be done in the center of the region (*geodesic*), at weaker edges (*pixel robot*) or next other class regions (*superpixel robot*) [47]. Figure 2.10 illustrates robot user.

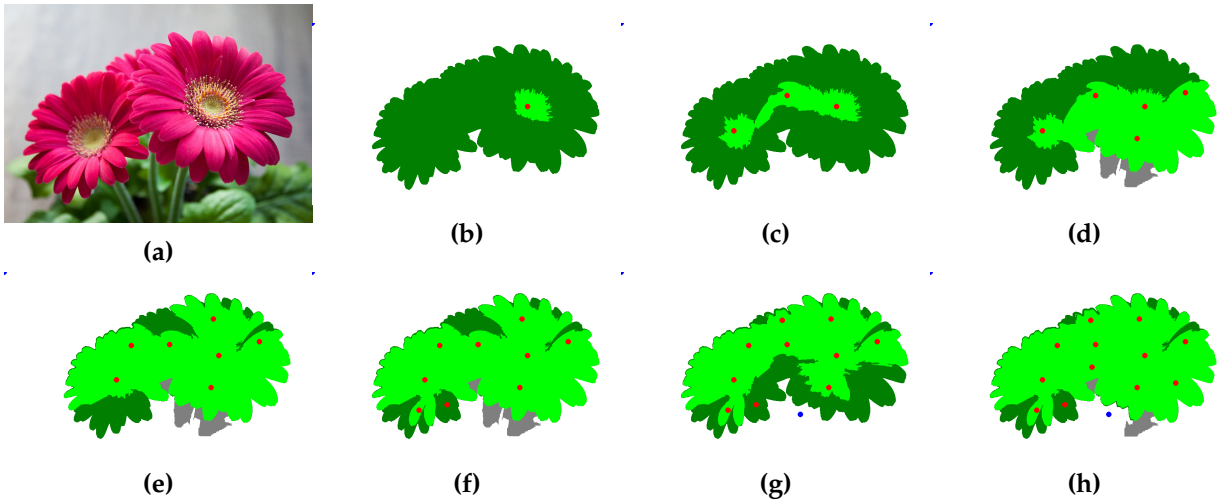


Figure 2.10: *Evaluation with robot user: (a) image, (b-h) false negative (dark green), true negative (white), false positive (grey) and true positive (light green) when adding 1-pixel seeds in a geodesic manner (center of biggest inscribed circle of FT and FN regions, with object (red) and background seeds (blue). Lighter (in this picture) means more accurate.*

Another approach is *erosion* [48] of groundtruth $\mathcal{G}$, which undergoes a process of erosion and dilation (or erosion of $V \setminus \mathcal{G}$), through parameter *radius*, which can be different for each class. The border of each resulting object is used as seeds. The method is executed, evaluated and the process is repeated, which generates an accuracy curve. Figure 2.11 illustrates the erosion.

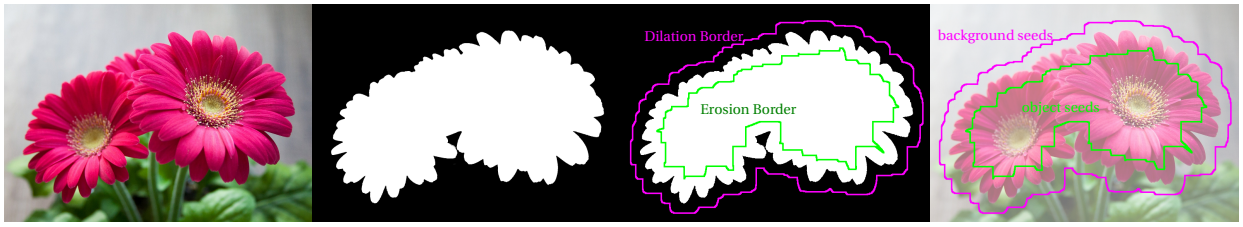


Figure 2.11: *Evaluation with erosion: (a) image, (b) groundtruth, (c) borders of erosion and dilation of object, (d) example of borders as background and object seeds. The radius defines the degree of segmentation flexibility.*

Chapter 3

Image-Foresting Transform

Image-Foresting Transform (IFT) [27] is an extension of Dijkstra shortest path algorithm [49, 50], contemplating multiple sources and different connectivity functions. Several image operators could be constructed from the same algorithm, favoring implementation in hardware [51] and helping to define relations between them. IFT can be used in: distance transform, multiscale skeleton, fractal dimensions, shape filtering, shape salience detection, shape descriptors, geodesic paths [27], morphological reconstruction [52], *Watershed* [19], *Live Wire* [53], *Riverbed* [54], *Grow-cut* by cellular automata [28, 55], fuzzy connectivity [56], data clustering [57] and supervised classification [58].

IFT solves OPSFP by receiving a graph $G = \langle V, E, \omega \rangle$, a connectivity function f, a seed set $\mathcal{S} \subseteq V$ and calculates a label map $L : V \to \mathcal{L}$ ($\mathcal{L} \subset \mathbb{N}$ represents the classes) and a connectivity map $C : V \to \mathbb{R}$ defined by a optimum-path spanning forest Pr, which converges to $C_{opt}(t) = f(\pi_t^*), \forall t \in V$, when f is smooth [27]. Algorithm 2 formulates IFT for maximization (primal) problem. For solving minimization (dual) problems, it is enough to replace $best > C(t)$ by $best < C(t)$. In a binary segmentation, $\mathcal{L} = \{l_b, l_o\}$ (often $\{0, 1\}$), the label L defines the object $\mathcal{O}$ (because $s \in \mathcal{O} \Leftrightarrow L(s) = l_o$) and $\mathcal{S} = \mathcal{S}_b \cup \mathcal{S}_o$.

Algorithm 2: Image-Foresting Transform (IFT)

INPUT : Graph $G = (V, E, w)$, seed set $\mathcal{S} \subseteq V$ and connectivity function $f : \Pi(G) \to \mathbb{R}$.
OUTUT : Forest $Pr : V \to V \cup \{nil\}$, Connectivity Map $C : V \to \mathbb{R}$ and Label Map $L : V \to \mathcal{L}$.
AUXILIAR: Priority Queue Q, variable $best$, and State Map $St : V \to \{0, 1\}$.

1 **For Each** $s \in V$, **Do**
2 $St(s) \leftarrow 0, Pr(s) \leftarrow nil$ and $C(s) \leftarrow f(\langle s \rangle)$
3 **If** $s \in \mathcal{S}$, **Then** add s in Q and initialize $L(s)$.
4 **While** $Q \neq \emptyset$, **Do**
5 Remove s from Q whose value $C(s)$ is the minimum and assign $St(s) \leftarrow 1$.
6 **For Each** t, *where* $\langle s, t \rangle \in E$ *and* $St(t) = 0$, **Do**
7 $best \leftarrow f(\pi_s^P \cdot \langle s, t \rangle)$.
8 **If** $best > C(t)$, **Then**
9 **If** Q *contains* t, **Then** remove t from Q.
10 $Pr(t) \leftarrow s, C(t) \leftarrow best, L(t) \leftarrow L(s)$ and add t in Q.

3.1 An illustrative example

Due to flexibility and robustness of IFT, a simple example can help clarifying it. Let a primal IFT with $f_{\min}$ and three classes applied to a graph of Figure 3.1. The map C (illustrated in the center of each node) are initialized and $\mathcal{S}$ is added to the priority queue Q. In each step, let s be the first element of Q. s is removed and processed (indicated by red/black border for current/next steps, respectively), so that $L(s)$ (indicated by a colored square) does not change anymore (the method finishes when $|V|$ steps are processed and the whole L are fixed).

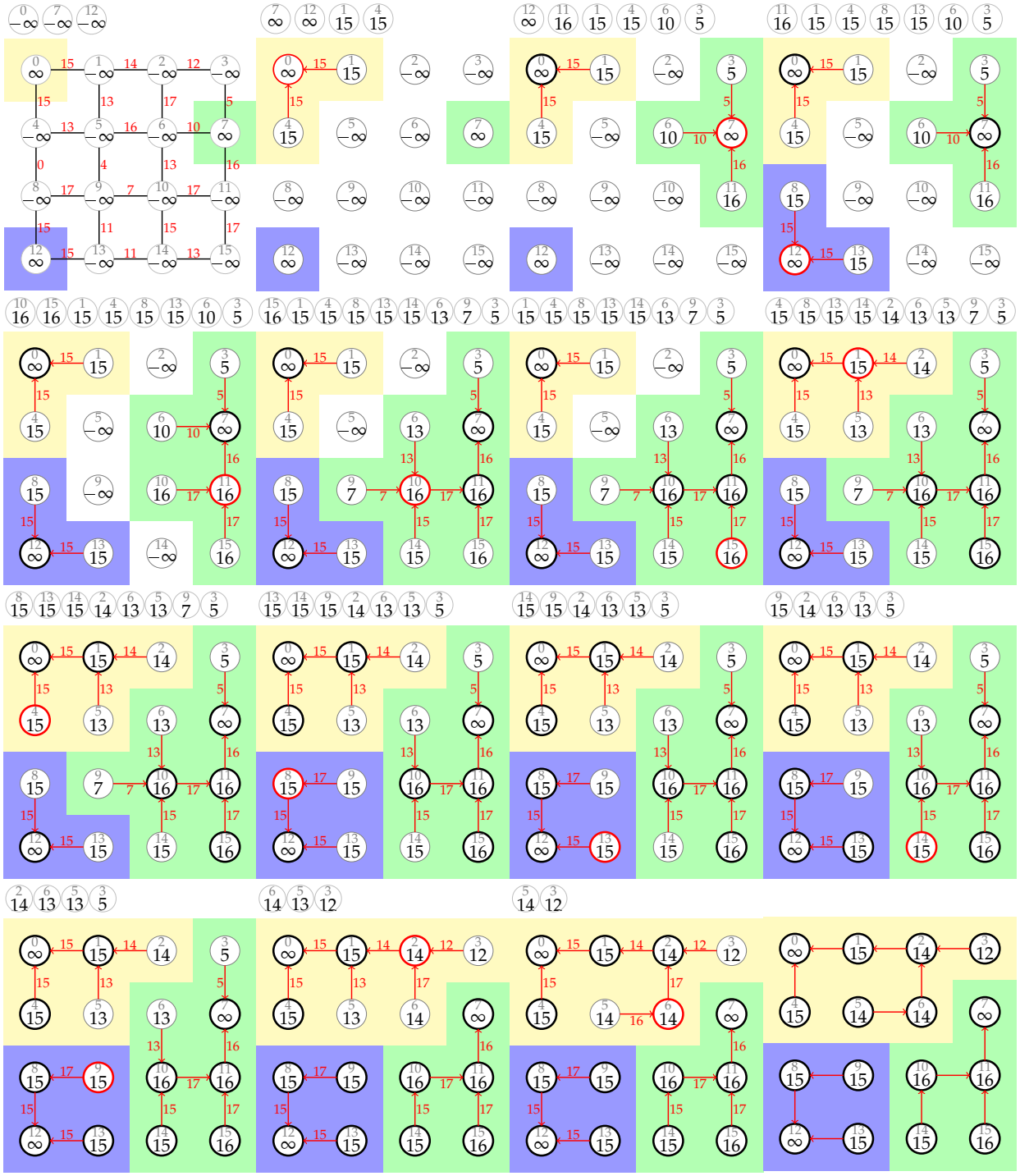


Figure 3.1: *Illustration of primal IFT ($f_{\min}$ and $\alpha = 0$) execution: priority queue Q (top nodes in each figure), labels L (colored squares), node index (grey labels), connectivity C (black numbers inside circle), edges, weights, descendants (red arrows), processing node (red border), old processed ones (black border). At the end we have an OPSF with $C = C_{opt}$ ($f_{\min}$ is smooth). Step 15 is omitted.*

For each non-processed neighbor t of s, s offers its best path (π_s^P) extended by the arc $\langle s,t\rangle$. If its connectivity $f_{\min}(\pi_s^P \cdot \langle s,t\rangle)$ is better than $C(t)$, $C(t)$ is updated and t becomes descendant of s ($P(t) \leftarrow s$, indicated by a red arc in Figure 3.1), as well as its label ($L(t) \leftarrow L(s)$). t is added to Q in the correct position (sorted by C). The method has linear complexity in respect to $|V|$.

With a smooth function, the final values of the connectivity map only decrease as it moves away from the initial nodes (in this case, the seeds). At each step, the next connectivity values are always less or equal than the connectivity of the first node of the priority queue (the algorithm does not have to "go back"), justifying its dynamic nature. The nodes do not enter the queue after being processed, justifying their linear complexity. Several works (Figure 3.2) have added features from the initial algorithm, such as boundary polarity [31, 32], shape constraints [59], connectedness [60] and star convexity [61]. This work investigates and uses two variations as part of the solution to the segmentation interactive repair problem: OIFT and ORFC.

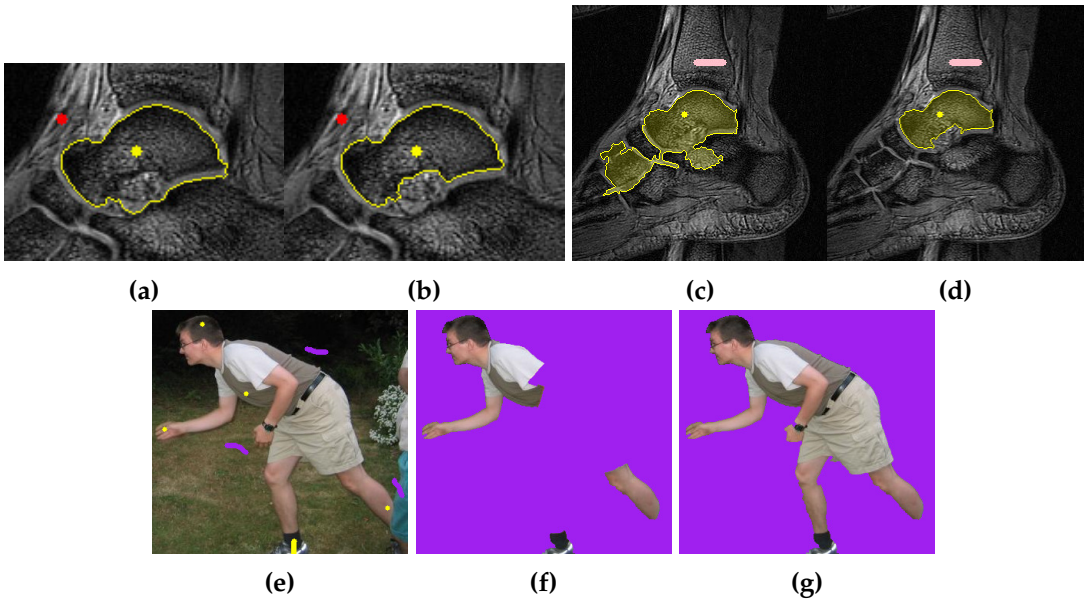


Figure 3.2: *Additional features added in traditional IFT formulation: (a-b) boundary polarity [31, 32] (favoring intensity transitions in one direction), (c-d) shape constraints [59] and (e-g) connectedness [60] (keeping results of object seeds connected).*

3.2 Oriented Image-Foresting Transform (OIFT)

OIFT is a ε_∞-minimization method [31, 32] build upon the IFT framework. It uses a connectivity function $f^{\sphericalangle}$ (Equation 3.1) in a symmetric digraph. In practice, OIFT uses weight values restricted to $\mathbb{N}$ and $f^{\sphericalangle}$ is also restricted to $\mathbb{N}$ in a bucket queue [27].

$$
\begin{aligned}
f^{\sphericalangle}(\langle t\rangle) &= \begin{cases} \infty & \text{if } t \in \mathcal{S}_o \cup \mathcal{S}_b \\ -\infty & \text{otherwise} \end{cases} \\
f^{\sphericalangle}(\pi_{r\rightsquigarrow s} \cdot \langle s,t\rangle) &= \begin{cases} \min\{f^{\sphericalangle}(\pi_{r\rightsquigarrow s}), 2\times\omega(\langle s,t\rangle)\} & \text{if } r \in \mathcal{S}_o \\ \min\{f^{\sphericalangle}(\pi_{r\rightsquigarrow s}), 2\times\omega(\langle t,s\rangle)+1\} & \text{otherwise} \end{cases}
\end{aligned}
\tag{3.1}
$$

For $\alpha > 0$ (Equation 2.2), OIFT favors intensity transitions from dark to bright, and $\alpha < 0$ favors the opposite orientation. We set odd connectivity values for $\pi_{\mathcal{S}_b\rightsquigarrow t}$ and even for $\pi_{\mathcal{S}_o\rightsquigarrow t}$ to avoid tie zones. The segmented object $A_{OIFT}(\mathcal{S}_o,\mathcal{S}_b)$ by OIFT is defined from the forest P computed by IFT with $f^{\sphericalangle}$, by taking as object all nodes conquered by paths rooted in $\mathcal{S}_o$, that is, $A_{OIFT}(\mathcal{S}_o,\mathcal{S}_b) = \{t \in V : \pi_t^P \in \Pi_{\mathcal{S}_o\rightsquigarrow t}\}$. Even though $f^{\sphericalangle}$ is not smooth, the optimality of $A_{OIFT}(\mathcal{S}_o,\mathcal{S}_b)$ is given by ε_∞-minimization problem.

