UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

VÁSQUEZ ROSALES DIONICIO ÁNGEL

Stochastic Methods for Image Segmentation Based on Stochastic Superpixels

Thesis presented in partial fulfillment
of the requirements for the degree of
Doctor of Computer Science

Advisor: Prof. Dr. Jacob Scharcanski

Porto Alegre
September 2020

CIP — CATALOGING-IN-PUBLICATION

Dionicio Ángel, Vásquez Rosales

Stochastic Methods for Image Segmentation Based on Stochastic Superpixels / Vásquez Rosales Dionicio Ángel. – Porto Alegre: PPGC da UFRGS, 2020.

121 f.: il.

Thesis (Ph.D.) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2020. Advisor: Jacob Scharcanski.

1. Superpixels. 2. Stochastic graph contraction. 3. Superpixels graph adjacency. 4. Multi-scale segmentation. 5. Hierarchical representation. 6. Eigen-decomposition. I. Scharcanski, Jacob. II. Título.

Dedicated to my parents, Ángela and Dionicio,
and to my siblings, Juliana, Isolina, Yissa, Manuel, Ray, Kevin and Angie.
Thank you so much for your infinite love, unconditional support and those phrases about perseverance that my lovely mother always told us since we were kids in school.

ACKNOWLEDGMENT

I would like to express my gratitude to all those who contributed to complete this work. Primarily, to my parents and siblings for their unconditional support and comprehension during all stages of my academic life.

Moreover, I sincerely thank to my advisor Dr. Jacob Scharcanski (UFRGS) and to Dr. Alexander Wong (U. Waterloo) for their guidance, comments and reviews of this PhD thesis. Also, I would like to thank to Dr. Eduardo Antônio Barros da Silva, Dr. Cláudio Jung, Dr. John Soldera and Dr. Luciano da Silva, for their suggestions to improve this work; and to all my professors and colleagues at UFRGS for the knowledge transmitted during my graduate studies.

Finally, I would like to thank Coordenação de Aperfeiçoamento de Nível Superior (CAPES) for the financial support.

ABSTRACT

Superpixel generation is one of the most widely used pre-processing step for computer vision tasks, ranging from image segmentation to saliency detection and object tracking. The use of superpixel representations reduce greatly informational redundancy by creating a condensed representation of the scene, and it also enables greater scalability by significantly reducing the computational complexity required to perform computer vision tasks. A number of different superpixel generation approaches have been proposed in literature, and the biggest challenges faced by the existing approaches are: i) poor adherence to the object boundaries, ii) difficulty in generating well-structured superpixels under non-ideal scene conditions, iii) weak boundary separation between objects, and iv) difficulty to obtain hierarchical and nested image representations. Motivated to tackle these key challenges, this thesis introduces new methods for image segmentation problem, in relation to the task of multi-scale superpixels generation from natural images. First, an Iterative Hierarchical Stochastic Graph Contraction (IHSGC) method is proposed which uses a stochastic strategy to generate multi-scale superpixels, and each superpixel is represented by a hierarchical structure and describes an image patch at fine and coarse scales simultaneously. The proposed method consists of two main steps: an initialization step based on a multi-channel unsupervised stochastic over-segmentation at the pixel level preserving the local spatial relationships at finer scale; and an iterative hierarchical stochastic graph contraction step for coarser scales generation by graph contraction operations until the desired number of superpixels is obtained. Second, is proposed an improved version of stochastic superpixel generation method based on stochastic graph contraction operations (SGC) which uses simple features to generate coarser scales and improve the first version (IHSGC) in terms of boundary adherence and running time. Finally, a Stochastic Spectral Graph Contraction approach (SSGC) is proposed as application of stochastic superpixels, to handle the scalability problem of the spectral image segmentation related to the eigen-decomposition process. Each superpixel is described using a 3D normalized CIELAB color histogram, and a superpixels adjacency graph is built. Then, stochastic contraction operations on superpixels graph structure and on a sub-space defined by K-eigenvectors are used to obtain K image regions. The experimental results based on the popular Berkeley segmentation databases BSDS300 and BSDS500 suggest that the proposed stochastic approaches potentially can outperform comparative state-of-the-art methods, for superpixels generation in terms of boundary recall and under-segmentation

error; and for spectral segmentation in terms of covering, probabilistic random index and volume of information metrics.

Keywords: Superpixels. Stochastic graph contraction. Superpixels graph adjacency. Multi-scale segmentation. Hierarchical representation. Eigen-decomposition.

Métodos Estocásticos para Segmentação Multi-escala de Imagens Naturais

RESUMO

A geração de superpixels é uma das etapas de pré-processamento mais usadas para tarefas de visão computacional, desde a segmentação de imagens até a detecção de saliências e o rastreamento de objetos. O uso de representações de superpixel reduz bastante a informação redundante, criando uma representação condensada da cena na imagem, e também permite maior escalabilidade, reduzindo significativamente a complexidade computacional necessária para executar tarefas de visão computacional. Várias abordagens diferentes de geração de superpixel foram propostas na literatura, e os maiores desafios enfrentados pelas abordagens existentes são: i) pobre aderência as bordas dos objetos na imagem; ii) dificuldade em gerar superpixels estruturados em condições não ideais (por exemplo, variabilidade de ruído, cor e iluminação); iii) pobre definição das bordas entre os objetos na imagem; e iv) dificuldade em obter representações hierárquicas e aninhadas da informação visual onde o superpixel em uma escala grosseira pode ser representado como a união do conjunto de superpixels em escalas mais finas (cobertas pelo superpixel da escala grosseira). Motivados para enfrentar esses principais desafios, esta tese introduz novos métodos para o problema de segmentação de imagens, em relação à tarefa de geração de superpixels em múltiplas escalas, em imagens naturais. Primeiro, é proposto o método Iterative Hierarchical Stochastic Graph Contraction (IHSGC), que utiliza uma estratégia estocástica para gerar superpixels em múltiplas escalas, e cada superpixel é representado por uma estrutura hierárquica e descreve um patch de imagem em escalas fina e grosseira simultaneamente. O método proposto consiste em duas etapas principais: uma etapa de inicialização baseada em uma sobre-segmentação estocástica não supervisionada multi-camada no nível de pixels, preservando as relações espaciais locais na escala mais fina; e uma etapa de contração estocástica, hierárquica e iterativa do grafo, para gerar escalas mais grosseiras por operações de contração de grafos até que o número desejado de superpixels seja obtido. Segundo, é proposta uma versão aprimorada do método IHSGC, chamada stochastic graph contraction (SGC) e a sua versão multi-escala, que utilizam feições simples para gerar escalas mais grosseiras, com melhor aderência às bordas e menor tempo de execução. Finalmente, propõe-se a abordagem Stochastic Spectral Graph Contraction (SSGC) como aplicação dos superpixels estocásticos, para lidar com o problema de escalabilidade da segmentação espectral de imagens, relacionada ao processo

de decomposição espectral. Um grafo de adjacência de superpixels é construído e cada superpixel é descrito com um histograma normalizado 3D do espaço de cor CIELAB. Em seguida, são usadas operações de contração estocástica do grafo dos superpixels e em um subespaço definido pelos K menores autovetores, para obter K regiões da imagem. Os resultados experimentais baseados nos bancos de imagens BSDS300 e BSDS500, sugerem que os métodos estocásticos propostos para geração de superpixels e segmentação espectral obtêm resultados comparáveis ou melhores do que os métodos propostos no estado-da-arte em termos de aderência às bordas e o erro de sub-segmentação, e métricas padrão para algoritmos de segmentação.

Palavras-chave: Superpixels, Contração estocástica de grafos, Grafo de adjacência e superpixels, Segmentação multi-escala, Representação hierárquica, Decomposição espectral..

LIST OF ABBREVIATIONS AND ACRONYMS

BR	Boundary Recall
BSDS300	Berkeley Standard Dataset - 300 images
BSDS500	Berkeley Standard Dataset - 500 images
CAPES	Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
CIELAB	Commission Internationale de l'Eclairage (CIE), the L*a*b* color space
COV	Segmentation Covering
DBSCAN	Density Based Spatial Clustering of Applications with Noise
ERS	Entropy Rate Superpixel Segmentation
FSLIC	Fuzzy Simple Linear Iterative Clustering
HSGC	Hierarchical Stochastic Graph Contraction
I2MTC	International Instrumentation and Measurement technology Conference
IHSGC	Iterative Hierarchical Stochastic Graph Contraction
ISFFCM	Improved Superpixel-based Fast Fuzzy C-Means Clustering
LSC	Linear Spectral Clustering
MSHIFT	Mean Shift algorithm
MST	Minimum Spanning Tree
NCUT	Normalized Cut
PRI	Probabilistic Random Index
SAG	Superpixel Adjacency graph
SC	Spectral Clustering
SCoW	Spatial Constrained Watersheds
SEEDS	Superpixels Extracted via Energy-Driven Sampling
SFFCM	Superpixel-based Fast Fuzzy C-Means Clustering
SGC	Stochastic Graph Contraction

SH	Superpixels Hierarchy
SLIC	Simple Linear Iterative Clustering
SNIC	Simple Non-Iterative Clustering
SRM	Stochastic Region Merging
SSGC	Stochastic Spectral Graph Contraction
$SSGC_E$	Efficient Stochastic Spectral Graph Contraction
TURBO	Turbo Pixels
UE	Under-segmentation Error
USA	United State of America
VoI	Volume of Information
WSHED	Watershed algorithm

LIST OF FIGURES

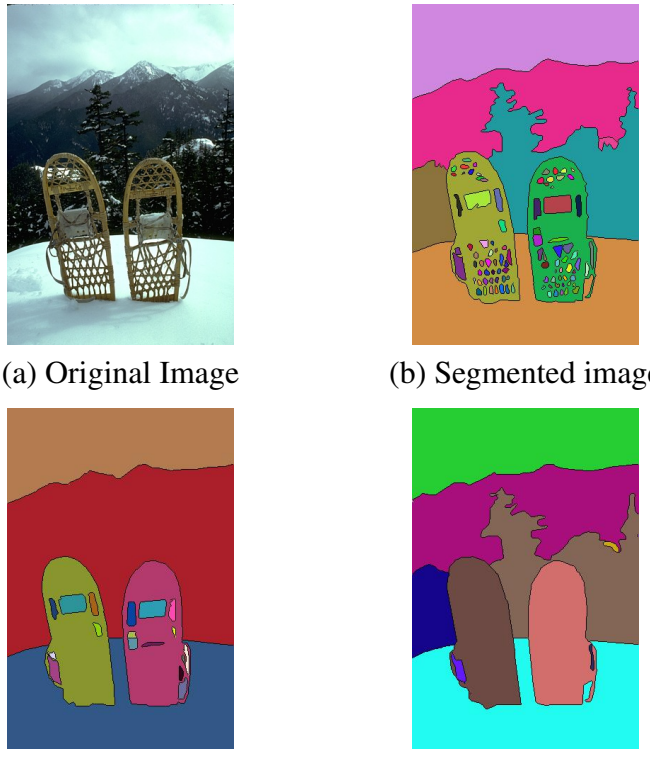
LIST OF TABLES

CONTENTS

1 INTRODUCTION

Image segmentation is the task of partitioning a digital image into multiple segments, that represent sets of pixels, and by assigning distinct labels to each segment, also known as image objects. The goal of segmentation are to decompose the image into segments for further analysis, and change the representation of the image into something that is more meaningful (SHAPIRO; STOCKMAN, 2001). Also, image segmentation is a fundamental task for many applications in computer vision, such as medical image segmentation (DAOUD et al., 2019), saliency detection (WANG et al., 2015; NAVA; KYBIC, 2015), (JIN; LI; LI, 2019), (ZHANG et al., 2019), object recognition (LI et al., 2015), (GHADIRI; BERGEVIN; BILODEAU, 2019) and visual pattern classification (WANG et al., 2013), (YANG et al., 2019). Fig. 1.1 shows an example of natural image in the Fig. 1.1(a) and the three human segmentations of this natural image in the Fig. 1.1(b) to Fig. 1.1(d).