3.3 Oriented Relative Fuzzy Connectedness (ORFC)

ORFC [33] is also a ε_∞-minimizer, which involves arcs from object to background. Let $\varepsilon_\infty^{\downarrow} = \min_{\mathcal{O}\in\mathcal{X}(\mathcal{S}_o,\mathcal{S}_b)}\{\varepsilon_\infty(\mathcal{O})\}$ and $\mathcal{X}_\infty^{\downarrow}(\mathcal{S}_o,\mathcal{S}_b) = \{\mathcal{O}\in\mathcal{X}(\mathcal{S}_o,\mathcal{S}_b) : \varepsilon_\infty(\mathcal{O}) = \varepsilon_\infty^{\downarrow}\}$. Equation 3.2 defines A_{ORFC} for seeds $\mathcal{S}_o$ and $\mathcal{S}_b$.

$$A_{ORFC}(\mathcal{S}_o,\mathcal{S}_b) = \left[\bigcup_{s_i\in\mathcal{S}_o} A_{ORFC}(\{s_i\},\mathcal{S}_b)\right], \text{ where } A_{ORFC}(\{s_i\},\mathcal{S}_b) = \underset{\mathcal{O}\in\mathcal{X}_\infty^{\downarrow}(\{s_i\},\mathcal{S}_b)}{\arg\min} |\mathcal{O}| \tag{3.2}$$

ORFC primal formulation (maximization) uses a connectivity function $f_{\min}^{\leftarrow}$ (Equation 3.3), a smooth function which processes reversed (antiparallel) arcs. *Relative Fuzzy Connectedness* (RFC) is a particular case of ORFC applied to non-oriented graphs (e.g., when $\alpha = 0$). Algorithm 3 demonstrates the computation of ORFC in a symmetrical digraph. Figure 3.3 illustrates ORFC. Although ORFC and OIFT are methods from the same energy class, their outputs are usually different with distinct characteristics (Figure 3.4). This kind of illustration represents a graph with invisible nodes (it could be any $|V|$), highlighting only important arcs (as borders and arrows). The shades are the labels of nodes.

$$f_{\min}^{\leftarrow}(\langle t\rangle) = \begin{cases}\infty & \text{if } t\in\mathcal{S}_b\\ -\infty & \text{otherwise}\end{cases} \qquad f_{\min}^{\leftarrow}(\pi_{r\rightsquigarrow s}\cdot\langle s,t\rangle) = \min\{f_{\min}^{\leftarrow}(\pi_{r\rightsquigarrow s}),\omega(\langle t,s\rangle)\} \tag{3.3}$$

Algorithm 3: Computing $A_{ORFC}(\{s_i\},\mathcal{S}_b)$

1 Get connectivity map C_{opt} with $f_{\min}^{\leftarrow}$ by IFT;
2 Create $G_> = (V,E',\omega)$ from $G = (V,E,\omega)$ where $E' = \{\langle s,t\rangle\in E : \omega(\langle s,t\rangle) > C_{opt}(s_i)\}$;
3 Return $DCC_{G_>}(s_i)$;

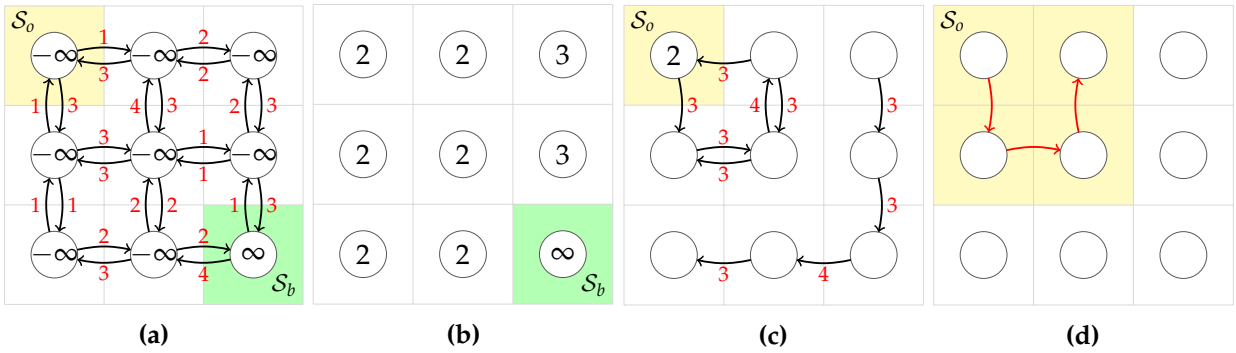


Figure 3.3: *Segmentation with ORFC from Algorithm 3: (a) graph and seeds, (b) Step 1, with C_{opt} from $\mathcal{S}_b$, (c) Step 2 and $G_>$ with only $\langle s,t\rangle$ where $\omega(\langle s,t\rangle) > C_{opt}(\mathcal{S}_o)$ and (d) $DCC_{G_>}$*

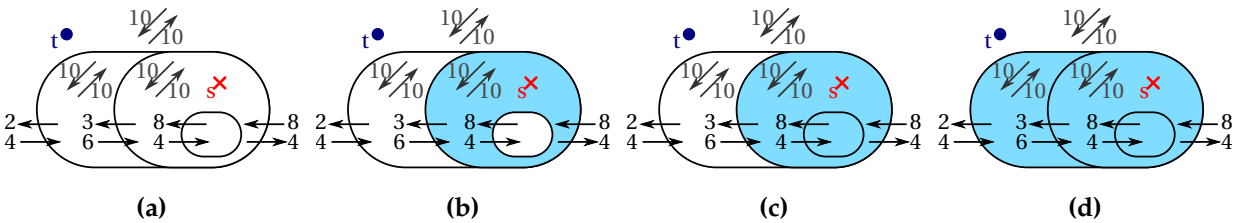


Figure 3.4: *(a) Input image graph with $\mathcal{S}_o = \{s\}$ and $\mathcal{S}_b = \{t\}$. (b) ORFC result. (c) A candidate solution. (d) OIFT result. Note that all the three solutions have the same energy $\varepsilon_\infty(\mathcal{O}) = 4$.*

3.4 IFT-SLIC

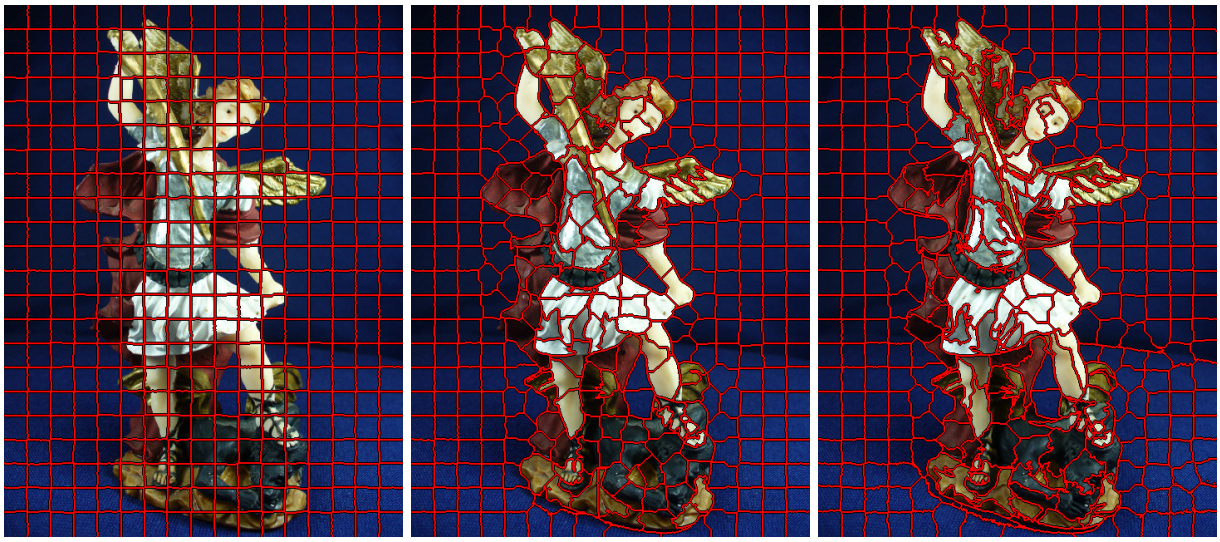
IFT-SLIC [34] combines benefits of both IFT and SLIC [62] to provide a more regular and powerful partition generation. IFT-SLIC was formulated according to the following requirements:

- Ability to adhere to image boundaries: respecting and preserving local structures;
- Flexibility in the number of superpixels it generates: preventing undersegmentation;
- Efficiency: fast running also for extending to higher dimensions;
- Hard segmentation: supernodes should not overlap each other;
- Compactness: regular and uniform shape.

Similar to SLIC, we convert the image color space to LAB and start with a selection of k initial cluster centers $Cl_i = [l_i\ a_i\ b_i\ x_i\ y_i]^T$, which are sampled on a *regular grid* spaced $\sqrt{|V|/k}$ nodes apart (Figure 3.6a). The main difference with SLIC lies in the *assignment* step. Instead of using an adaptive k-means clustering approach, we consider the computation of a dual IFT (minimization) with the non-smooth [63] connectivity function f_D (Equation 3.4):

$$\begin{aligned} f_D(\pi_t = \langle t \rangle) &= \begin{cases} 0, & \text{if } t \in \mathcal{S} \\ +\infty, & \text{otherwise} \end{cases} \\ f_D(\pi_{r \rightsquigarrow s} \cdot \langle s,t \rangle) &= f_D(\pi_{r \rightsquigarrow s}) + \underbrace{(\|I(t) - I(r)\| \cdot \delta)^\beta}_{\text{Boundary Adherence}} + \underbrace{d_{euc}(s,t)}_{\text{Compactness}} \end{aligned} \tag{3.4}$$

where $I(t)$ is the color vector at voxel t, i.e., $I(t) = [l_t\ a_t\ b_t]^T$, $I(r)$ is the color vector of the cluster center of seed r, and δ and β weights the importance between boundary adherence and compactness. At the end of the assignment step, each cluster/supervoxel is represented by its respective tree in the spanning forest computed by the IFT (Figure 3.6b). After that, an *update* step adjusts the cluster centers. Differently from SLIC, we take the coordinate of the cluster voxel closest to the mean position (Figure 3.6c). The assignment and update steps are iterated until n steps or based on another stop criteria (Figure 3.6d). Different values of δ affect the adhesion of clusters to the image boundaries (Figure 3.5).



(a) **(b)** **(c)**

Figure 3.5: *Effect of δ in the IFT-SLIC result: (a) $\delta = 0.01$, (b) $\delta = 0.04$ and (c) $\delta = 0.08$. Note the difference of image boundary adherence effect.*

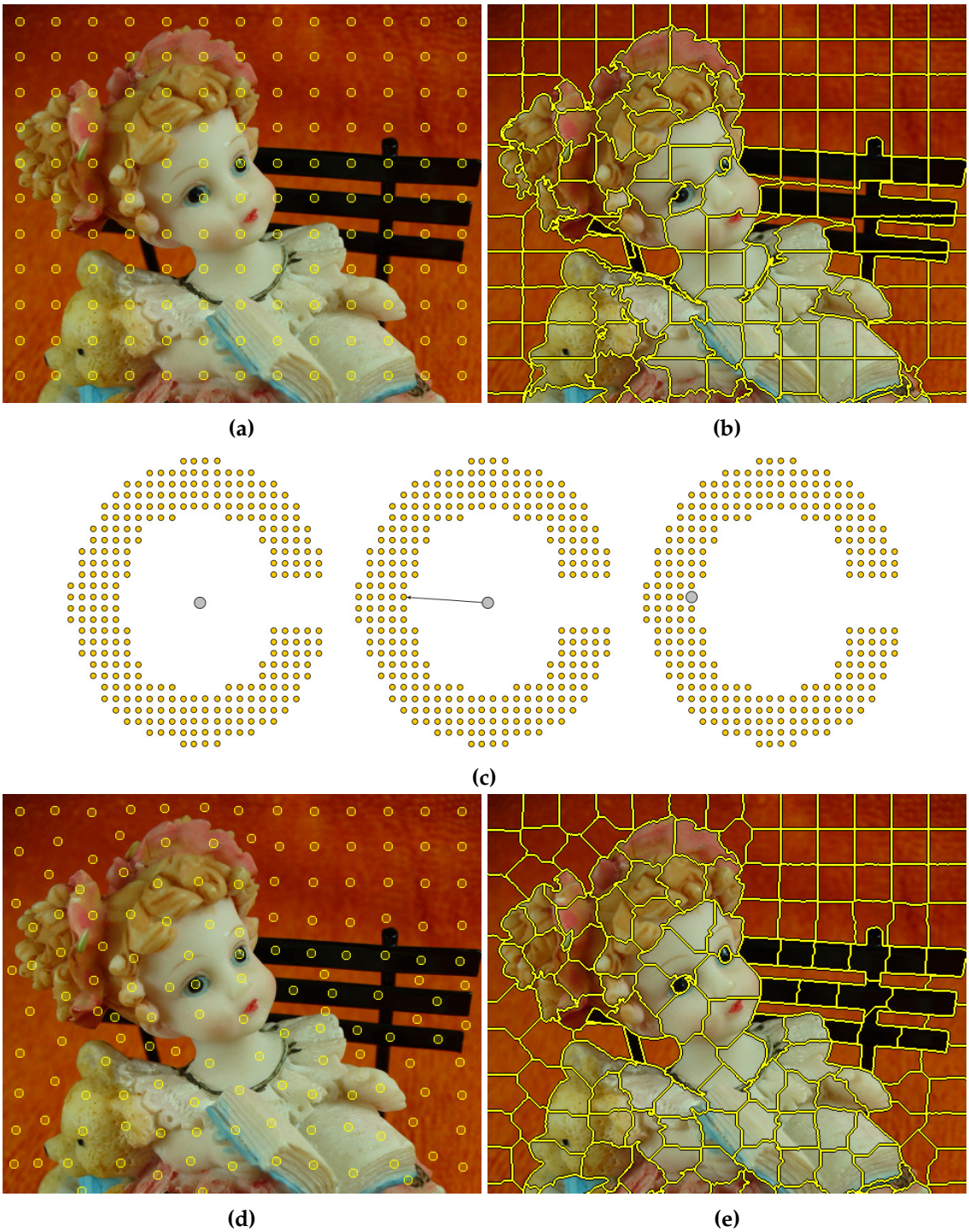


Figure 3.6: *IFT-SLIC process: (a) starting with a regular grid of cluster centers (seeds), we run dual IFT (minimization) with non-smooth connectivity function f_D, (b) resulting in a compact and regular partition, (c) and we move the seeds to a position inside the cluster closest to the mean. (d-e) The result after 10 iterations.*

Chapter 4

Seed Robustness Analysis

In this chapter, we define concepts of robustness, equivalence and cores, formulate relations between OIFT and ORFC, algorithms to compute cores (Section 4.1 and Section 4.2) as well as defining a metric (Section 4.3) for robustness comparisons. Without loss of generality, we will constrain the analysis of robustness only to internal seeds, being the external seeds a completely symmetric problem. In order to define the concept of core, we must first introduce the notion of seed equivalence (Definition 1).

Definition 1. *(Equivalent seeds). Two internal seeds s_1 and s_2 are said* equivalent *if they separately produce the same result. That is, for a given external seed set $\mathcal{S}_b$, we have that $A(\{s_1\}, \mathcal{S}_b) = A(\{s_2\}, \mathcal{S}_b)$.*

The *equivalence relation* between s_1 and s_2 is denoted as $s_1 \equiv s_2$, a binary relation which is reflexive (s_1 produces same result as s_1), symmetric ($s_1 \equiv s_2 \implies s_2 \equiv s_1$) and transitive (if s_2 and s_3 produce same result as $s_1, s_2 \equiv s_3$) over $V \setminus \mathcal{S}_b$. This relation partitions $V \setminus \mathcal{S}_b$ into *equivalence classes* $[s] = \{t \in A(\{s\}, \mathcal{S}_b) : s \equiv t\}$, also denoted by *cores* $\mathcal{N}(\{s\}, \mathcal{S}_b) = [s]$. By fixing $\mathcal{S}_b$, to get an object composed by n cores, at most n internal seeds (one for each core) are necessary for segmenting it. Some theoretical relations between ORFC and OIFT, as well as their cores, are defined in the following propositions. Figure 4.1 illustrates the Proposition 1.

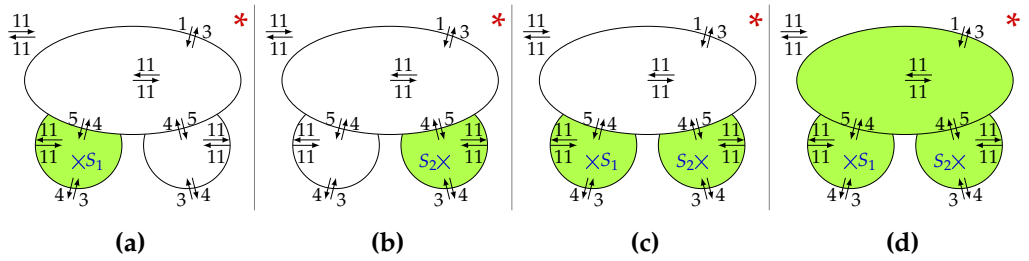


Figure 4.1: *Showing that $A_{ORFC}(\mathcal{S}_o, \mathcal{S}_b) \subseteq A_{OIFT}(\mathcal{S}_o, \mathcal{S}_b)$, with $\mathcal{S}_o$ ($\times$) in different positions and $\mathcal{S}_b$ ($\star$) fixed: (a) $A_{ORFC}(\{s_1\}, \mathcal{S}_b) = A_{OIFT}(\{s_1\}, \mathcal{S}_b)$, (b) $A_{ORFC}(\{s_2\}, \mathcal{S}_b) = A_{OIFT}(\{s_2\}, \mathcal{S}_b)$, (c) $A_{ORFC}(\{s_1, s_2\}, \mathcal{S}_b)$ with two object seeds. (d) $A_{OIFT}(\{s_1, s_2\}, \mathcal{S}_b)$ with two object seeds.*

Proposition 1. *For any sets of seeds $\mathcal{S}_o$ and $\mathcal{S}_b$, we have that $A_{ORFC}(\mathcal{S}_o, \mathcal{S}_b) \subseteq A_{OIFT}(\mathcal{S}_o, \mathcal{S}_b)$.*

Proof. For a single internal seed s_i, by Equation 3.2, $A_{ORFC}(\{s_i\}, \mathcal{S}_b) \in \mathcal{X}_\infty^{\downarrow}(\{s_i\}, \mathcal{S}_b)$ and, based on Miranda and Mansilla [31,32], we also have $A_{OIFT}(\{s_i\}, \mathcal{S}_b) \in \mathcal{X}_\infty^{\downarrow}(\{s_i\}, \mathcal{S}_b)$. As $A_{ORFC}(\{s_i\}, \mathcal{S}_b)$ is the smallest element in $\mathcal{X}_\infty^{\downarrow}(\{s_i\}, \mathcal{S}_b)$ (Equation 3.2), we have that $A_{ORFC}(\{s_i\}, \mathcal{S}_b) \subseteq A_{OIFT}(\{s_i\}, \mathcal{S}_b)$. Therefore, in case of multiple internal seeds:

$$A_{ORFC}(\mathcal{S}_o, \mathcal{S}_b) = \bigcup_{s_i \in \mathcal{S}_o} A_{ORFC}(\{s_i\}, \mathcal{S}_b) \subseteq \bigcup_{s_i \in \mathcal{S}_o} A_{OIFT}(\{s_i\}, \mathcal{S}_b) \tag{4.1}$$

Note that, $\forall t \in V, f^{\sphericalangle}(\pi_{\mathcal{S}'_o \rightsquigarrow t}) \leq f^{\sphericalangle}(\pi_{\mathcal{S}_o \rightsquigarrow t})$ for $\mathcal{S}'_o \subseteq \mathcal{S}_o$. Hence, $A_{OIFT}(\mathcal{S}'_o, \mathcal{S}_b) \subseteq A_{OIFT}(\mathcal{S}_o, \mathcal{S}_b)$ and, consequently, $A_{OIFT}(\{s_i\}, \mathcal{S}_b) \subseteq A_{OIFT}(\mathcal{S}_o, \mathcal{S}_b), \forall s_i \in \mathcal{S}_o$. Then $\bigcup_{s_i \in \mathcal{S}_o} A_{OIFT}(\{s_i\}, \mathcal{S}_b) \subseteq A_{OIFT}(\mathcal{S}_o, \mathcal{S}_b)$. By joining it with Equation 4.1, we conclude that $A_{ORFC}(\mathcal{S}_o, \mathcal{S}_b) \subseteq A_{OIFT}(\mathcal{S}_o, \mathcal{S}_b)$. □

Next, we present an analysis of the core of the methods. Given that the delineated object by RFC corresponds to the core of IFT-*Watershed* [12, 22], we may ask if $A_{ORFC}(\mathcal{S}_o, \mathcal{S}_b) = \mathcal{N}_{OIFT}(\mathcal{S}_o, \mathcal{S}_b)$, but Figure 4.2 shows a counterexample. We also could think that ORFC and OIFT possess the same core, but Figure 4.3 shows another counterexample. Another question is to find whether the core of $A_{CoH(ORFC)}(\mathcal{S}_o, \mathcal{S}_b))$ (ORFC followed by a post-processing by *Closing of Holes* [52]) corresponds to the core of OIFT, but Figure 4.4 shows another counterexample. These results suggest the Proposition 2.