Figure 1.1: Example of image segmentation. Example of natural image in the (a) and the three human segmentations of this natural image in (b) to (d)



(a) Original Image (b) Segmented image

(c) Segmented image (d) Segmented image

Source: The Author.

Within this context, *superpixels* have been used to over-segment images by grouping pixels that have similar properties, such as intensity, color or texture, while reduc-

ing redundant information and the computational complexity of the overall segmentation process. Besides, superpixels potentially can enable greater scalability by reducing the computational complexity required to perform computer vision tasks.

The segmentation of natural images often faces real challenging, such as noise, color and illumination variability, and in several applications the image over-segmentation based on superpixels is crucial to represent conveniently the image regions, specially when dealing with high resolution images due to scalability problem. Therefore, the superpixels generation process from natural images has four of the biggest challenges faced by such existing approaches: i) poor adherence to the object boundaries, and ii) difficulty in generating well-structured superpixels under non-ideal scene conditions (e.g., noise, color and illumination variability), iii) weak boundary separation between objects, and iv) difficulty to obtain hierarchical and nested image representations where a multi-scale superpixel at a coarser scale can be represented as the union of the set of finer scales superpixels covered by the coarser scale superpixel.

Boundary adherence is a crucial property, which the segments obtained by an algorithm preserve the images edges (i.e the boundaries between objects in the image). So, due to the issues in natural images, such as noise, color and illumination variations, the boundary adherence to the objects in the image is already a challenge, because this property has a negative impact in the result of subsequent tasks (e.g. segmentation, classification) if the initialization step with superpixels is not suitable (VASQUEZ; SCHARCANSKI; WONG, 2015b). Therefore, superpixels generation process and its use as initialization step to other more complex tasks is already an open problem and it is the focus of this thesis.

Superpixel methods tend to provide over-segmentation results with desirable properties, such as: 1) each pixel is assigned to only one superpixel; 2) all pixels contained within a superpixel have similar features; and 3) the number of superpixels generated to over-segment an image is controllable. However, other important feature to add to superpixels generation methods is the ability to represent the input image in multiple scales, preserving the spatial relationships among local image regions in the hierarchical structure (VASQUEZ; SCHARCANSKI, 2018). So, it would be desirable to obtain hierarchical and nested image representations where a multi-scale superpixel at a coarser scale can be represented by the union of the set of superpixels at finer scales, and covered by one superpixel at coarser scale. Such hierarchical superpixel representation would enable to access rich visual information in the image, while allowing to approach more complex

pattern recognition process, and such issues are addressed in this work.

Currently, there are several methods for superpixels generation, because this is a crucial step for many other tasks. The Fig. 1.2 shows examples of results of six superpixels generation methods (1.2 (a) ERS (Entropy Rate Superpixels) (LIU et al., 2011), 1.2 (b) LSC (Linear spectral clustering) (LI; CHEN, 2015), 1.2 (c) MST (Minimum Spanning Tree), 1.2 (d) SNIC (Simple Non-iterative Clustering) (ACHANTA; SUSSTRUNK, 2017), 1.2 (e) TURBO (Turbopixels) (LEVINSHTEIN et al., 2009), 1.2 (f) SEEDS (Superpixels Extracted via Energy-Driven Sampling) (BERGH et al., 2015); in this figure is shown the results of each approach, with superpixels with regular and irregulars shapes, superpixels with poor or better boundary adherence to the objects in the image, and superpixels with approximate number of segments to the desired number of superpixels.

Figure 1.2: Example of results for six superpixels generation methods. (a) ERS (Entropy Rate Superpixels) (LIU et al., 2011), (b) LSC (Linear spectral clustering) (LI; CHEN, 2015), (c) MST (Minimum Spanning Tree), (d) SNIC (Simple Non-iterative Clustering) (ACHANTA; SUSSTRUNK, 2017), (e) TURBO (Turbopixels) (LEVINSHTEIN et al., 2009), (f) SEEDS (Superpixels Extracted via Energy-Driven Sampling) (BERGH et al., 2015).

(a) ERS (b)LSC (c) MST

(d) SNIC (e) TURBO (f) SEEDS

Source: (ACHANTA; SUSSTRUNK, 2017).

Is important to mention that boundary recall and under-segmentation error are two evaluation measures that often are used to quantify the boundary adherence of superpixels and compare distinct superpixels approaches. Boundary recall is the fraction of ground truth edges that fall within a distance d of at least one superpixel boundary. Under-segmentation error is the fraction of the superpixels that leaks across the ground truth segments boundaries. In addition to the aforementioned properties, superpixels may have regular shape and size, which may reduce the over-segmentation accuracy in terms

of boundary recall (see Section 2.4.1) and under-segmentation error (see Section 2.4.2). This is because the shapes of the objects occurring in natural images (or of their parts) tend to be irregular, and breaking them in regular superpixels tend to have a negative impact on the final image segmentation quality (VASQUEZ; SCHARCANSKI; WONG, 2013; VASQUEZ; SCHARCANSKI; WONG, 2015a)..