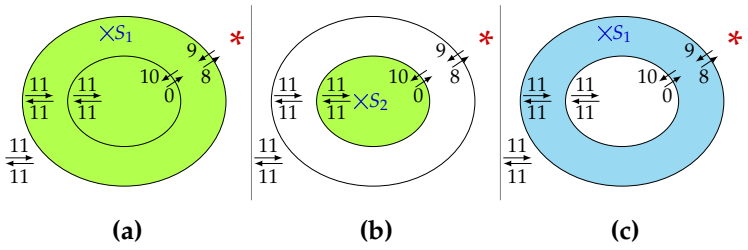


Figure 4.2: *Showing that $A_{ORFC}(\{s_1\}, \mathcal{S}_b) \neq \mathcal{N}_{OIFT}(\{s_1\}, \mathcal{S}_b)$, with $\mathcal{S}_o$ (×) in different positions and $\mathcal{S}_b$ (⋆) fixed. (a) $A_{ORFC}(\{s_1\}, \mathcal{S}_b) = A_{OIFT}(\{s_1\}, \mathcal{S}_b)$, (b) $A_{ORFC}(\{s_2\}, \mathcal{S}_b) = A_{OIFT}(\{s_2\}, \mathcal{S}_b)$ (c) $\mathcal{N}_{ORFC}(\{s_1\}, \mathcal{S}_b) = \mathcal{N}_{OIFT}(\{s_1\}, \mathcal{S}_b)$, but $A_{ORFC}(\{s_1\}, \mathcal{S}_b) \neq \mathcal{N}_{OIFT}(\{s_1\}, \mathcal{S}_b)$.*

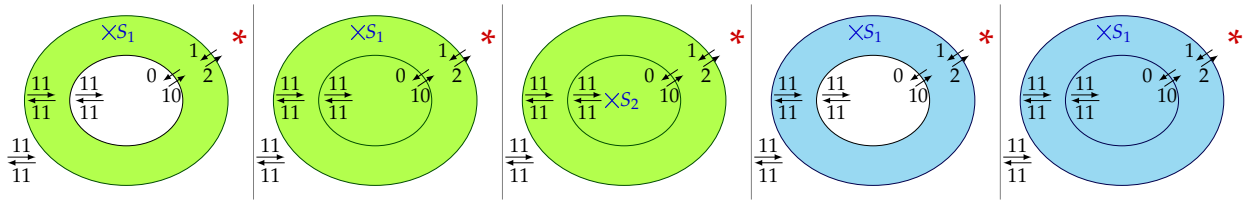


Figure 4.3: *Showing that $\mathcal{N}_{OIFT}(\{s_1\}, \mathcal{S}_b) \neq \mathcal{N}_{ORFC}(\{s_1\}, \mathcal{S}_b)$, with $\mathcal{S}_o$ (×) in different positions and $\mathcal{S}_b$ (⋆) fixed. (a) $A_{ORFC}(\{s_1\}, \mathcal{S}_b)$, where $C_{opt}(s_1) = 2$. (b) $A_{OIFT}(\{s_1\}, \mathcal{S}_b)$ (c) $A_{ORFC}(\{s_2\}, \mathcal{S}_b) = A_{OIFT}(\{s_2\}, \mathcal{S}_b)$, where $C_{opt}(s_2) = 2$. (d) $\mathcal{N}_{ORFC}(\{s_1\}, \mathcal{S}_b)$, (e) $\mathcal{N}_{OIFT}(\{s_1\}, \mathcal{S}_b)$. Note that $\mathcal{N}_{OIFT}(\{s_1\}, \mathcal{S}_b) \neq \mathcal{N}_{ORFC}(\{s_1\}, \mathcal{S}_b)$.*

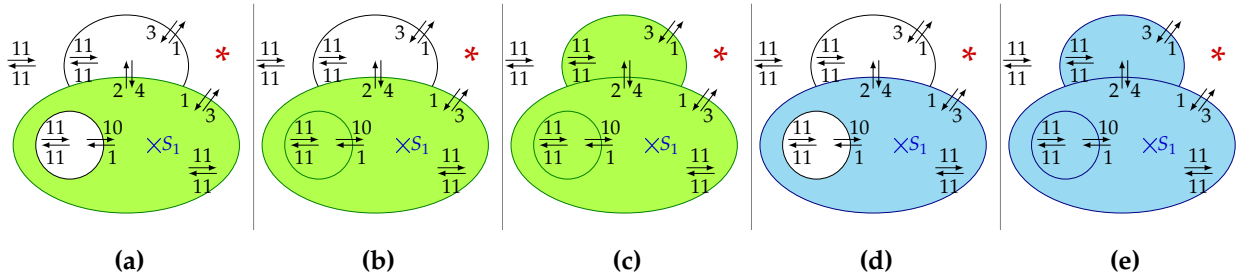


Figure 4.4: *Showing that $\mathcal{N}_{CoH(ORFC)}(\{s_1\}, \mathcal{S}_b) \neq \mathcal{N}_{OIFT}(\{s_1\}, \mathcal{S}_b)$, with $\mathcal{S}_o$ (×) in different positions and $\mathcal{S}_b$ (⋆) fixed. (a) $A_{ORFC}(\{s_1\}, \mathcal{S}_b)$, (b) $A_{CoH(ORFC)}(\{s_1\}, \mathcal{S}_b)$, (c) $A_{OIFT}(\{s_1\}, \mathcal{S}_b)$, (d) $\mathcal{N}_{ORFC}(\{s_1\}, \mathcal{S}_b)$, (e) $\mathcal{N}_{OIFT}(\{s_1\}, \mathcal{S}_b)$. As $\mathcal{N}_{CoH(ORFC)}(\{s_1\}, \mathcal{S}_b) \subseteq A_{CoH(ORFC)}(\{s_1\}, \mathcal{S}_b) \subset \mathcal{N}_{OIFT}(\{s_1\}, \mathcal{S}_b)$, therefore $\mathcal{N}_{CoH(ORFC)}(\{s_1\}, \mathcal{S}_b) \neq \mathcal{N}_{OIFT}(\{s_1\}, \mathcal{S}_b)$.*

Proposition 2. *For any seed s_i and seed set $\mathcal{S}_b$:*

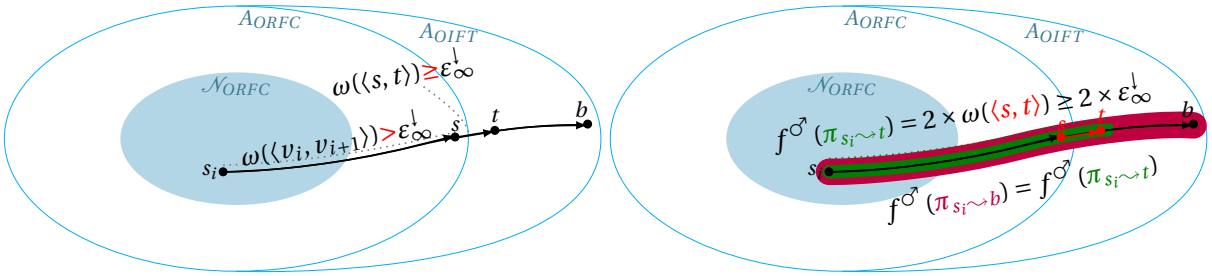
$$\mathcal{N}_{ORFC}(\{s_i\}, \mathcal{S}_b) \subseteq \mathcal{N}_{CoH(ORFC)}(\{s_i\}, \mathcal{S}_b) \subseteq \mathcal{N}_{OIFT}(\{s_i\}, \mathcal{S}_b) \tag{4.2}$$

Proof. The first part of Equation 4.2, $\mathcal{N}_{ORFC}(\{s_i\},\mathcal{S}_b) \subseteq \mathcal{N}_{CoH(ORFC)}(\{s_i\},\mathcal{S}_b)$, has a trivial proof, since any seed position in the core of ORFC, will produce the same output with closing of holes as well, by definition.

$\mathcal{N}_{ORFC}(\{s_i\},\mathcal{S}_b) \subseteq \mathcal{N}_{OIFT}(\{s_i\},\mathcal{S}_b)$ can be proved as follows: Let $\mathcal{C}_{ORFC} = \mathcal{C}(A_{ORFC}(\{s_i\},\mathcal{S}_b))$ and $\mathcal{C}_{OIFT} = \mathcal{C}(A_{OIFT}(\{s_i\},\mathcal{S}_b))$. We know that $\omega(\langle s,t\rangle) \leq \varepsilon^{\downarrow}_{\infty}$ for any arc $\langle s,t\rangle$ in $\mathcal{C}_{ORFC} \cup \mathcal{C}_{OIFT}$, by definition of $\varepsilon^{\downarrow}_{\infty}$. By Equation 3.2, $\forall s \in A_{ORFC}(\{s_i\},\mathcal{S}_b), \exists \pi_{s_i \rightsquigarrow s} \in \Pi_t$ where $\pi_{s_i \rightsquigarrow s} = \langle s_i = v_1,\ldots,v_n = s\rangle$ and $\omega(\langle v_i,v_{i+1}\rangle) > \varepsilon^{\downarrow}_{\infty}$, for $i = 1,\ldots,n-1$, otherwise s would not be part of $DCC_{G_>}(s_i)$ (Algorithm 3). Hence, for any arc $\langle s,t\rangle \in \mathcal{C}_{ORFC}$, $f^{\sigma}(\pi^P_{s_i \rightsquigarrow s}) > 2 \times \varepsilon^{\downarrow}_{\infty}$ and $f^{\sigma}(\pi^P_{s_i \rightsquigarrow s} \cdot \langle s,t\rangle) = 2 \times \omega(\langle s,t\rangle) \geq 2 \times \varepsilon^{\downarrow}_{\infty}$, where $\pi^P_{s_i \rightsquigarrow s}$ corresponds to the path calculated in the forest P by IFT with function f^{σ}.

Since $f^{\sigma}(\pi^P_{s_i \rightsquigarrow s} \cdot \langle s,t\rangle) = 2 \times \omega(\langle s,t\rangle)$ and $\langle s,t\rangle \in \mathcal{C}_{ORFC}$, consequently the connectivity values offered by f^{σ} to nodes outside the $A_{ORFC}(\{s_i\},\mathcal{S}_b)$ are defined exclusively based on arcs from $\mathcal{C}_{ORFC}$. Therefore, the result of $A_{OIFT}(\{s_i\},\mathcal{S}_b)$ is not affected by changes in the position of s_i within $\mathcal{N}_{ORFC}(\{s_i\},\mathcal{S}_b)$, since these changes leads to the same $\mathcal{C}_{ORFC}$. □

The part $\mathcal{N}_{CoH(ORFC)}(\{s_i\},\mathcal{S}_b) \subseteq \mathcal{N}_{OIFT}(\{s_i\},\mathcal{S}_b)$ can be proved, in an analog way, by using essentially the same arguments. Figure 4.5 illustrates the proof of Proposition 2.



(a) **(b)**

Figure 4.5: *Illustration of Proof of Proposition 2: (a)* $\omega(\langle v_i,v_{i+1}\rangle) > \varepsilon^{\downarrow}_{\infty}$ *for* $\pi^P_{s_i \rightsquigarrow s} = \langle s_i = v_1,\ldots,v_n = s\rangle$ *and* $\omega(\langle s,t\rangle) \geq \varepsilon^{\downarrow}_{\infty}$*, so (b)* $\langle s,t\rangle$ *defines not only* $f^{\sigma}(\pi_{s_i \rightsquigarrow t})$ *but also* $f^{\sigma}(\pi_{s_i \rightsquigarrow b})$ *for* $b \in A_{OIFT}$*. Changing* s_i *inside* $\mathcal{N}_{ORFC}$ *doesn't change* $\mathcal{C}_{ORFC}$*, which doesn't change* A_{OIFT}*, which implies* $\mathcal{N}_{ORFC} \subseteq \mathcal{N}_{OIFT}$*.*

4.1 ORFC Core

We present a formal definition and an efficient algorithm to compute the core $\mathcal{N}_{ORFC}(\{s_i\},\mathcal{S}_b)$ of a ORFC seed s_i.

If $s_1 \equiv s_2$, then $A_{ORFC}(\{s_1\},S_b) = A_{ORFC}(\{s_2\},S_b)$ and, consequently, by Lemma 1 from Bejar and Miranda [33], $C_{opt}(s_1) = C_{opt}(s_2) = \varepsilon^{\downarrow}_{\infty}$, for the connectivity function $f^{\leftarrow}_{\min}$. Therefore, nodes in the same core must have the same value in the map C_{opt}. This condition is necessary, but not sufficient. Note that, since $C_{opt}(s_1) = C_{opt}(s_2)$, in Step 2 of Algorithm 3, the digraph $G_>$ will be the same for equivalent seeds. As $A_{ORFC}(\{s_1\},\mathcal{S}_b) = A_{ORFC}(\{s_2\},\mathcal{S}_b)$, $\{\pi_{s_1 \rightsquigarrow s_2}, \pi_{s_2 \rightsquigarrow s_1}\} \subseteq \Pi(G_>)$ (i.e., $s_1 \in DCC_{G_>}(s_2)$ and $s_2 \in DCC_{G_>}(s_1)$). Therefore, $\mathcal{N}_{ORFC}(\{s\},\mathcal{S}_b)$ forms a SCC in $G_>$, considering only arcs $\langle s,t\rangle$, such that $C_{opt}(s), C_{opt}(t) = \varepsilon^{\downarrow}_{\infty}$, which can be computed in linear time by Tarjan's algorithm [42] (Algorithm 1). Algorithm 4 calculates $\mathcal{N}_{ORFC}(\{s\},\mathcal{S}_b)$. In the case of multiple seeds, we consider $\mathcal{N}_{ORFC}(\mathcal{S}_o,\mathcal{S}_b) = \bigcup_{s_i \in \mathcal{S}_o} \mathcal{N}_{ORFC}(\{s_i\},\mathcal{S}_b)$. To find $\mathcal{N}_{ORFC}(\mathcal{S}_o,\mathcal{S}_b)$, instead of computing separately each individual core and applying a union procedure, we can find all cores at once. Note that in Step 2 of Algorithm 4, as $C(s) = C(s_i)$, we can change $\omega(\langle s,t\rangle) > C_{opt}(s_i) \wedge C_{opt}(s) = C_{opt}(t) = C_{opt}(s_i)$ by $w(\langle s,t\rangle) > C_{opt}(s) \wedge C_{opt}(s) = C_{opt}(t)$, apply Tarjan's algorithm and label the SCCs of all internal seeds as objects, so that the complexity

does not depend on the number of seeds (Algorithm 5). Figure 4.6 shows one example of $\mathcal{N}_{ORFC}(\{s_i\}, \mathcal{S}_b)$ computed by Algorithm 4.

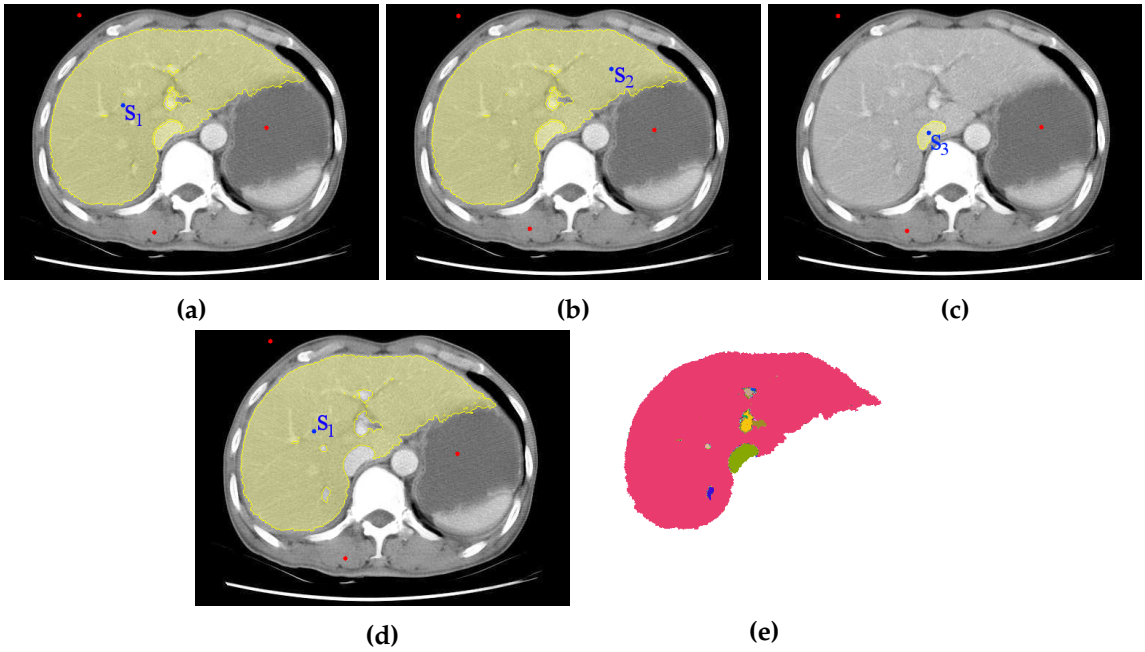


Figure 4.6: *A slice image from a CT thoracic study of the liver. (a-c) ORFC segmentation results for different internal seeds (s_1,s_2, and s_3). (d) $\mathcal{N}_{ORFC}(\{s_1\}, \mathcal{S}_b)$: The core of s_1 by ORFC computed by Algorithm 4. From the above results, we can conclude that $s_1 \equiv s_2$, but s_1 is not equivalent to s_3. Note that by comparing ORFC from (a) and the core of seed s_1 from (d), it is easy to get the inferior vena cava, as the largest residual component. (e) Multiple ORFC Cores computed at once by Algorithm 5.*

Algorithm 4: Computing $\mathcal{N}_{ORFC}(\{s_i\}, \mathcal{S}_b)$

1 Get connectivity map C_{opt} with $f_{\min}^{\leftarrow}$ by IFT;
2 Create $G_> = \langle V, E', \omega \rangle$ from $G = \langle V, E, \omega \rangle$ where $E' = \{\langle s,t \rangle \in E : \omega(\langle s,t \rangle) > C_{opt}(s_i) \wedge C_{opt}(s) = C_{opt}(t) = C_{opt}(s_i)\}$;
3 Return $SCC_{G_>}(s_i)$;

Algorithm 5: Computing $\mathcal{N}_{ORFC}(\mathcal{S}_o, \mathcal{S}_b)$

1 Get connectivity map C_{opt} with $f_{\min}^{\leftarrow}$ by IFT;
2 Create $G_> = \langle V, E', \omega \rangle$ from $G = \langle V, E, \omega \rangle$ where $E' = \{\langle s,t \rangle \in E : \omega(\langle s,t \rangle) > C_{opt}(s) \wedge C_{opt}(s) = C_{opt}(t)\}$;
3 Apply Tarjan's algorithm in $G_>$;
4 Return only SCCs which contains internal seeds;

4.2 OIFT Core

From Proposition 2, we know that, for any $s_i \in \mathcal{S}_o$, $\mathcal{N}_{ORFC}(\{s_i\}, \mathcal{S}_b) \subseteq \mathcal{N}_{OIFT}(\{s_i\}, \mathcal{S}_b)$. If a node s_i is equivalent to a node s_2 for OIFT (i.e., $s_1 \overset{\text{oift}}{\equiv} s_2$), and they belong to different ORFC cores (i.e., $\mathcal{N}_{ORFC}(\{s_1\}, \mathcal{S}_b) \neq \mathcal{N}_{ORFC}(\{s_2\}, \mathcal{S}_b)$), then by transitivity we have that $c \overset{\text{oift}}{\equiv} d$ for any $c \in \mathcal{N}_{ORFC}(\{s_1\}, \mathcal{S}_b)$ and $d \in \mathcal{N}_{ORFC}(\{s_2\}, \mathcal{S}_b)$. This observation allow us to drastically reduce the complexity of the OIFT core computation problem, allowing us to work in a *Region Adjacency Graph* (RAG), composed by the ORFC cores that can be fast computed, rather than working at the node level.

Since $\mathcal{N}_{OIFT}(\mathcal{S}_o, \mathcal{S}_b) \subseteq A_{OIFT}(\mathcal{S}_o, \mathcal{S}_b)$, we first compute $A_{OIFT}(\mathcal{S}_o, \mathcal{S}_b)$, then we compute all the ORFC cores inside OIFT segmentation $A_{OIFT}(\mathcal{S}_o, \mathcal{S}_b)$. Figure 4.7 illustrates one example, showing all the ORFC cores inside the object for a given image graph (Figure 4.7i). Figure 4.8a shows the resulting RAG, with a node for each ORFC core and one external node x for the

background. The arc weights of the RAG are selected as the highest arc values interconnecting their regions.

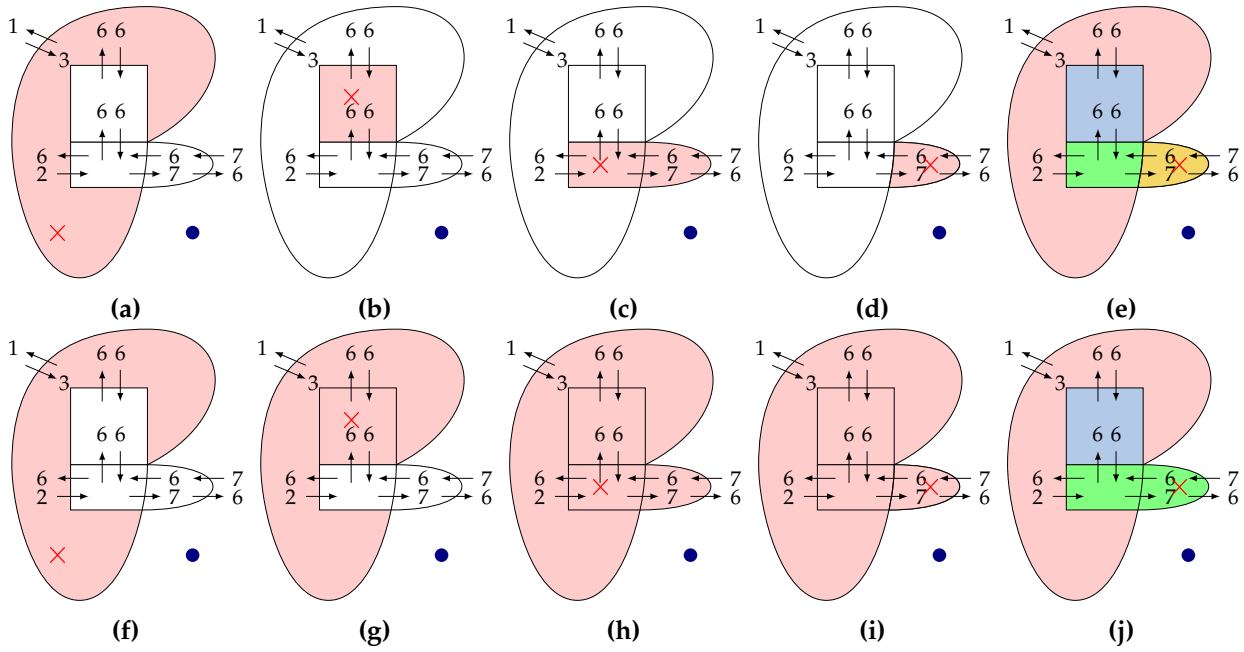


Figure 4.7: *(a-d) Results of ORFC, where $\omega(\langle s,t\rangle) = 10$ for non-contour edges, with a fixed external seed • and an internal seed × in different places, (e-h) OIFT results for different internal seeds, (i) ORFC cores and (j) OIFT cores.*

The proposed algorithm to compute the OIFT cores uses a disjoint-set data structure. Initially, each RAG node is its own representative. For each pair $\langle c,d\rangle$, $c \neq x$ and $d \neq x$, of neighboring nodes in the RAG an equivalence test is performed and if the test is satisfied they are joined (union operation). The value $C_c(d) = f(\pi_d^*)$ of an optimum path π_d^* by the connectivity function $f_{\min}^{\leftarrow}$ (Eq. 3.3) is computed in the induced subgraph $G[V \setminus \{c\}]$ from x to d (Figure 4.8b). Similarly we also compute $C_d(c) = f(\pi_c^*)$ as the value of an optimum path π_c^* for $f = f_{\min}^{\leftarrow}$ in the induced subgraph $G[V \setminus \{d\}]$ from x to c (Figure 4.8c). If $\omega(\langle c,d\rangle) > C_c(d)$ and $\omega(\langle d,c\rangle) > C_d(c)$ we can conclude that $c \overset{\text{oift}}{\equiv} d$ and we perform their union operation (Figure 4.8d). Dual OIFT works with $>$ instead of $<$.

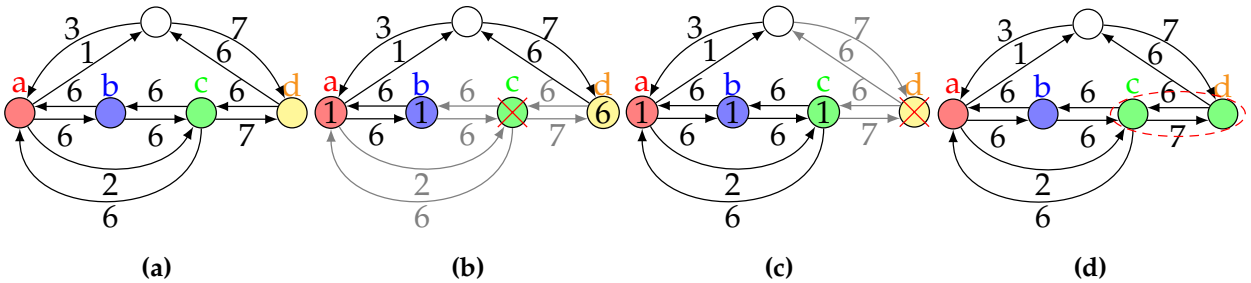


Figure 4.8: *(a)* Region Adjacency Graph *(RAG), composed by the ORFC cores from Figure 4.7. (b-c) The equivalence test: (b) $\omega(\langle c,d\rangle) = 7 > C_c(d) = 6$, and (c) $\omega(\langle d,c\rangle) = 6 > C_d(c) = 1$. (d) The union operation.*

In order to understand the equivalence test performed in RAG, we need to know the following property that distinguishes OIFT from ORFC. From Figure 3.4, we can note that in the case of multiple solutions with the same energy, the OIFT result gives preference to select boundary pieces with lower energy values. For example, between the border segments with outgoing arcs with values 3 and 2, from Figures 3.4c and d, OIFT selects the one with the lowest value in Figure 3.4d. This result can be verified theoretically by a proof similar to Theorem 2 (Piecewise optimum property) in [29]. In the equivalence test, $\omega(\langle c,d\rangle)$ and $C_c(d)$ essentially repre-

sent the energies of two boundary pieces. Since OIFT gives preference to lower energy values, $\omega(\langle c,d\rangle) > C_c(d)$ implies that a OIFT segmentation from a seed in c would conquer d, and $\omega(\langle d,c\rangle) > C_d(c)$ implies that d would conquer c leading to equivalent seeds. Figure 4.7j shows the resulting OIFT cores at the pixel level derived from the RAG in Figure 4.8d.

Since we have to evaluate the equivalence test, and consequently $C_c(d)$, for all arcs $\langle c,d\rangle$ in the RAG, the final complexity of the algorithm becomes $O(|V|^2 + |E| \cdot |V|)$, where $|V|$ and $|E|$ are the number of nodes and arcs in the RAG. Note that to compute the maps C_c for all $c \in V$ requires $|V|$ IFT's executions and each IFT takes $O(|V| + |E|)$. In practice, the algorithm is fast, because the RAG has a small number of nodes compared to the image graph.

4.3 Robustness Coefficient

We define a measure to evaluate the robustness of the methods in relation to the seed positioning. For a given segmentation algorithm $A(\mathcal{S}_o, \mathcal{S}_b)$ with cores given by $\mathcal{N}(\mathcal{S}_o, \mathcal{S}_b)$, the *Robustness Coefficient* (RC) is defined as:

$$RC = \frac{|\mathcal{N}(\mathcal{S}_o, \mathcal{S}_b)|}{|A(\mathcal{S}_o, \mathcal{S}_b)|} \tag{4.3}$$

RC provides an analytic solution to measure the reproducibility of experiments. The higher the RC value, the lower is the sensitivity of the method in relation to inter- and intra-user variability in image segmentation. Note that a high RC value does not imply that the method has a high accuracy, the RC measure only evaluates how easy it is to reproduce the same segmentation, regardless of its accuracy. In this sense, it is a complementary measure to traditional accuracy measures.

The next two chapters applies all these robustness analysis for solving the main studied problems with experimental results. Chapter 6 tackles the segmentation editing problem and Chapter 5 develops a hybrid approach to solve Graph Cut problems.

Chapter 5

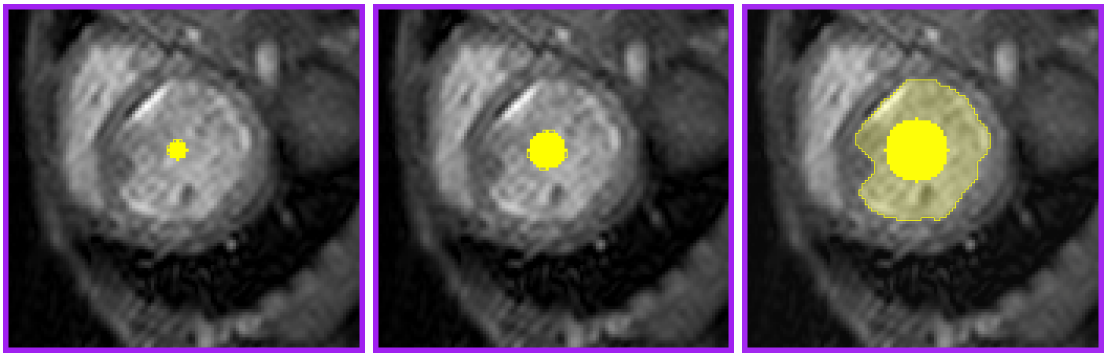
Hybrid Method $ORFC_{Core} + GC$

The study of cores also can help analyze and improving robustness of segmentation methods. This chapter describes a hybrid method ORFC+GC [33], for reducing *Graph Cut* (Section 5.1) problems by using ORFC cores.

5.1 Graph Cut (GC)

Different from solving OPSFP, *Graph Cut* (GC) [20] finds a partition by solving *minimum cut / maximum flow* [64, 65] problem. Let graph $G = \langle V, E, w \rangle$ and $N = \langle G, c, s, t \rangle$ be a *flow network* where $c : E \to \mathbb{R}^+$ assigns for each arc a *capacity*, the maximum *flow* $fl : E \to \mathbb{R}$ allowed to pass through it, $s \in V$ the source and $t \in V$ the sink, which respects *flow conservation* $\sum_{\langle i,v \rangle \in E} fl(\langle i,v \rangle) = \sum_{\langle v,o \rangle \in E} fl(\langle v,o \rangle), \forall v \in V \setminus \{s,t\}$ and *antisymmetry* $fl(\langle u,v \rangle) = -fl(\langle v,u \rangle), \forall \langle u,v \rangle \in E$. The goal of maximum flow problem is maximize network flow $|fl| = \sum_{\langle s,v \rangle \in E} fl(\langle s,t \rangle) = \sum_{\langle v,t \rangle \in E} fl(\langle v,t \rangle)$, equivalent to minimum cut problem, that is, to choose a partition $\{S,T\}$ of V ($s \in S$ and $t \in T$) which minimizes capacity $\sum_{\langle u,v \rangle \in \mathcal{C}} c(\langle u,v \rangle)$ of cut $\mathcal{C} = \{\langle u,v \rangle \in E : u \in S, v \in T\}$.

In case of graph $G = \langle V, E, w \rangle$ originated from images, an interactive segmentation with seeds $\mathcal{S} = \mathcal{S}_o \cup \mathcal{S}_b \subseteq V$ uses a modified network $N = \langle G', c, s, t \rangle$, where $G' = \langle V', E', w' \rangle$, with $V' = V \cup \{s,t\}$, $E' = E \cup E_o \cup E_b, E_o = \{\langle s,i \rangle : i \in \mathcal{S}_o\}, E_b = \{\langle i,t \rangle : i \in \mathcal{S}_b\}, \omega'(e) = \omega(e), \forall e \in E$, $\omega'(e) = \infty, \forall e \in E_o \cup E_b$, and $c(\langle u,v \rangle) = \omega'(\langle u,v \rangle)$, partitioning V onto $A_{GC}(\mathcal{S}_o, \mathcal{S}_b) = \mathcal{O}$ and $V \setminus \mathcal{O}$ and making *Graph Cut* a ε_1-minimizer. One problem with ε_1-minimizer is the preference for smaller objects (with low perimeter), which degenerates to $\mathcal{S}_o$, a problem denoted *shrinking bias* (Figure 5.1).



(a) **(b)** **(c)**

Figure 5.1: *Example of Shrinking Bias of Graph Cut, where yellow opaque pixels represent internal seeds and purple for external ones. (a-b) Segmentations are equivalent to proper seeds, (c) the user needs to add more seeds (and effort) to get desired region.*

5.2 $ORFC_{Core} + GC$

ORFC, as a ε_∞-minimizer, does not have shrinking bias, it has complexity $O(n)$ and it is more robust (bigger cores) than GC, with complexity $O(n^{2.5})$ [66]. GC helps ORFC by being more robust against weaker borders. Algorithm 6 expands $\mathcal{S}_o$ and $\mathcal{S}_b$ through $\mathcal{N}_{ORFC}$ (reducing shrinking bias possibility and raising robustness) and sends expanded sets $\mathcal{S}'_o$ and $\mathcal{S}'_b$ to GC for labeling remaining nodes $V \setminus \mathcal{S}'_o \cup \mathcal{S}'_b$.

Algorithm 6: Computing $A_{ORFC_{Core}+GC}(\mathcal{S}_o, \mathcal{S}_b)$

1 $\mathcal{S}'_o \leftarrow \mathcal{N}_{ORFC}(\mathcal{S}_o, \mathcal{S}_b)$ in G;
2 $\mathcal{S}'_b \leftarrow \mathcal{N}_{ORFC}(\mathcal{S}_b, \mathcal{S}_o)$ in G^T;
3 Return $\leftarrow A_{GC}(\mathcal{S}'_o, \mathcal{S}'_b)$ in G;

5.2.1 Experimental Results

In the experiments, we used 40 slice images from realMR images of the foot, to perform the segmentation of the bones calcaneus and talus, for all the methods ($IRFC$ [23], RFC [24], Power Watershed (PW) [67], $OIFT$ [31], $RFC + GC$ [68], OGC [20] - the graph cut with boundary polarity, $ORFC$ [33], $ORFC + GC$ [33], and the proposed hybrid method $ORFC_{Core} + GC$), for different seed sets automatically obtained by eroding and dilating the ground truth at different radius values, totaling a total of 1200 executions for each method. By varying the radius value, we can repeat the segmentation for different seed sets and trace accuracy curves using the Dice coefficient of similarity However, in order to generate a more challenging situation, we considered a larger radius of dilation for the external seeds (twice the value of the inner radius), resulting in an asymmetrical arrangement of seeds.