Also, multi-scale representation is an important property that can be added to superpixels, since spatial relationships among local image regions often can be well represented by a hierarchical structure as in (WEI et al., 2018). Unfortunately, most popular superpixels methods work with one single scale as in (ACHANTA et al., 2012) and (LI; CHEN, 2015). Then, to generate a multi-scale representation of the image using superpixels with boundaries correspondence between superpixels at different scales is also an open problem.

Image segmentation is essential for many application in computer vision (e.g. object recognition and image classification). However, image segmentation can be challenging when the images are noisy, present color and illumination variability, or have low contrast. This work proposes an automatic method for color image segmentation based on stochastic graph contraction operations in the spectral domain using superpixels as an initialization step. In order to handle to spectral decomposition scalability problem, stochastic superpixels are used as an image pre-processing step. The superpixels are represented in the spectral domain to promote their clustering, and stochastic contraction operations are performed on a superpixels adjacency graph (SAG) to obtain the image segmentation.

1.1 Thesis Goals

To develop superpixels generation and spectral image segmentation methods, based on stochastic approaches to handle the main challenges of the segmentation and superpixels generation process from natural images.

1.2 Contributions

The main contributions of this work include:

- One method to obtain stochastic superpixels at single-scale for representation of the

image with better boundary adherence to the objects boundaries in the image.

- Two methods to obtain Multi-scale stochastic superpixels with hierarchical representation of the image with preservation of the boundaries relationship at the different scales.
- One method for spectral image segmentation based on stochastic graph contraction operations and its efficient approach, as application of the stochastic superpixels to handle the scalability problem.

1.3 Outline

This work is organized as follows:

- Chapter 2 introduces some required definitions like graph, trees, spectral clustering theory and several metrics for evaluation of superpixels and image segmentation results.
- Chapter 3 introduces the state-of-the-art in superpixels generation methods and spectral image segmentation.
- Chapter 4 introduces our proposed methods for stochastic superpixels based on stochastic graph contraction. These proposed methods can obtain single-scale and multi-scale results.
- Chapter 5 shows our proposed method for the spectral image segmentation using the superpixels results as initialization step and a stochastic strategy to contract the superpixels adjacency graph using eigenvectors as features.
- Chapter 6 shows the experimental results of our proposed stochastic superpixels and stochastic spectral image segmentation approaches.
- Chapter 7 describes the principal conclusions about this PhD thesis.
- Chapter 8 describes the publications and submitted papers as results of this PhD thesis.

2 FUNDAMENTAL CONCEPTS AND BACKGROUND

This chapter provides some fundamental definitions such as graphs and trees which are presented in Sections 2.1 and 2.2, respectively, since these concepts will be relevant to describe the image representation and the multi-scale superpixels organization. Also, we describe the spectral clustering method used as application of our stochastic superpixels approach. Finally, this chapter describes the standard metrics to evaluate superpixels and segmentation results.

2.1 Graphs

A ***graph*** is a pair $G = (V, E)$ that is constituted by a set V of N vertices and a set E of M edges, which are denoted as follows (DIESTEL, 2000):

$$V = \{v_1, v_2, v_3, ..., v_N\}, \tag{2.1}$$

$$E = \{e_1, e_2, e_3, ...e_M\}, \tag{2.2}$$

and each edge e_m belonging to the set E has two associated vertices v_i and v_j as follows:

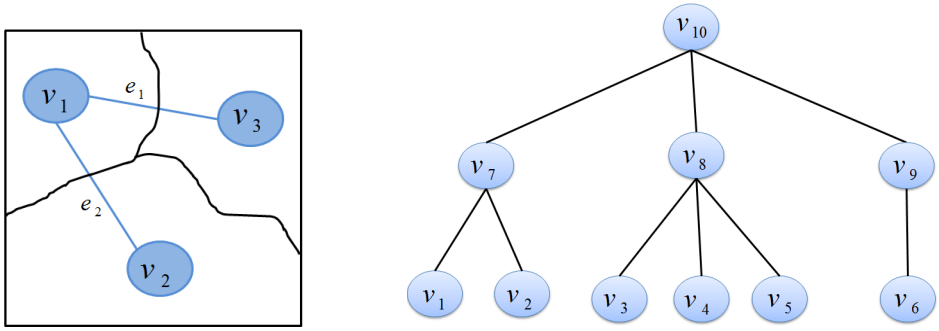
$$e_m(v_i, v_j) \equiv e_m, \tag{2.3}$$

where $e_m(v_i, v_j)$ represents the relationship between vertices v_i and v_j. A graph G is connected if there is a path from any vertex v_i to any vertex v_j in the graph. The graph is disconnected if there exists one or more isolated vertices. In this work, we use undirected graphs (i.e. we ignore the edges directions), so $(v_i, v_j) = (v_j, v_i)$. A path in a graph G is a finite sequence of edges e, which joins a sequence of vertices v.

A ***weighted graph*** is a graph G where each edge has a value and is denoted by $G = (V, E, W)$, where W is the set of weights $W = \{w_1, w_2, w_3, ..., w_M\}$, and w_m represents the weight of an edge e_m in E.

Graphically, we can illustrate vertices by points, and edges by lines or arcs connecting such vertices pairs, as illustrated in Fig. 2.1(a), where a set of vertices $V = \{v_1, v_2, v_3\}$ and a set of edges $E = \{e_1, e_2\}$. As mentioned above, each edge connects a pair of vertices (e.g. $e_1(v_1, v_3) \equiv e_1$ and $e_2(v_1, v_2) \equiv e_2$), and the edges weights of the

Figure 2.1: Graph example: (a) weighted connected graph, where a set of vertices $V = \{v_1, v_2, v_3\}$ and a set of edges $E = \{e_1, e_2\}$; and (b) tree, with the set of vertices $V = \{v_1, v_2, ..., v_{10}\}$, and the root vertex $v_r = v_{10}$.