Several different procedures can be adopted for $\delta(a, b)$ [44, 69]. For example, Figures 4.6 and 5.2 show some results for user-selected markers using the image-based weight assignment from [43]. For the sake of simplicity, in the quantitative experiments, we adopted the weight assignment $\delta(a, b) = K - |G(a) + G(b)|$, where $G(a)$ denotes the gradient magnitude of the Sobel operator and the dual IFT. In Equation 2.2, α could be in the range of $[-1, 1]$, We used $\alpha = -0.5$, in all experiments involving $OIFT$, OGC, $ORFC$, $ORFC + GC$, and $ORFC_{Core} + GC$; since the foot bones present transitions from dark to bright pixels; and $\alpha = 0.0$ in the case of undirected approaches.

The PW code comes from a software library in C developed by Camille Couprie, which is available at SourceForge[1]. The OGC code comes from a software library in C++[2] developed by Yuri Boykov and Vladimir Kolmogorov. It implements the max-flow algorithm [66]. In the case of $ORFC$ and RFC, we considered a post-processing by *Closing of Holes* [52] to improve their results.

Figure 5.3 shows the experimental curves. OGC has a decreasing accuracy for higher radius values due to the shrinking problem, while the proposed hybrid method $ORFC_{Core} + GC$ can conserve a high accuracy, with better results in general than $ORFC + GC$ [33].

[1] http://sourceforge.net/projects/powerwatershed/
[2] http://pub.ist.ac.at/~vnk/software/maxflow-v3.04.src.zip

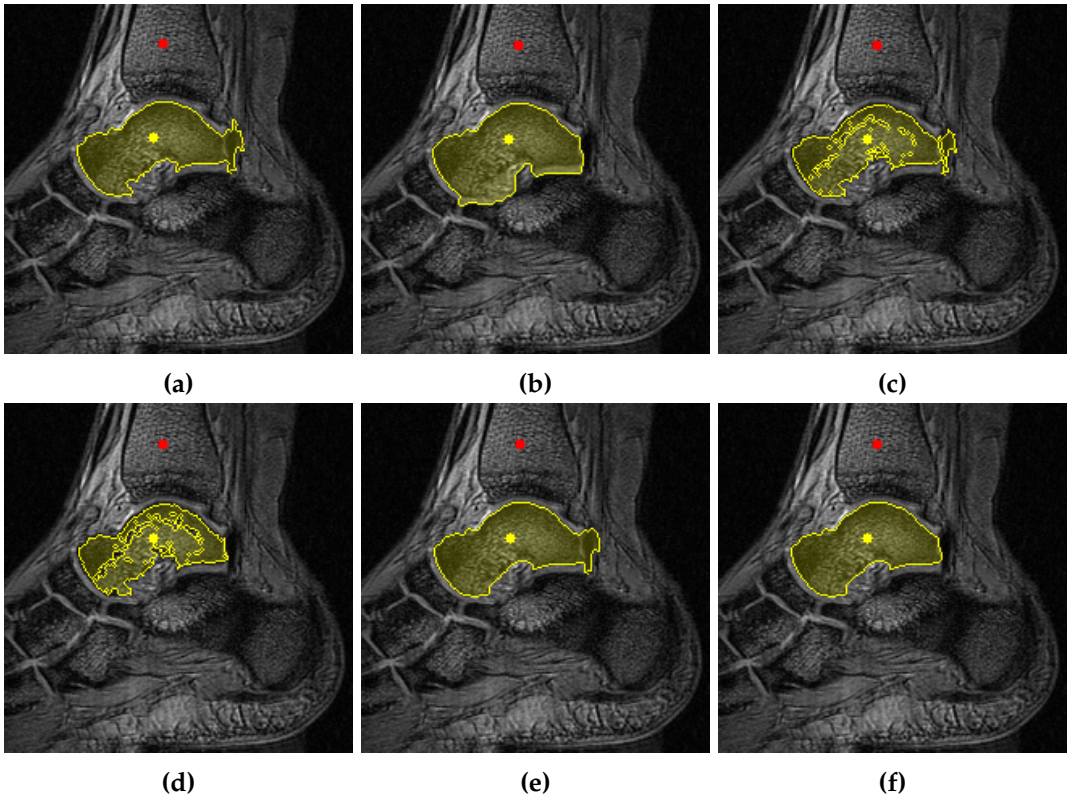


Figure 5.2: *AMRimage of the foot. Segmentation results of the talus bone for the given user-selected markers by: (a) OIFT, (b) RFC + GC, (c) ORFC, (d) Core of ORFC, (e) ORFC + GC, (f) $ORFC_{Core}$ + GC.*

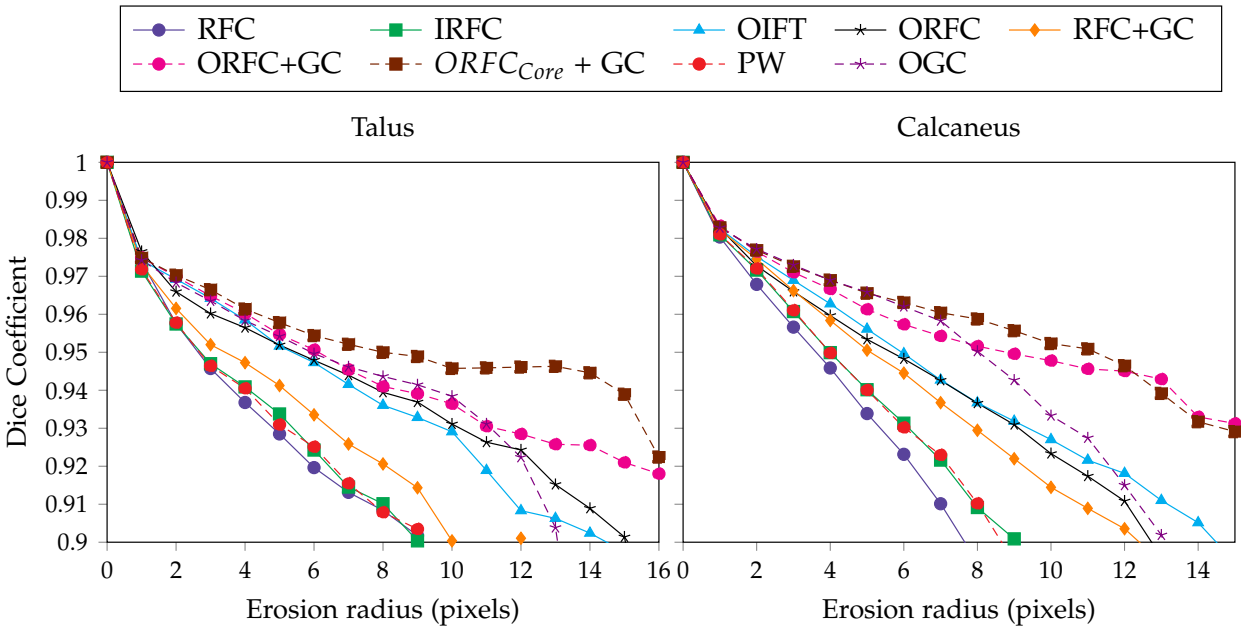


Figure 5.3: *The mean accuracy curves (Dice coefficient of similarity), using non-equally eroded-dilated seeds, for segmenting talus and calcaneus.*

Chapter 6

Interactive Segmentation Repair

Given a graph G (generated from an image, biological or social network, etc) and its binary presegmentation $\mathcal{O}$ (generated by any interactive, manual or automatic process), the main problem of this study refers to estimating a set of seeds $\mathcal{S}$ which can be used for continuing or fixing the result through future interactive sessions. This work solves *Reverse Interactive Segmentation Problem* (RISP), formulated in Problem 1 and illustrated in Figure 1.3, allowing the user to run IFT for repairs, which should generate $\mathcal{O}$ (or an approximated version) for G and $\mathcal{S}$.

Problem 1. Reverse Interactive Segmentation Problem *(RISP)*

- *Given: graph G, binary segmentation $\mathcal{O}$ and algorithm $A(\mathcal{S}_o, \mathcal{S}_b)$*
- *Return: set of seeds $\mathcal{S} = \mathcal{S}_o \cup \mathcal{S}_b$ so that $A(\mathcal{S}_o, \mathcal{S}_b) \sim \mathcal{O}$*

Two approaches are developed for solving RISP. IFT-SLIC iteratively moves an initial set $\mathcal{S}$ which generates supervoxels and analytical methods produce a minimal set $\mathcal{S}$ by equivalence classes (cores) and redundancy analysis.

6.1 IFT-SLIC

Although IFT-SLIC was originally conceived for graph partitioning, the natural returning of seeds turns the method a solution for RISP. IFT-SLIC automatically estimates $\mathcal{S}_o$ and $\mathcal{S}_b$, which is respectively the seeds inside and outside $\mathcal{O}$. The size of $\mathcal{S}$ is defined by the user before algorithm execution. An advantage of IFT-SLIC is the regular spacing of seeds, compactness and boundary adherence. The running time can be improved by computing just the bounding box around $\mathcal{O}$, specially if $\mathcal{O}$ is much smaller than V. In this case, we can keep the size of $\mathcal{S}$ proportional to the size of the bounding box, to have the density constant.

The union of all supervoxels from seeds in $\mathcal{S}_o$, gives us an initial approximation of the presegmentation, denoted as the *initial supervoxel segmentation*, which does not perfectly resemble the presegmentation (Figure 6.1d). To further boost the results, we improve the final supervoxel segmentation by changing the connectivity function to f'_D (Figure 6.1e) as follows:

$$\begin{aligned} f'_D(\pi_t = \langle t \rangle) &= f_D(\pi_t = \langle t \rangle) \\ f'_D(\pi_{r \rightsquigarrow s} \cdot \langle s,t \rangle) &= f'_D(\pi_{r \rightsquigarrow s}) + \underbrace{d_{euc}(s,t)}_{\text{Compactness}} + \underbrace{(\|I(t) - I(r)\| \cdot \delta \cdot \gamma^{B(r,t)} + \gamma \cdot B(r,t))^{\beta}}_{\text{Boundary Adherence}} \end{aligned} \tag{6.1}$$

where $B(r,t) = |B(r) - B(t)|$, that is, $B(r,t)$ captures the transitions in the binary mask B of the presegmentation, and γ plays the same role as the *liberal* and *conservative* forces used in [13]. For higher values of γ, the final supervoxel segmentation better resembles the presegmentation, conserving its fine details. Thus, higher values of γ allow us to reduce the number of supervoxels k, giving more freedom to the user to perform corrections. So we empirically used $k = vol\,/\,(200 \cdot \gamma)$, where vol is the number of object voxels in the presegmentation.

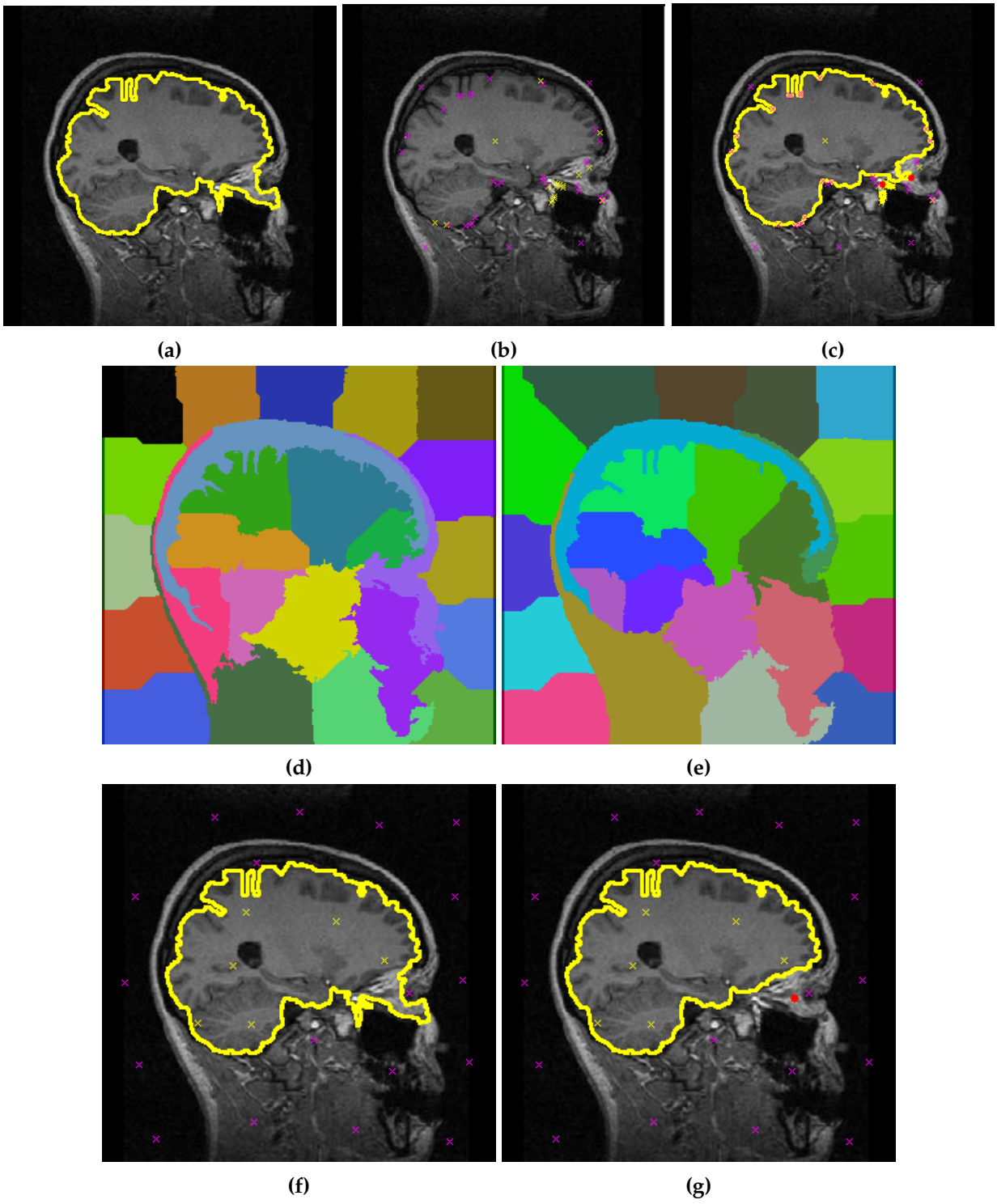


Figure 6.1: *IFT-SLIC for segmentation editing: (a) The given presegmentation. (b) Seed set computed by ISBI2011 [13] has many non-uniformly distributed seeds, and (c) its attempt to fix the segmentation by placing new background markers (red dots) fails. Proposed editing method: (d) Supervoxels by IFT-SLIC to find the seed set. (e) Supervoxels better conforming to the presegmentation are obtained by changing the cost function to f'_D. (f) The union of supervoxels from seeds contained in the presegmentation gives us a starting point to perform corrections. (g) A corrected result is obtained by adding a new background seed (red dot) and running DIFT.*

The final supervoxel segmentation can then be used as a starting point, so that the user can insert and/or remove seeds from $\mathcal{S}_0$ and $\mathcal{S}_1$ in order to correct the segmentation in a differential way, by using *Differential Image Foresting Transform* (DIFT) [30] with function f'_D (Figures 6.1f-g). Therefore, the corrections take sublinear time.

6.1.1 Experimental Results

In this section, we conducted experiments to measure the user involvement in the editing process of the wrong parts of the presegmentation in real 3 Tesla MRI-T1 images of the brain of size $240 \times 240 \times 180$ voxels with severe inhomogeneity problems. We also quantified the number of estimated seeds, where lower values indicate more flexibility for posterior user corrections. We compared our proposed method with the best solution so far by IFT, denoted as ISBI2011 [13]. In all cases, the corrective actions were conducted by a robot user [41], in order to get impartial results, with a spherical brush size of 5 voxels, using an Intel core i3 laptop with 4GB memory.

Table 6.1 shows the results of the first experiment (data set D1) to correct the wrong parts of automatic segmentation of the cerebral hemispheres, where the errors are related mainly to the bad positioning of the fuzzy model [29] (Figures 6.2a-b). The mean execution time to obtain the initial seeds by the proposed method was 24.0s and 13.5s for ISBI2011 [13]. The mean Dice value for the initial supervoxel segmentation using the seeds by IFT-SLIC increased from 89.75% to 99.96% when changing the path cost-function to f'_D for $\gamma = 3$, and from 88.64% to 99.98% for $\gamma = 4$. We noted that lower values of γ ($\gamma < 3$) can lead to a loss of presegmentation details. The proposed method reduced the number of markers required for corrective actions in 68.2% and reduced the total number of initial seeds in 4.3% for $\gamma = 3$. For $\gamma = 4$, we had a reduction of 60.8% for corrective actions and 29.2% for the number of initial seeds.

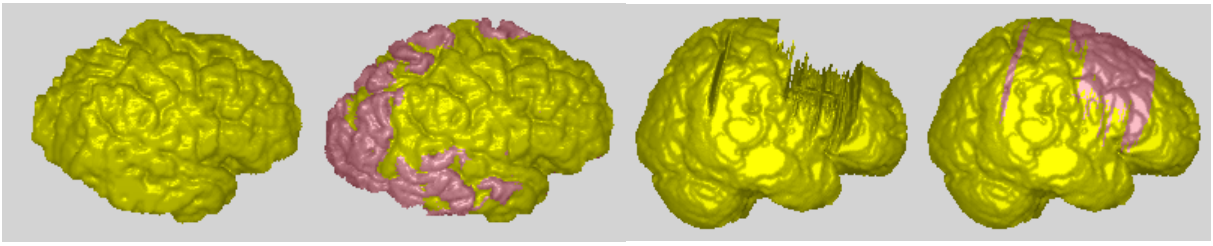
	Proposed ($\gamma = 3$)		Proposed ($\gamma = 4$)		ISBI2011	
image#	nm,	ns (‰),	nm,	ns (‰),	nm,	ns (‰),
01	5,	0.0729	7,	0.0463	46,	0.0657
02	10,	0.0608	13,	0.0463	35,	0.0766
03	12,	0.0729	12,	0.0502	42,	0.0811
04	15,	0.0602	18,	0.0463	33,	0.0949
05	8,	0.0781	11,	0.0648	23,	0.0443
06	6,	0.0677	10,	0.0463	15,	0.0683
07	8,	0.0677	10,	0.0463	26,	0.0750
08	15,	0.0729	16,	0.0501	20,	0.0470
09	6,	0.0501	8,	0.0463	20,	0.0672
10	9,	0.0502	11,	0.0405	36,	0.0631
Mean	9.4,	0.0653	11.6,	0.0483	29.6,	0.0683

Table 6.1: *Data set D1: Number of markers (*nm*) required for corrective actions and number of computed initial seeds (*ns*) per voxels in parts per thousand.*

	Proposed ($\gamma = 3$)		Proposed ($\gamma = 4$)		ISBI2011	
image#	nm,	ns (‰),	nm,	ns (‰),	nm,	ns (‰),
01	20,	0.1633	26,	0.1252	33,	0.8230
02	20,	0.1633	22,	0.1379	28,	1.2129
03	23,	0.1516	21,	0.1253	32,	0.9770
04	19,	0.1908	18,	0.1484	37,	0.6807
05	23,	0.1909	22,	0.1485	67,	1.6071
06	19,	0.1379	18,	0.1253	34,	1.0022
07	17,	0.1516	19,	0.1273	31,	0.3774
08	17,	0.1633	23,	0.1253	24,	0.4172
09	21,	0.1633	21,	0.1157	42,	0.4365
10	18,	0.1633	25,	0.0936	30,	0.4303
Mean	19.7,	0.1639	21.5,	0.1272	35.8,	0.7965

Table 6.2: *Data set D2: Number of markers (*nm*) required for corrective actions and number of computed initial seeds (*ns*) per voxels in parts per thousand.*

On the second experiment (data set D2 in Table 6.2), we considered a more challenging scenario. We conducted experiments to fix the segmentation of the cortical surface of the brain, where several pronounced errors were intentionally introduced by manual editing along the 3D surface (Figures 6.2c-d). The mean Dice value for the initial supervoxel segmentation using the seeds by IFT-SLIC increased from 93.08% to 99.95% when changing the path cost-function to f'_D for $\gamma = 3$, and from 92.48% to 99.95% for $\gamma = 4$. The proposed method reduced the number of markers required for corrective actions in 45% (39.9%) and reduced the total number of initial seeds in 79.4% (84%) for $\gamma = 3$ ($\gamma = 4$). Figure 6.9 shows an implementation of IFT-SLIC segmentation repair on Brain Image Analyser (BIA) software.

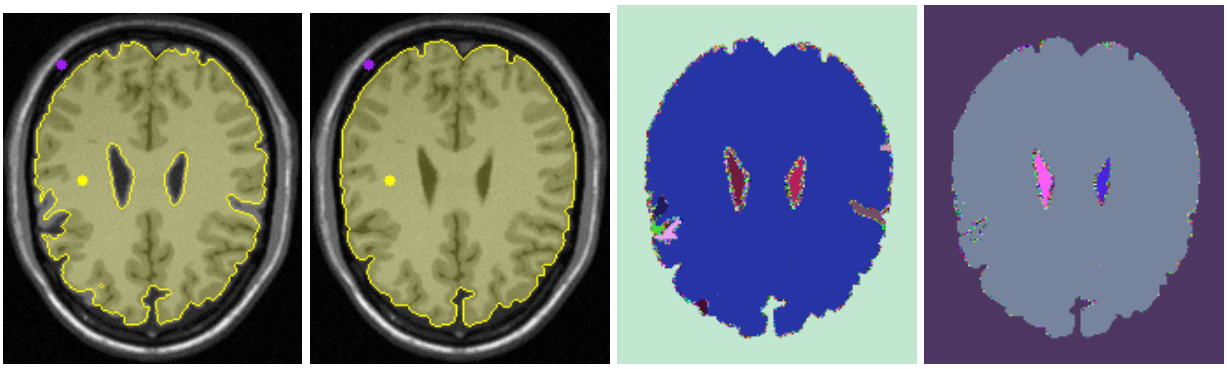


(a) **(b)** **(c)** **(d)**

Figure 6.2: *3D renditions of presegmentations with errors (a and c) and respective ground truths (b and d), with their main differences highlighted in another color. Examples (each data set has 10 different images) from: (a-b) Data set D1. (c-d) Data set D2 with severe errors.*

6.2 Segmentation editing by seed robustness

Figures 6.3, 6.4, 6.5, 6.6 and 6.7 show examples of the incremental computation of the cores by OIFT, from the ORFC cores, for a variety of real images. We will be using Robustness Coefficient measure from Equation 4.3 to compare the cores.



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

Figure 6.3: *A brain image from the BrainWeb - Simulated Brain Database. (a) ORFC segmentation with RC = 99.95%. (b) OIFT segmentation with RC = 96.23%. (c) ORFC cores inside OIFT mask. (d) OIFT cores.*

In the experiments, we used 40 slice images from real MR images of the foot, to perform the segmentation of the bones talus and calcaneus, and 40 slice images from CT cervical spine studies of 10 subjects to segment the spinal-vertebra. We used different seed sets automatically obtained by eroding and dilating the ground truth at different radius values. By varying the radius value, we can repeat the segmentation for different seed sets and trace accuracy curves, using the Dice coefficient of similarity, and curves of the robustness coefficient. However, in order to generate a more challenging situation, we considered a larger radius of dilation for the external seeds (twice the value of the inner radius), resulting in an asymmetrical arrangement of seeds.

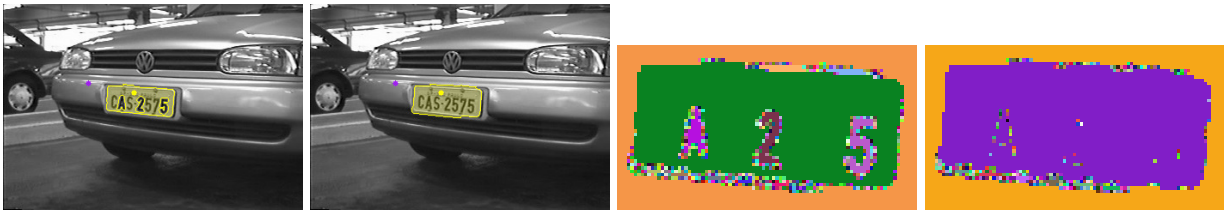


Figure 6.4: *Image of a license plate. (a) ORFC segmentation with RC = 97.89%. (b) OIFT segmentation with RC = 89.06%. (c) ORFC cores inside OIFT mask. (d) OIFT cores.*

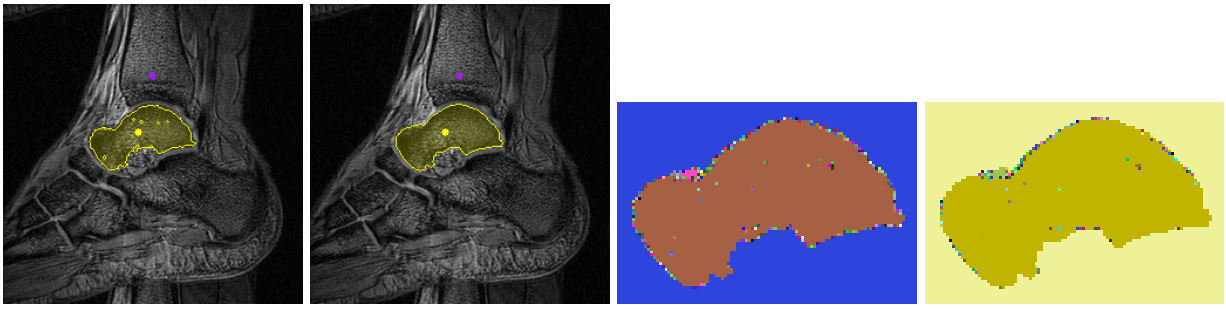


Figure 6.5: *MR image of a talus bone with good boundary contrast. (a) ORFC segmentation with RC = 98.60%. (b) OIFT segmentation with RC = 96.01%. (c) ORFC cores inside OIFT mask. (d) OIFT cores.*

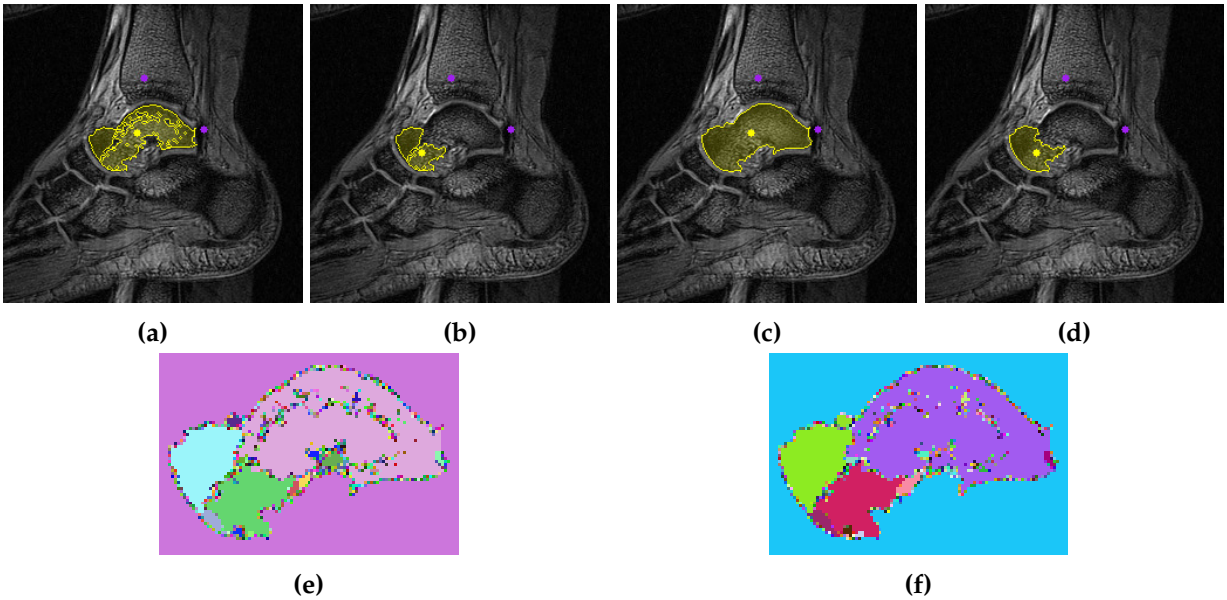


Figure 6.6: *MR image of a talus bone with poor boundary contrast. (a) ORFC segmentation with RC = 58.87%. (b) Effect of placing the seed outside its core. (c) OIFT segmentation with RC = 52.04%. (d) Effect of placing the seed outside its core. (e) ORFC cores inside the OIFT mask. (f) OIFT cores.*

In order to show the robustness coefficient, we considered in the evaluation only methods with known procedure to compute their cores: $IRFC$ [23], RFC [24], $OIFT$ [31], $ORFC$ [33], or at least with a good lower bound estimation of their cores: $RFC + GC$ [68], $ORFC + GC$ [33], and $ORFC_{Core} + GC$ [36]. For $RFC + GC$ we considered $RC = \frac{|\mathcal{N}_{RFC}(\mathcal{S}_o,\mathcal{S}_b)|}{|A_{RFC+GC}(\mathcal{S}_o,\mathcal{S}_b)|}$, $RC = \frac{|\mathcal{N}_{ORFC}(\mathcal{S}_o,\mathcal{S}_b)|}{|A_{ORFC+GC}(\mathcal{S}_o,\mathcal{S}_b)|}$ for $ORFC + GC$, and $RC = \frac{|\mathcal{N}_{ORFC}(\mathcal{S}_o,\mathcal{S}_b)|}{|A_{ORFC_{Core}+GC}(\mathcal{S}_o,\mathcal{S}_b)|}$ for $ORFC_{Core} + GC$.

In the quantitative experiments, we adopted the weight assignment $\delta(a,b) = K - |G(a) + G(b)|$, where $G(a)$ denotes the gradient magnitude of the Sobel operator, and dual (maximiza-

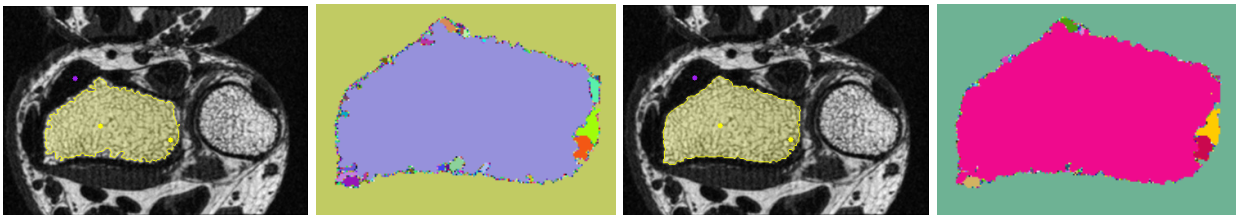


Figure 6.7: *An MR image of a wrist with two seed pixels selected inside the bone. (a) ORFC segmentation with RC = 99.17%. (b) OIFT segmentation with RC = 95.40%. (c) ORFC cores inside the OIFT mask. (d) OIFT cores.*

tion) IFT. For approaches based on directed graphs, we used $\alpha = -0.5$, for the foot bones (transitions from dark to bright pixels) and $\alpha = 0.5$ for the spinal-vertebra; and $\alpha = 0.0$ in the case of undirected approaches.

Figure 6.8 shows the experimental results. Note that the robustness coefficient of RFC is always 100%, since $\mathcal{N}_{RFC}(\mathcal{S}_o, \mathcal{S}_b) = A_{RFC}(\mathcal{S}_o, \mathcal{S}_b)$ [24]. For the bones datasets, with respect to the Dice measure, $OIFT$ is among the first three methods, losing only to the hybrid methods $ORFC + GC$ and $ORFC_{Core} + GC$. However, with respect to the robustness coefficient, $OIFT$ usually gives better results than $ORFC + GC$ and $ORFC_{Core} + GC$, losing only to RFC and $ORFC$. For the spinal-vertebra, the Dice values of all methods decrease rapidly because the object has thin parts and the erosion process rapidly eliminates seeds in several important regions of the object. $OIFT$ has the best Dice values for the spinal-vertebra, and the third best robustness coefficient. So we can conclude that $OIFT$ has a good balance between accuracy and robustness.

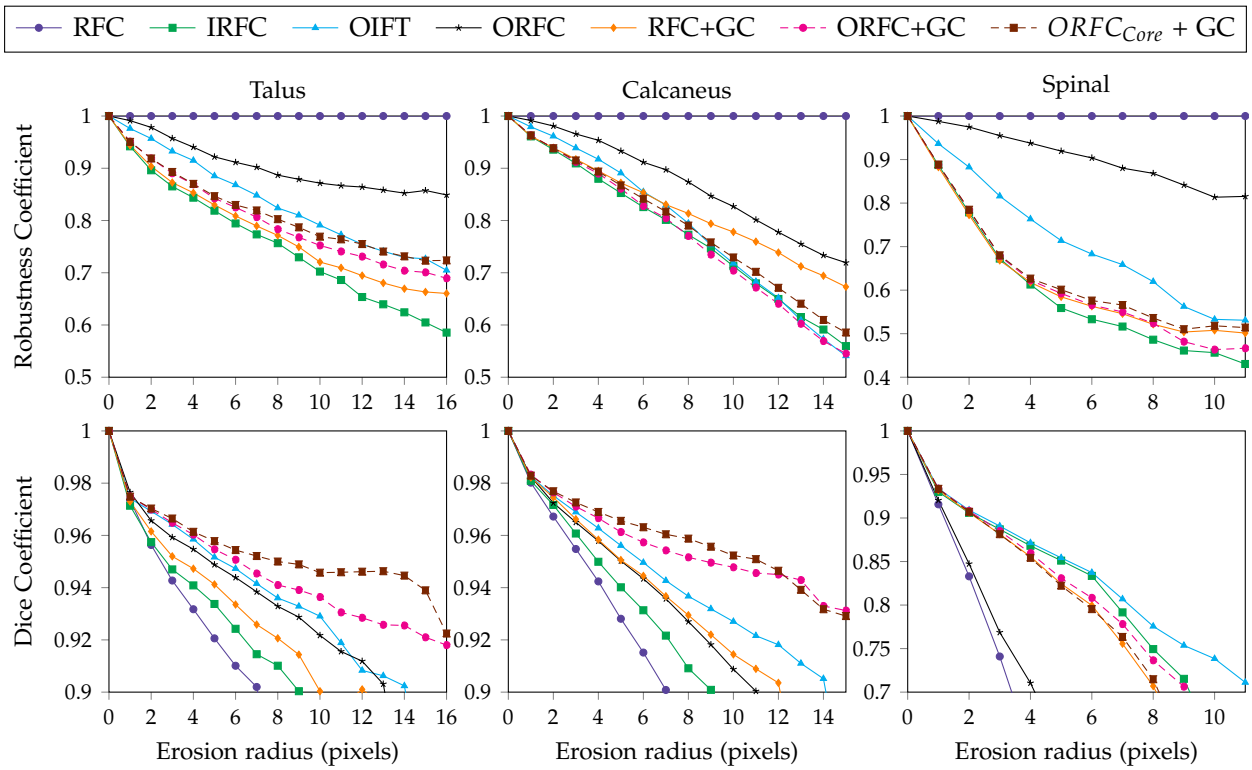


Figure 6.8: *The mean robustness coefficient curves and the mean accuracy curves (Dice coefficient), using non-equally eroded-dilated seeds, for segmenting talus, calcaneus, and spinal-vertebra.*