(a) (b)

Source: The Author.

edges e_1 and e_2, are denoted by w_1 and w_2, respectively. An important graph operation used in this work is the ***graph contraction***, and particularly the vertex contraction property is used to generate simplified versions of a graph (SCHARCANSKI, 2011). Vertex contraction is an operation that merges two vertices v_x and v_y of a graph G, and when necessary removes the self-loops and parallel edges created in the process.

Let v_i and v_j be two vertices of a graph $G = (V, E)$, and let G' be the graph obtained by *contracting* the pairs of vertices v_i and v_j of G into a new contracted vertex v_{ij} of G' which becomes adjacent to all vertices that formerly were neighbors of v_i and v_j in G. Formally, G' is a graph $G' = (V', E')$ with a vertex set $V' = \{V - \{v_i, v_j\}\} \cup \{v_{ij}\}$, where v_{ij} is the contracted vertex $v_{ij} \notin V$, and the edges set E' (removed the self-loops and parallel edges) is defined as follows (DIESTEL, 2000):

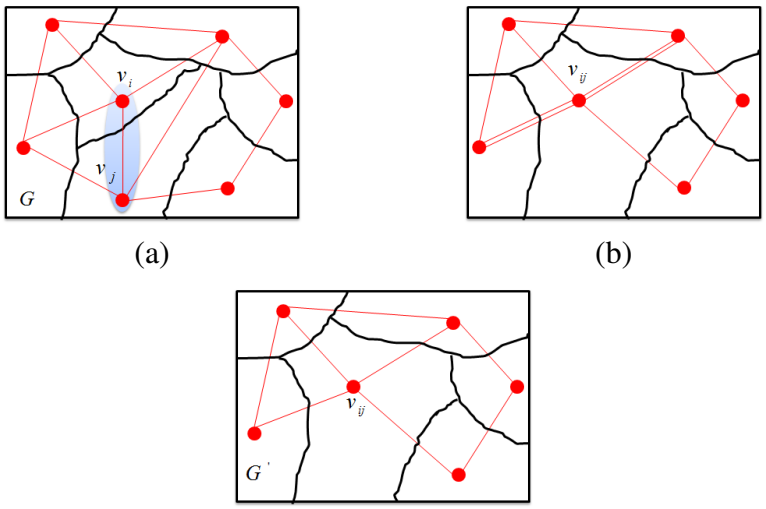
$$\begin{aligned} E' = \{e(v_p, v_q) \in E | \{e(v_p, v_q)\} \cap \{e(v_i, v_j)\} = \emptyset\} \\ \cup \{e(v_{ij}, v_r) | e(v_i, v_r) \in \{E - \{e(v_i, v_j)\}\} \\ \text{or } e(v_j, v_r) \in \{E - \{e(v_i, v_j)\}\}\}, \end{aligned} \tag{2.4}$$

where $e(v_i, v_j)$ is the edge that links the vertices v_i and v_j.

An example of vertex contraction is shown in Fig. 2.2, where the contracted graph G' is obtained by contracting the vertices pairs v_i and v_j of the original graph G. Fig. 2.2(a) shows the original graph G containing the vertices v_i and v_j to be contracted (blue area). The intermediate step of the vertex contraction operation is shown in Fig. 2.2(b), where the contracted vertex v_{ij} represents the vertex contraction result, and the vertices $\{v_i, v_j\}$ and the edge $e_{ij} = (v_i, v_j)$ are removed and do not appear in the graph G'. Addi-

tionally, the parallel edges created by the contraction operation are removed in this intermediate step, obtaining the final graph contraction result which is shown in Fig. 2.2(c), where we can see the updated adjacency graph for the regions in G' after removing the self-loops and parallel edges.

Figure 2.2: Example of vertex contraction v_i and v_j to generate the new vertex v_{ij}. (a) shows the original graph G containing the vertices v_i and v_j to be contracted (blue area). The intermediate step of the vertex contraction operation is shown in (b), where the contracted vertex v_{ij} represents the vertex contraction result, and the vertices $\{v_i, v_j\}$ and the edge $e_{ij} = (v_i, v_j)$ are removed and do not appear in the graph G'. (c) shows the result of graph contraction.



Source: The Author.

2.2 Trees

A ***tree*** is a connected acyclic graph (WEST, 2001), i.e. is a graph with no cycles and no isolated vertices. A tree can be directed or undirected, and is denoted by $T = (V, E)$.

An N-vertex graph $T = (V, E)$ (with $N \geq 1$) is a tree with N vertices if: (a) T is connected and has no cycles; (b) T has $N - 1$ edges; and (c) for each $v_i, v_j \in V$ there is exactly one path between v_i and v_j. Other interesting tree properties can be found in (WEST, 2001).

A ***rooted tree*** is a tree in which a vertex v_r is the *root* (WEST, 2001). For example, let us consider the tree in Fig. 2.1(b) with the set of vertices $V = \{v_1, v_2, ..., v_{10}\}$, and the root vertex $v_r = v_{10}$. For each vertex v_i with $i = [1, 9]$, there is a unique $< v_i, v_r >$-path

$\rho(v_i)$ between v_i and the root v_r. The *parent* of v_i is its neighbor in $\rho(v_i)$, and its *children* are its remaining neighbors not in $\rho(v_i)$. The *ancestors* of v_i are the vertices $\rho(v_i) - v_i$. The *descendants* of v_i are the vertices v_j having $\rho(v_j)$ that also contains v_i. The *leaves* are the vertices with no children. For example, let $v_i = v_7$ be the vertex in analysis and let $v_r = v_{10}$ be the root of the tree in Fig.2.1(b). The $< v_7, v_{10} >$-path is $\rho(v_7) = \{v_7, v_{10}\}$ where the parent of v_7 is the vertex v_{10}, and the children of v_7 are its other neighbors v_1 and v_2. The only ancestor of v_7 is the vertex v_{10}, and the descendants of v_7 are the vertices v_1 and v_2, because $\rho(v_1) = \{v_1, v_7, v_{10}\}$ and $\rho(v_2) = \{v_2, v_7, v_{10}\}$ contain v_7. Finally the leaves are v_j with $j = [1, 6]$.

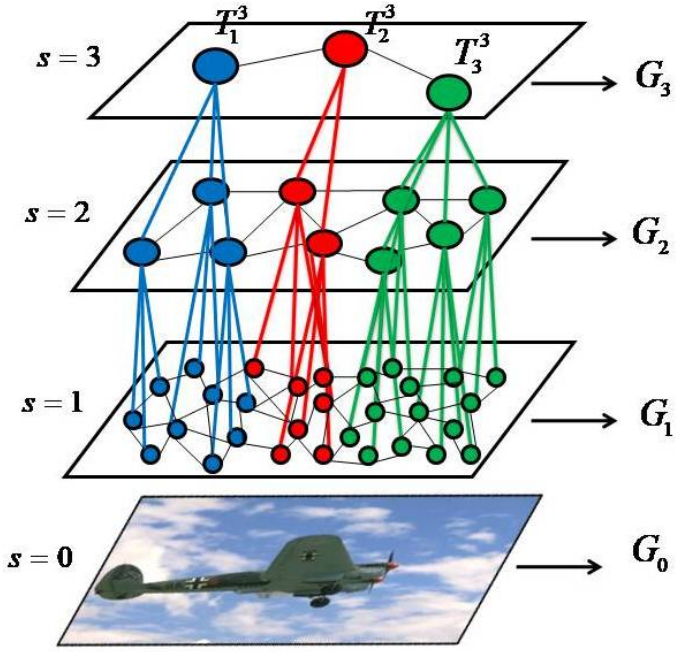
In this work, we use the ***hierarchical tree model*** to represent the relationship between adjacent superpixels (T_i^s) at finer and coarser scales (s). This hierarchical tree model is obtained using the proposed iterative hierarchical tree building process illustrated in Fig. 2.3. At the end of this process, the obtained hierarchical trees (T) describe the multi-scale superpixels, and can be used to represent the image contents in multiple scales, as detailed next.

Let s be a scale of the image representation with $s = 0, 1, 2, ..., S$, where $s = 0$ denotes the image pixels level. For each scale $s = k$, with $k > 0$, each superpixel i (T_i^k) (which within this framework is a multi-scale superpixel i at scale k), is a root of a rooted tree and can represent a rooted tree itself if $s > 1$. Each superpixel T_i^s at scale s is part of an adjacency graph of multi-scale superpixels (G_s) at scale s.

Fig. 2.3 shows an illustration of the contents of an image represented in terms of the proposed multi-scale superpixels, and also illustrates the adjacency graph of superpixels for each scale s (G_s). Each superpixel (T_i^s) in fact is a rooted tree itself, that represents the relationship among superpixels located at different scales $s > 0$ and the pixels at the finest scale $(s = 0)$. Therefore, each scale s ($s > 0$) has N_s multi-scale superpixels, and each multi-scale superpixel is associated with a hierarchical tree T_i^s with $i = 1, 2, ..., N_s$. Each tree at each scale s (T_i^s) (in blue, red and green colors) corresponds to one multi-scale superpixels at scale s, which inherits the same hierarchical structure of the multi-scale superpixels at finer scales $s' < s$, allowing to represent the image contents in different levels of detail. In this work, we use indistinctively the terms superpixels and multi-scale superpixels, since each superpixel in fact is a multi-scale superpixel, and vice-versa.

A graph of multi-scale superpixels at scale s is denoted by $G_s = (V_s, E_s)$, where the G_s vertices are multi-scale superpixels $V_s = \{T_1^s, T_2^s, T_3^s, ..., T_{N_s}^s\}$, and each multi-

Figure 2.3: Illustration of the multi-scale representation of an image by multi-scale superpixels based on the hierarchical tree model. It represents the relationship between adjacent superpixels (T_i^s) at finer and coarser scales (s).



Source: The Author.

scale superpixel T_i^s represents the local image contents since each multi-scale superpixel at scale s with $s > 0$ represents a region of the original image I (i.e. a set of pixels of I), allowing to represent I at the scale s by the union of multi-scale superpixels T_i^s as follows:

$$I = \bigcup_{i=1}^{N_s} T_i^s \text{ with } T_i^s \cap T_{i'}^s = \emptyset,\ \forall i \neq i'. \tag{2.5}$$

The hierarchical relationship between a parent superpixel $T_j^{s_2}$ and its children superpixels $T_i^{s_1}$ at two different scales s_1 and s_2, where s_2 is coarser than s_1 and $N_{s_2} < N_{s_1}$, is described as follows :

$$T_j^{s_2} = \bigcup_{i=1}^{N_{s_1}^i} T_i^{s_1} \text{ with } T_i^{s_1} \cap T_{i'}^{s_1} = \emptyset,\ \forall i \neq i', \tag{2.6}$$

where $N_{s_1}^i$ is the number of superpixels that are child of $T_j^{s_2}$ at scale s_1, $\bigcup_{i=1}^{N_{s_1}^i} T_i^{s_1}$ is the union of the superpixels $T_i^{s_1}$ that are child of $T_j^{s_2}$ at scale s_1. The proposed hierarchical relationship among superpixels can be described by Eqs. (2.5) and (2.6), which provides a hierarchical structure that can be used for representing the image contents at different scales (i.e., at finer and coarser scales).