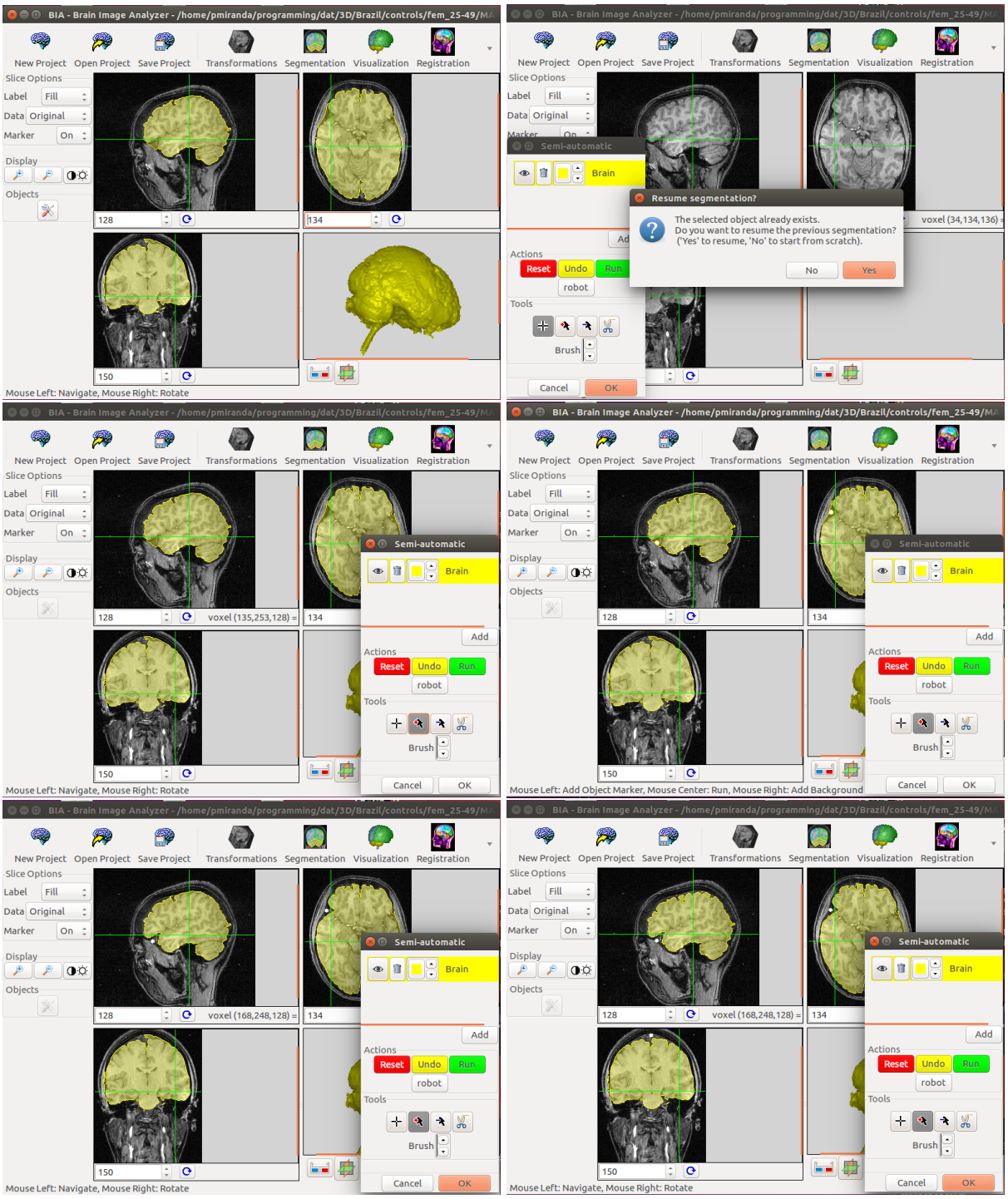


Figure 6.9: *Repairing a 3D segmentation by IFT-SLIC: loading a mask, continuing a segmentation, automatically estimating seeds, adding a new internal seed, processing a new mask and repeating the process in a differential way.*

6.2.1 Redundancy Analysis

Sections 4.1 and 4.2 describe how calculate cores $\mathcal{N}_{ORFC}(\mathcal{S}_o, \mathcal{S}_b)$ and $\mathcal{N}_{OIFT}(\mathcal{S}_o, \mathcal{S}_b)$. If each core is replaced by just one seed, a built seed set can be used in a interactive repair. Besides that, if a core is contained inside the segmentation of another one, the first core is considered redundant and could be discarded. This process is denoted *redundancy analysis*.

After core computation and redundancy analysis, a minimal set of seeds is returned. Following Miranda et al. [12, 13] notation, we use $\mathcal{N}_1 \propto \mathcal{N}_2$ to represent a core $\mathcal{N}_1$ redundant to $\mathcal{N}_2$. This is transitive ($\mathcal{N}_1 \propto \mathcal{N}_2 \wedge \mathcal{N}_2 \propto \mathcal{N}_3 \implies \mathcal{N}_1 \propto \mathcal{N}_3$). Any cycle of redundancies implies in equivalence and $t \propto s$ means that node t is redundant in relation to s, so: $t \propto s \wedge s \propto t \iff s \equiv t$.

Cores of ORFC are SCCs of $G_>$ after removing arcs $\langle s,t \rangle$ from G where $\omega(\langle s,t \rangle) \leq \varepsilon^{\downarrow}_{\infty}$ and $C_{opt}(s) = C_{opt}(t) = \varepsilon^{\downarrow}_{\infty}$. Note that we do not need to compute $\varepsilon^{\downarrow}_{\infty}$, we just test $\omega(\langle s,t \rangle) > C_{opt}(s) \wedge C_{opt}(s) = C_{opt}(t)$, removing $\langle s,t \rangle$ otherwise. In A_{ORFC} (Algorithm 3), an internal seed s results in its DCC. If $t \propto s$, then $t \in DCC_{G_>}(s)$. We might think that $t \in DCC_{G_>}(s)$ is not only necessary but sufficient for $t \propto s$ to be true, but Figure 6.10 shows a counterexample.

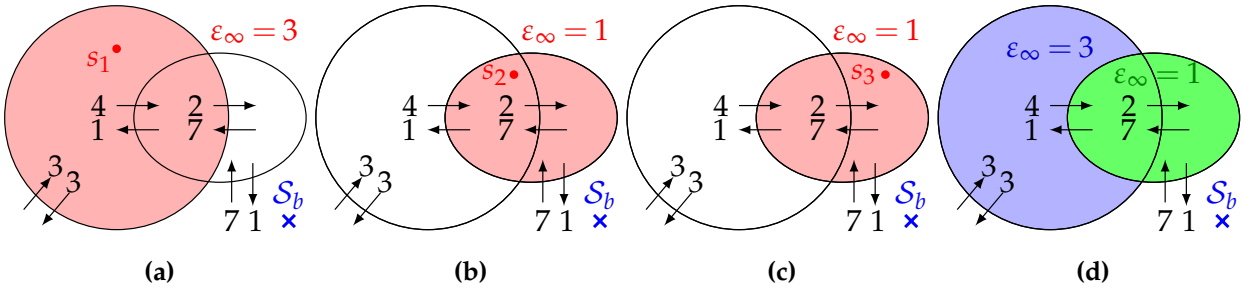


Figure 6.10: *Counterexample to show that $t \in DCC_{G_>}(s)$ does not imply $t \propto s$. (a-b) Even though $s_2 \in DCC_{G_>}(s_1)$, (c-d) $s_3 \in \mathcal{N}(\{s_2\}, \mathcal{S}_b)$ but $s_3 \notin \mathcal{N}(\{s_1\}, \mathcal{S}_b)$, so s_2 is not redundant in relation to s_1.*

Let $t \in DCC_{G_>}(s)$. For any $k \in DCC_{G_>}(t)$, there is a path $\pi_{t \rightsquigarrow k} = \langle t = v_1, \ldots, v_n = k \rangle$, where $\omega(\langle v_i, v_{i+1} \rangle) > C_{opt}(t), 1 \leq i \leq n$. If $C_{opt}(t) \geq C_{opt}(s)$, then there is a path from s to k whose arcs have weight values higher than C_{opt}(s). In this case, $k \in DCC_{G_>}(s)$, and $DCC_{G_>}(t) \subseteq DCC_{G_>}(s)$ (from Proposition 1 of [33]). As $\mathcal{N}_{ORFC}(\{t\}, \mathcal{S}_b) \subseteq DCC_{G_>}(t)$, then we can formulate Proposition 3.

Proposition 3. $t \in DCC_{G_>}(s) \wedge C_{opt}(t) \geq C_{opt}(s) \implies \mathcal{N}_{ORFC}(\{t\}, \mathcal{S}_b) \propto \mathcal{N}_{ORFC}(\{s\}, \mathcal{S}_b)$.

So, after computing $\mathcal{N}_{ORFC}$ by Algorithm 4, we create a *Region Adjacency Graph* (RAG) and remove node $\mathcal{N}_t$ (which represents core $\mathcal{N}_{ORFC}(\{t\}, \mathcal{S}_b)$) in this RAG if $\exists \langle \mathcal{N}_s, \mathcal{N}_t \rangle$ where $\omega(\langle \mathcal{N}_s, \mathcal{N}_t \rangle) > C_{opt}(s)$ and $C_{opt}(t) \geq C_{opt}(s)$.

We might think the other side of Proposition 3 is true: $\mathcal{N}_{ORFC}(\{t\}, \mathcal{S}_b) \propto \mathcal{N}_{ORFC}(\{s\}, \mathcal{S}_b) \implies t \in DCC_{G_>}(s) \wedge C_{opt}(t) \geq C_{opt}(s)$. However, Figure 6.11 shows a counterexample. It is similar to Figure 6.10, but one arc had its weighted changed. In this example, we just need $\mathcal{N}_{ORFC}(\{s_1\}, \mathcal{S}_b)$.

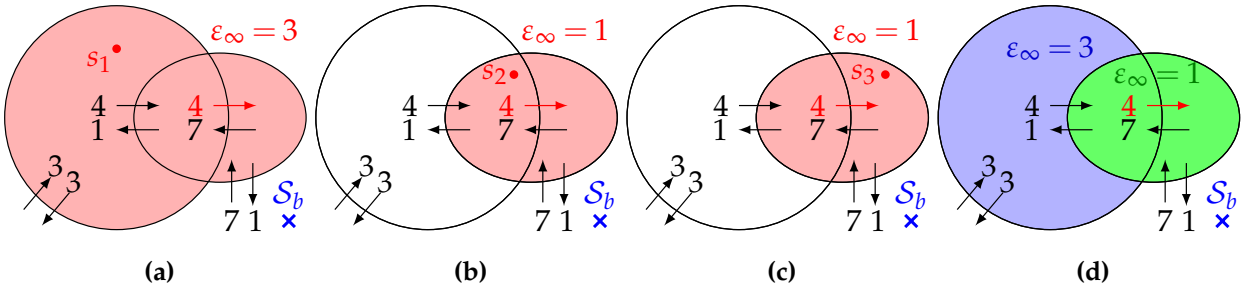


Figure 6.11: *Counterexample to show that $\mathcal{N}(\{t\}, \mathcal{S}_b) \propto \mathcal{N}(\{s\}, \mathcal{S}_b)$ does not imply $t \in DCC_{G_>}(s) \wedge C_{opt}(t) \geq C_{opt}(s)$. Even though $C_{opt}(s_2) < C_{opt}(s_1)$ (that is, $1 < 3$), $\mathcal{N}(\{s_2\}, \mathcal{S}_b) \propto \mathcal{N}(\{s_1\}, \mathcal{S}_b)$.*

It means that we have developed a method which can reduce the number of estimated seeds, but we cannot assert that it is the smallest possible seed set. Further research is needed to find a method to return the smallest estimated seed set, even if $C_{opt}(t) < C_{opt}(s)$.

The same algorithm can be applied in the RAG of $\mathcal{N}_{OIFT}$. Figure 6.12 shows OIFT applied to the same graph of Figure 6.10. It suggest we could remove the restriction $C_{opt}(t) \geq C_{opt}(s)$ and test in the RAG the condition $\omega(\langle \mathcal{N}_s, \mathcal{N}_t \rangle) > C_{opt}(t)$. The validation of this hypothesis is another source of future research.

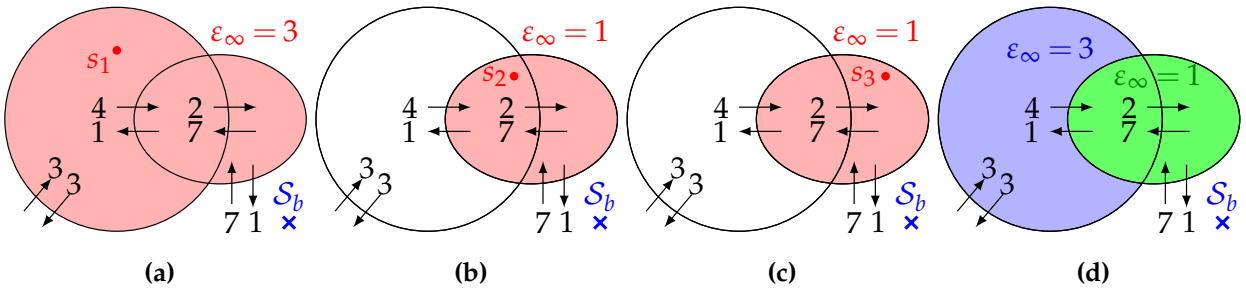


Figure 6.12: *Example of OIFT applied to the graph of Figure 6.10. Note that there is an arc $\langle s, t \rangle$ between $\mathcal{N}_{OIFT}(\{s_1\}, \mathcal{S}_b)$ and $\mathcal{N}_{OIFT}(\{s_2\}, \mathcal{S}_b)$ where $\omega(\langle s, t \rangle) = 4 > 1 = C_{opt}(s_2)$, which implies $\mathcal{N}_{OIFT}(\{s_2\}, \mathcal{S}_b) \subseteq A_{OIFT}(\{s_1\}, \mathcal{S}_b)$.*

Chapter 7

Conclusion

We have developed techniques to allow the user edit the segmentation previously generated from automatic, interactive or manual methods. Instead of editing it manually or discarding it to generate another, which is cumbersome, Image-Foresting Transform (IFT) can reduce the effort in a interactive way.

Initial seed are automatically estimated by using two approaches: IFT-SLIC (Section 3.4), which moves a grid of equally spaced seeds to partition the graph into supernodes, and center of cores (Chapter 4), regions of redundant seeds, returning a small set of seeds (one for each core). We can also reduce this set, by discarding the cores whose segmentations are contained in the results obtained from other cores.

Results of Chapter 6 shows experiments for both approaches, validating the hypothesis and showing the potential of the method. The method of repairing segmentations via OIFT/ORFC should be used whenever we wish to continue a previous segmentation obtained also from OIFT/ORFC. IFT-SLIC is a good choice for images with field inhomogeneity and low boundary contrast, as it is based on an additive function of relative intensities. OIFT is indicated for images with boundary polarity well defined (from dark to bright or vice-versa).

The reader may try some implementations like our @kv[1] library, both in Git versioning system. Our library is briefly described in Section 7.1.

7.1 Contributions

- Application and investigation of IFT-SLIC [34] method in obtaining the seed set for interactive segmentation repair. Features such as compactness and boundary adherence are benefits of using this method. The format of partitions, compared to other methods, is more regular [35] (Section 6.1);
- Theoretical analysis of the ORFC robustness and development of the algorithm that calculates the ORFC core, with proof of correctness [36] (Section 4.1);
- Theoretical analysis of the OIFT robustness and development of the algorithm that calculates the OIFT core, with proof of correctness [37] (Section 4.2);
- Development of the hybrid method $ORFC_{Core} + GC$, and comparative evaluation with other methods [36] (Chapter 5);
- Core redundancy analysis and algorithm development [36,37] (Chapter 6);
- Segmentation repair algorithm (for OIFT and ORFC) using the algorithms of core calculation and redundancy analysis (Chapter 6);

[1] https://atkv.github.io/

- Definition of robustness index of methods [37] (Section 4.3);
- Development of the @kv framework, with described algorithms implemented in C11 for computer vision problems. The goal of @kv is to build an Application Program Interface (API), compiled in a set of shared libraries to be used in any project. NIFTI and DICOM parsers, IFT, OIFT, ORFC, OIFT_{Core}, ORFC_{Core}, IFT-SLIC, primal and dual approaches, as well as reading and writing PNG, JPG, PPM and PGM also are available. Charts and widgets made in GTK+3 for visualization purposes also are developed. The user can save/read intermediate results from/to a compressed file (with .atz extension). Most of the algorithms come with unit tests. A documentation for the API as well as some tutorials also are available. Wrappers for other languages (i.e., Python and Java) also are in the roadmap. A WebAssembly[2] module also could be an interesting way for web apps to provide binary and compiled IFT algorithms for the web, although WebAssembly is a very new and developing technology.

7.2 Difficulties

- The lack of libraries for IFT developed over a Version Control System (VCS) like Git makes modifying and extending it with implementations of studied algorithms some kind difficult. We tackle this problem by developing a new library which use VCS so that users can use any version of the system, even legacy ones;
- Comparison between robustness of different segmentation methods is not trivial, because, differently from our IFT-based approaches, most methods do not have a known efficient algorithm to compute their cores. As this is an ongoing work, more research is needed to get a more general picture of relations between these methods.

7.3 Future Work

As future work, we plan to investigate online training with IFT with Cores. For example, in a dataset of 50 similar images (i.e., segmentation of liver), the effort of segmenting subsequent images could be reduced by exploiting the previous results to train a classifier in order to update the IFT parameters, connectivity functions and weights. Cores can reduce the search space of parameters in the training phase.

Another interesting research may be the impact of non-smooth connectivity functions to robustness of the method. As it is a diverse class of functions, a careful analysis could lead to improvements for many applications which uses these functions.

Experiments for validating the hypothesis of reducing the number of seeds when applying redundancy analysis to ORFC and OIFT is another future work. The reduction ratio may be one of the metrics for comparison between RISP with and without redundancy analysis.

A more comprehensive study between segmentation methods can be improved with robustness analysis. Computing cores of methods other than IFT-based ones also may improve understanding the relations.