Fig. 2.4 illustrates an example of multi-scale superpixels, where three natural images from the BSDS500 dataset (ARBELAEZ et al., 2011) are used for demonstration purposes. Left column Fig. 2.4(a), shows the original images; middle column Fig. 2.4(b), shows a finer scale with 1000 superpixels with their boundaries in red color overlaid on the original image; and Fig. 2.4(c) right column, shows a coarser scale with 100 superpixels with their boundaries in green color overlaid on the original image. In these examples, the corresponding boundaries of the superpixels at the finer and coarser scales have correspondence, and the finer scales superpixels are contained within the coarser scales superpixels.

The relationship among finer and coarser superpixels are organized in a multi-scale hierarchical structure, where the superpixels provide an image representation with different levels of detail, that could be useful for many computer vision and pattern recognition applications.

Figure 2.4: Example of multi-scale superpixels for three natural images of the BSDS500 dataset. Left column (a), shows the original images; middle column (b), shows a finer scale with 1000 superpixels with their boundaries in red color overlaid on the original image; and (c) right column, shows a coarser scale with 100 superpixels with their boundaries in green color overlaid on the original image.

(a) (b) (c)

Source: The Author.

2.3 Spectral Clustering

Spectral methods work with the similarity (or distance) matrix of the input data, instead of working with the original data points in their original dimensions. Let $\{x_1, ..., x_N\}$ be the original data points (e.g. nodes of a graph), represented by the matrix X, and the similarity between data points x_i and x_j be w_{ij}, which is an element of a $N \times N$ similarity matrix W. There are different ways to evaluate the similarity between a pair of data points, and this issue will be detailed later. Furthermore, the value of w_{ij} can be assumed 0 if the similarity between x_i and x_j is smaller than a threshold, making W sparse by connecting only the nearest neighbors to any data point x_i. Spectral methods can embed the data points in X into a different space, which could have a lower dimensionality than the original data, also performing dimensionality reduction in this case. For multidimensional data X, as N increases, the large similarity matrix W may be redundant, unless the data is noisy and high dimensional (AGGARWAL; REDDY, 2014).

The similarity matrix W may also be viewed as the nodes adjacency matrix of a graph $G = (V, E, W)$, in which each graph node/vertex v_i corresponds to a data point x_i, and the weight of an edge $e_{ij} = (v_i, v_j)$ corresponds to the similarity between the vertices v_i and v_j, that in fact are representing the data points x_i and x_j. In this case, spectral methods can be seen as finding optimal cuts in this graph $G = (V, E, W)$, which correspond to graph partitions which are weakly connected by edges representing similarity. Consequently, a spectral method can be considered a graph-based clustering method for data points by representing a network structure (i.e. a graph) in the similarity matrix. Given a symmetric similarity measure and a data set X, the similarity matrix W is symmetric, exhaustive, may be noisy and encode a significant amount of information about the graph. Spectral analysis uses the eigenvector-analysis of this matrix W to abstract noise and artifacts while recovering and sharpening the latent information in the similarity matrix, though at a rather high cost as the number of data points increases, as will de detailed later.

In order to understand how spectral clustering works, let us discuss initially the problem of mapping the points x_i onto a 1-dimensional space where data clustering is performed. Afterwards, the k-dimensional case will be discussed. Therefore, we wish to map the data points in $X = \{x_1, ..., x_N\}$ into a set of points $Y = \{y_1..., y_N\}$ lying on a line, in a way that similar points are mapped to closer locations on this line, and more dissimilar points are mapped onto distant points on this line. To meet the criteria above

and determine values of y_i, the following cost function C could be minimized:

$$min(C) = argmin_Y\{\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}(y_i - y_j)^2\}, \tag{2.7}$$

indicating that C is minimized similar points (i.e. those points with larger similarities w_{ij}) have the smaller distances $(y_i - y_j)$ (i.e. are mapped closer to each other).

The cost function C can be rewritten in terms of the Laplacian matrix L of W, which is given by $L = D - W$, where D is a diagonal matrix satisfying $D_{ii} = \sum_{j=1}^{N} w_{ij}$, as follows:

$$min(C) = argmin_Y\{2Y^T LY\}. \tag{2.8}$$

The trivial solution to Eq. 2.8, $Y = \{y_1..., y_N\} = 0$, is not informative, and a scaling constraint such as $Y^T DY = 1$ could be imposed to ensure that the trivial value of $y_i = 0$ is not chosen as a solution. It shall be observed that different weights w_{ij} are applied to the data points (or graph vertices), and graph vertices v_i with greater similarity to other vertices tend to be more relevant in the minimization $min(C)$ of Eq. 2.8, suggesting that there may be dominant sets of graph vertices/data points in the clustering process. The optimization in Eq. 2.8 is in generalized eigenvector format, and the solution is the eigenvector u_{min} for which the generalized eigenvectors relationship $Lu = \lambda Du$ is satisfied, and u_{min} is the eigenvector with the smallest eigenvalue. However, the smallest generalized eigenvalue corresponds to the non-informative trivial solution, where $u = 1$ is the normalized unit vector, and it is not used in the analysis. The second-smallest eigenvalue then provides a best solution, which is more informative. The other $N - 2$ eigenvectors in ascending order of eigenvalue also provide solutions to the optimization in Eq. 2.8, and correspond to successive orthogonal directions in the data X structure, and this set of orthogonal n directions result in the best mapping for X into Y. This results in a set of n eigenvectors $U = \{u_1, u_2, ..., u_N\}$ (of which the first eigenvector u_1 is trivial and irrelevant to the problem solution), and their corresponding eigenvalues are $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_N$.