[2] http://webassembly.org

Bibliography

[1] R. C. Gonzalez and R. E. Woods, *Digital image processing*, 3rd ed. Pearson, 2007. 1, 2

[2] S. J. D. Prince, *Computer Vision: Models, Learning, and Inference*, 1st ed. Cambridge University Press, 2012. 1

[3] K. D. Toennies, *Guide to medical image analysis: Methods and algorithms*. Springer, 2012. DOI:10.1007/978-1-4471-2751-2 2

[4] P. Suetens, *Fundamentals of Medical Imaging*, 2nd ed. New York: Cambridge University Press, 2009, vol. 5. 2

[5] A. M. Dale, B. Fischl, and M. I. Sereno, "Cortical surface-based analysis. I. Segmentation and surface reconstruction." *NeuroImage*, vol. 9, no. 2, pp. 179–94, feb 1999. DOI:10.1006/nimg.1998.0395 2

[6] R. S. J. Frackowiak, *Human brain function*. Elsevier Academic Press, 2004. 2

[7] R. Beichel, C. Bauer, A. Bornik, E. Sorantin, and H. Bischof, "Segmentation of Diseased Livers: A 3D Refinement Approach," in *Handbook of Biomedical Imaging*, ser. Lecture Notes in Computer Science, K. Mori, I. Sakuma, Y. Sato, C. Barillot, and N. Navab, Eds. Boston, MA: Springer US, 2015, vol. 8151, ch. IV, pp. 403–412. 2

[8] M. P. Jackowski, M. Satter, and A. Goshtasby, "Approximating digital 3D shapes by rational Gaussian surfaces," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 1, pp. 56–69, jan 2003. DOI:10.1109/TVCG.2003.1175097 2, 4, 5

[9] Y. Kang, K. Engelke, and W. A. Kalender, "Interactive 3D editing tools for image segmentation," *Medical Image Analysis*, vol. 8, no. 1, pp. 35–46, mar 2004. DOI:10.1016/j.media.2003.07.002 2

[10] W. Valenzuela, S. J. Ferguson, D. Ignasiak, G. Diserens, P. Vermathen, C. Boesch, and M. Reyes, "Correction tool for Active Shape Model based lumbar muscle segmentation," in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, vol. 2015. IEEE, aug 2015, pp. 3033–3036. DOI:10.1109/EMBC.2015.7319031 2, 4, 5

[11] H.-F. Yang and Y. Choe, *An Interactive Editing Framework for Electron Microscopy Image Segmentation*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, vol. 6938, pp. 400–409. 2, 4, 5

[12] P. A. V. Miranda, A. X. Falcão, and G. C. S. Ruppert, "How to Complete Any Segmentation Process Interactively via Image Foresting Transform," in *2010 23rd SIBGRAPI Conference on Graphics, Patterns and Images*. IEEE, aug 2010, pp. 309–316. DOI:10.1109/SIBGRAPI.2010.48 2, 3, 4, 5, 22, 38

[13] P. A. V. Miranda, A. X. Falcão, G. C. S. Ruppert, and F. A. Cappabianco, "How to fix any 3D segmentation interactively via Image Foresting Transform and its use in MRI brain segmentation," in *2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*. Chicago, IL: IEEE, mar 2011, pp. 2031–2035. DOI:10.1109/ISBI.2011.5872811 2, 3, 4, 5, 31, 32, 33, 38

[14] N. El-Zehiry, M.-P. Jolly, and M. Sofka, "A splice-guided data driven interactive editing," in *2013 IEEE 10th International Symposium on Biomedical Imaging*. IEEE, apr 2013, pp. 1098–1101. DOI:10.1109/ISBI.2013.6556670 2, 4

[15] A. P. Harrison, N. Birkbeck, and M. Sofka, *IntellEditS: Intelligent Learning-Based Editor of Segmentations*. Nagoya, Japan: Springer Berlin Heidelberg, 2013, pp. 235–242. 2, 4

[16] D. Maleike, M. Nolden, H.-P. H.-P. Meinzer, and I. Wolf, "Interactive segmentation framework of the Medical Imaging Interaction Toolkit," *Computer Methods and Programs in Biomedicine*, vol. 96, no. 1, pp. 72–83, oct 2009. DOI:10.1016/j.cmpb.2009.04.004 2

[17] A. Karimov, G. Mistelbauer, T. Auzinger, and S. Bruckner, "Guided Volume Editing based on Histogram Dissimilarity," *Computer Graphics Forum*, vol. 34, no. 3, pp. 91–100, jun 2015. DOI:10.1111/cgf.12621 2, 4, 5

[18] J. Cousty, G. Bertrand, L. Najman, and M. Couprie, "Watershed Cuts: Thinnings, Shortest Path Forests, and Topological Watersheds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 5, pp. 925–939, may 2010. DOI:10.1109/TPAMI.2009.71 2

[19] R. A. Lotufo, A. X. Falcão, and F. Zampirolli, "IFT-Watershed from gray-scale marker," in *Proceedings. XV Brazilian Symposium on Computer Graphics and Image Processing*. IEEE Comput. Soc, 2002, pp. 146–152. DOI:10.1109/SIBGRA.2002.1167137 2, 15

[20] Y. Boykov and G. Funka-Lea, "Graph Cuts and Efficient N-D Image Segmentation," *International Journal of Computer Vision*, vol. 70, no. 2, pp. 109–131, nov 2006. DOI:10.1007/s11263-006-7934-5 2, 27, 28

[21] Y. Boykov and M.-p. Jolly, "Interactive Organ Segmentation Using Graph Cuts," in *Springer-Verlag*, 2000, pp. 276–286. 2

[22] R. Audigier and R. Lotufo, "Seed-Relative Segmentation Robustness of Watershed and Fuzzy Connectedness Approaches," in *XX Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI 2007)*. IEEE, oct 2007, pp. 61–70. DOI:10.1109/SIBGRAPI.2007.26 2, 22

[23] K. C. Ciesielski, J. K. Udupa, P. K. Saha, and Y. Zhuge, "Iterative Relative Fuzzy Connectedness for Multiple Objects with Multiple Seeds." *Computer vision and image understanding : CVIU*, vol. 107, no. 3, pp. 160–182, sep 2007. DOI:10.1016/j.cviu.2006.10.005 2, 28, 35

[24] P. K. Saha and J. K. Udupa, "Relative Fuzzy Connectedness among Multiple Objects: Theory, Algorithms, and Applications in Image Segmentation," *Computer Vision and Image Understanding*, vol. 82, no. 1, pp. 42–56, apr 2001. DOI:10.1006/cviu.2000.0902 2, 28, 35, 36

[25] L. Grady, "Random Walks for Image Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 11, pp. 1768–1783, nov 2006. DOI:10.1109/TPAMI.2006.233 2, 4

[26] X. Bai and G. Sapiro, "Distancecut: Interactive Segmentation and Matting of Images and Videos," in *2007 IEEE International Conference on Image Processing*, vol. 2. IEEE, 2007, pp. II – 249–II – 252. DOI:10.1109/ICIP.2007.4379139 2

[27] A. X. Falcão, J. Stolfi, and R. A. Lotufo, "The Image Foresting Transform: Theory, Algorithms, and Applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 19–29, jan 2004. DOI:10.1109/TPAMI.2004.1261076 2, 3, 10, 15, 17

[28] V. Vezhnevets, V. Vezhnevets, and E. L., "GrowCut – Interactive Multi-Label N-D Image Segmentation By Cellular Automata," in *Proc. of GraphiCon*, Novosibirsk Akademgorodok, 2005. DOI:10.1.1.59.8092 2, 15

[29] P. A. V. Miranda and A. X. Falcão, "Links Between Image Segmentation Based on Optimum-Path Forest and Minimum Cut in Graph," *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 128–142, oct 2009. DOI:10.1007/s10851-009-0159-9 2, 3, 25, 33

[30] A. X. Falcão and F. P. Bergo, "Interactive Volume Segmentation With Differential Image Foresting Transforms," *IEEE Transactions on Medical Imaging*, vol. 23, no. 9, pp. 1100–1108, sep 2004. DOI:10.1109/TMI.2004.829335 2, 5, 32

[31] L. A. C. Mansilla and P. A. V. Miranda, "Image segmentation by oriented image foresting transform: Handling ties and colored images," in *18th International Conference on Digital Signal Processing (DSP)*. Santorini: IEEE, 2013. DOI:10.1109/ICDSP.2013.6622806 3, 17, 21, 28, 35

[32] P. A. V. Miranda and L. A. C. Mansilla, "Oriented Image Foresting Transform Segmentation by Seed Competition," *IEEE Transactions on Image Processing*, vol. 23, no. 1, pp. 389–398, jan 2014. DOI:10.1109/TIP.2013.2288867 3, 17, 21

[33] H. H. C. Bejar and P. A. V. Miranda, "Oriented relative fuzzy connectedness: theory, algorithms, and its applications in hybrid image segmentation methods," *EURASIP Journal on Image and Video Processing*, vol. 2015, no. 1, p. 21, dec 2015. DOI:10.1186/s13640-015-0067-4 3, 18, 23, 27, 28, 35, 38

[34] E. B. Alexandre, A. S. Chowdhury, P. A. V. Miranda, and A. X. Falcão, "IFT-SLIC: A general framework for superpixel generation based on simple linear iterative clustering and image foresting transform," in *SIBGRAPI*, Salvador, 2015. 3, 6, 19, 41

[35] A. C. M. Tavares, P. A. V. Miranda, T. V. Spina, and A. X. Falcão, "A Supervoxel-based Solution to Resume Segmentation for Interactive Correction by Differential Image-Foresting Transforms," in *13th International Symposium on Mathematical Morphology*. Fontainebleau, France: Springer, 2017. DOI:10.1007/978-3-319-57240-6_9 3, 6, 41

[36] A. C. M. Tavares, H. H. C. Bejar, and P. A. V. Miranda, "Seed Robustness of Oriented Relative Fuzzy Connectedness: Core Computation and its Applications," in *SPIE Medical Imaging*, Orlando, EUA, 2017. DOI:10.1117/12.2254646 3, 6, 35, 41

[37] A. C. M. Tavares, H. H. C. Bejar, and P. A. V. Miranda, "Seed Robustness of Oriented Image Foresting Transform: Core Computation and the Robustness Coefficient," in *13th International Symposium on Mathematical Morphology*. Fontainebleau, France: Springer, 2017. DOI:10.1007/978-3-319-57240-6_10 3, 6, 41, 42

[38] F. Heckel, J. H. Moltz, C. Tietjen, and H. K. Hahn, "Sketch-Based Editing Tools for Tumour Segmentation in 3D Medical Images," *Computer Graphics Forum*, vol. 32, no. 8, pp. 144–157, dec 2013. DOI:10.1111/cgf.12193 4

[39] T. V. Spina, S. B. Martins, and A. X. Falcão, "Interactive Medical Image Segmentation by Statistical Seed Models," in *SIBGRAPI*, São José dos Campos, SP, Brazil, 2016. DOI:10.1109/SIBGRAPI.2016.42 4, 5

[40] L. Grady and G. Funka-Lea, "An energy minimization approach to the data driven editing of presegmented images/volumes." *Medical image computing and computer-assisted intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention*, vol. 9, no. Pt 2, pp. 888–95, jan 2006. 4

[41] V. Gulshan, C. Rother, A. Criminisi, A. Blake, and A. Zisserman, "Geodesic star convexity for interactive image segmentation," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, jun 2010, pp. 3129–3136. DOI:10.1109/CVPR.2010.5540073 5, 14, 33

[42] R. Tarjan, "Depth-First Search and Linear Graph Algorithms," *SIAM Journal on Computing*, vol. 1, no. 2, pp. 146–160, jun 1972. DOI:10.1137/0201010 9, 23

[43] P. A. V. Miranda, A. X. Falcão, and J. K. Udupa, "Synergistic arc-weight estimation for interactive image segmentation using graphs," *Computer Vision and Image . . .*, 2010. 11, 28

[44] K. C. Ciesielski and J. K. Udupa, "Affinity functions in fuzzy connectedness based image segmentation II: Defining and recognizing truly novel affinities," *Computer Vision and Image Understanding*, vol. 114, no. 1, pp. 155–166, jan 2010. DOI:10.1016/j.cviu.2009.09.005 11, 28

[45] L. R. Dice, "Measures of the Amount of Ecologic Association Between Species," *Ecology*, vol. 26, no. 3, pp. 297–302, jul 1945. DOI:10.2307/1932409 13

[46] V. Labatut and H. Cherifi, "Accuracy Measures for the Comparison of Classifiers," jul 2012. 13

[47] P. E. Rauber, A. X. Falcão, T. V. Spina, and P. J. de Rezende, "Interactive Segmentation by Image Foresting Transform on Superpixel Graphs," in *2013 XXVI Conference on Graphics, Patterns and Images*. IEEE, aug 2013, pp. 131–138. DOI:10.1109/SIBGRAPI.2013.27 14

[48] A. K. Sinop and L. Grady, "A Seeded Image Segmentation Framework Unifying Graph Cuts And Random Walker Which Yields A New Algorithm," in *2007 IEEE 11th International Conference on Computer Vision*. IEEE, 2007, pp. 1–8. DOI:10.1109/ICCV.2007.4408927 14

[49] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network flows : theory, algorithms, and applications*, 1st ed. Prentice Hall, 1993. 15

[50] E. W. Dijkstra and E. W., "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, dec 1959. DOI:10.1007/BF01386390 15

[51] F. A. Cappabianco, G. Araujo, and A. X. Falcão, "The Image Forest Transform Architecture," in *2007 International Conference on Field-Programmable Technology*. IEEE, dec 2007, pp. 137–144. DOI:10.1109/FPT.2007.4439242 15

[52] A. X. Falcão, B. S. Cunha, and R. A. Lotufo, "Design of connected operators using the image foresting transform," in *Medical Imaging*, M. Sonka and K. M. Hanson, Eds., jul 2001, pp. 468–479. DOI:10.1117/12.431120 15, 22, 28

[53] A. X. Falcão, J. K. Udupa, S. Samarasekera, S. Sharma, B. E. Hirsch, and R. A. Lotufo, "User-Steered Image Segmentation Paradigms: Live Wire and Live Lane," *Graphical Models and Image Processing*, vol. 60, no. 4, pp. 233–260, 1998. DOI:10.1006/gmip.1998.0475 15

[54] P. A. V. Miranda, A. X. Falcão, and T. V. Spina, "Riverbed: A Novel User-Steered Image Segmentation Method Based on Optimum Boundary Tracking," *IEEE Transactions on Image Processing*, vol. 21, no. 6, pp. 3042–3052, jun 2012. DOI:10.1109/TIP.2012.2188034 15

[55] P. A. V. Miranda and A. X. Falcão, "Elucidating the Relations among Seeded Image Segmentation Methods and their Possible Extensions," in *2011 24th SIBGRAPI Conference on Graphics, Patterns and Images*. Maceio, Alagoas: IEEE, aug 2011, pp. 289–296. DOI:10.1109/SIBGRAPI.2011.13 15

[56] K. C. Ciesielski, J. K. Udupa, A. X. Falcão, and P. A. V. Miranda, "Fuzzy Connectedness Image Segmentation in Graph Cut Formulation: A Linear-Time Algorithm and a Comparative Analysis," *Journal of Mathematical Imaging and Vision*, vol. 44, no. 3, pp. 375–398, nov 2012. DOI:10.1007/s10851-012-0333-3 15

[57] L. M. Rocha, F. A. M. Cappabianco, and A. X. Falcão, "Data clustering as an optimum-path forest problem with applications in image analysis," *International Journal of Imaging Systems and Technology*, vol. 19, no. 2, pp. 50–68, jun 2009. DOI:10.1002/ima.20191 15

[58] J. P. Papa, A. X. Falcão, P. A. V. Miranda, C. T. N. Suzuki, and N. D. A. Mascarenhas, "Design of robust pattern classifiers based on optimum-path forests," in *Proceedings of the 8th International Symposium on Mathematical Morphology*, vol. 1. Rio de Janeiro: MCT/INPE, 2007, pp. 337–348. 15

[59] C. d. M. Braz and P. A. V. Miranda, "Image segmentation by image foresting transform with geodesic band constraints," in *2014 IEEE International Conference on Image Processing (ICIP)*. IEEE, oct 2014, pp. 4333–4337. DOI:10.1109/ICIP.2014.7025880 17

[60] L. A. C. Mansilla and P. A. V. Miranda, "Oriented Image Foresting Transform Segmentation: Connectivity Constraints with Adjustable Width," in *2016 29th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. IEEE, oct 2016, pp. 289–296. DOI:10.1109/SIBGRAPI.2016.047 17

[61] L. A. C. Mansilla, M. P. Jackowski, and P. A. V. Miranda, "Image foresting transform with geodesic star convexity for interactive image segmentation," in *IEEE International Conference on Image Processing*. IEEE, sep 2013, pp. 4054–4058. DOI:10.1109/ICIP.2013.6738835 17

[62] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "SLIC Superpixels Compared to State-of-the-Art Superpixel Methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, nov 2012. DOI:10.1109/TPAMI.2012.120 19

[63] L. A. C. Mansilla, P. A. V. Miranda, and F. A. Cappabianco, "Image Segmentation by Image Foresting Transform with Non-smooth Connectivity Functions," in *2013 XXVI Conference on Graphics, Patterns and Images*. IEEE, aug 2013, pp. 147–154. DOI:10.1109/SIBGRAPI.2013.29 19

[64] L. R. Ford and D. R. Fulkerson, "Maximal flow through a network," *Canadian Journal of Mathematics*, vol. 8, pp. 399–404, jan 1956. DOI:10.4153/CJM-1956-045-5 27

[65] A. V. Goldberg and R. E. Tarjan, "A new approach to the maximum flow problem," in *Proceedings of the eighteenth annual ACM symposium on Theory of computing - STOC '86*. New York, New York, USA: ACM Press, 1986, pp. 136–146. DOI:10.1145/12130.12144 27

[66] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1124–1137, sep 2004. DOI:10.1109/TPAMI.2004.60 28

[67] C. Couprie, L. Grady, L. Najman, and H. Talbot, "Power watersheds: A new image segmentation framework extending graph cuts, random walker and optimal spanning forest," in *2009 IEEE 12th International Conference on Computer Vision*, no. Iccv. IEEE, sep 2009, pp. 731–738. DOI:10.1109/ICCV.2009.5459284 28

[68] K. C. Ciesielski, P. A. V. Miranda, A. X. Falcão, and J. K. Udupa, "Joint graph cut and relative fuzzy connectedness image segmentation algorithm," *Medical Image Analysis*, vol. 17, no. 8, pp. 1046–1057, dec 2013. DOI:10.1016/j.media.2013.06.006 28, 35

[69] K. C. Ciesielski and J. K. Udupa, "Affinity functions in fuzzy connectedness based image segmentation I: Equivalence of affinities," *Computer Vision and Image Understanding*, vol. 114, no. 1, pp. 146–154, 2010. DOI:10.1016/j.cviu.2009.09.006 28

Index