The corresponding vector representation of the data points $X = \{x_1, ..., x_N\}$ along each eigenvector $U = \{u_1, u_2, ..., u_N\}$ is denoted by $Y = \{y_1..., y_N\}$. If the data items X are ordered along a small magnitude eigenvector u_i, and since $Lu_i = \lambda_i Du_i$ and $\lambda_i Du_i$ is minimized for small λ_i since $||u_i|| = 1$, a graph cut is obtained and the weight of the edges across the cut are likely to be small. This means that this is a graph cut, and each partition

is a cluster in the space of data items (i.e. graph vertices). If just one small magnitude eigenvector e_i is used, then $\{y_1..., y_N\}$ are the corresponding projections of $\{x_1, ..., x_N\}$ on a line oriented along u_i. It shall be observed that if the top (i.e. smaller) K eigenvectors generate a $N \times K$ matrix containing the vector representations $\{y_N, ..., y_{N-K+1}\}$ for the data items, a K-dimensional embedded representation is obtained of the entire data set of N points, which preserves optimally the maximum amount of information, from the minimum squares point of view (AGGARWAL; REDDY, 2014). Consequently, spectral methods can be used in order to simultaneously perform clustering and dimensionality reduction.

Therefore, if each data point x_i is projected into the space spanned by the k eigenvectors of the Laplacian L, the data point x_i is represented by a vector y_i with dimensionality $1 \times K$ as a row of the $N \times K$ matrix containing the eigenvectors of the Laplacian L as columns. For any of these vectors y_i (HASTIE; TIBSHIRANI; FRIEDMAN, 2009):

$$Y^T LY = \sum_{i=1}^{N} d_i y_i^2 - \sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j w_{ij} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(y_i - y_j)^2 w_{ij}, \tag{2.9}$$

indicating that vertices (v_i, v_j) (or data points (x_i, x_j)) will cluster together. This occurs because smaller values of $Y^T LY$ will be obtained when pairs of vertices (v_i, v_j) (or data points (x_i, x_j)) with larger similarities w_{ij} have coordinates y_i and y_j close to each other in the space spanned by the k smaller eigenvectors $\{u_N, ..., u_{N-K+1}\}$ of the Laplacian L.

Using L the cost function in Eq. 2.8 is minimized when the number of vertices in each cluster is minimized, but this may lead to small clusters and uninteresting solutions. However, by normalizing the Laplacian more interesting solutions can be found, since the equation for the normalized Laplacian would be minimized for the volume of each cluster, as discussed next. Besides, the un-normalized Laplacian works well for the regular graphs, but the normalized Laplacian works well for regular and also for irregular graphs[1] (LUXBURG, 2007). Therefore, it has been proposed to normalize the Laplacian with respect to the graph vertices degrees d_{ii}, for example, $L_{rw} = D^{-1}L = I - D^{-1}W$. When the graph Laplacian is normalized, it is possible to interpret the spectral clustering method as a random walk on the graph with a transition probability matrix $P = (p_{ij}) = D^{-1}W$, for $i, j = 1, ..., n$ (HASTIE; TIBSHIRANI; FRIEDMAN, 2009). It shall be observed that the normalized Laplacian can be written in terms of the transition probability matrix as $L_{rw} = I - P$. From this random walk perspective, spectral clustering groups graph

[1]Every vertex of a regular graph has the same degree. Otherwise, the graph is irregular.

vertices in a way that the random walk rarely would transition from one group of vertices to another, and this can be done by choosing the most likely random walk steps on the graph. Therefore, the transition probability of the random walk jumping in one step from vertex v_i to vertex v_j is proportional to the edge weight w_{ij} and is given by $p_{ij} = \frac{w_{ij}}{d_{ii}}$. If the graph is connected and non-bipartite, then the random walk always possesses a unique stationary distribution $P = (p_{11}, ..., p_{NN})$, which is given by $p_{ii} = \frac{d_{ii}}{vol(G)}$, where $vol(G) = \sum_{i=1}^{n} d_{ii}$ (LUXBURG, 2007). We will return to this random walk interpretation when the proposed method is detailed.

In fact, the normalized Laplacian L_{rw} relates to the eigenvalue problem with L and D discussed before :

$$\begin{aligned} &LU = \lambda DU = D^{-1}LU = \lambda U \\ &(D^{-1}D - D^{-1}W)U = \lambda U \\ &(I - P)U = \lambda U \\ &L_{rw}U = \lambda U; \end{aligned} \tag{2.10}$$

where the matrix U has the eigenvectors u_i as columns, with $i = 1, \ldots, N$. As can be seen above, the standard eigenvalue problem with L_{rw} yields the same results as the generalized eigenvalue problem with L and D, and the discussion above also applies to the normalized Laplacian L_{rw}.

In general, spectral clustering tends to provide high-quality results. However, spectral clustering works with an $N \times N$ similarity matrix W, and its overall complexity of $O(N^3)$ in general (YAN; HUANG; JORDAN, 2009) may become prohibitive as the number of data points N increases. For example, in image processing the number of image pixels N can be quite large. In this context, working with superpixels instead of pixels is attractive, since the number of superpixels in an image usually is much smaller than the number of pixels, as will be discussed next.

Our proposed approach builds on the normalized spectral clustering method (SHI; MALIK, 2000), which is detailed in Algorithm 1.

In Algorithm 1, each pixel i of the input image I can be interpreted as a graph vertex v_i, $1 \leq i \leq N$. From this perspective, Algorithm 1 partitions a graph into K sub-graphs containing a set of similar vertices. Consequently, the vertex v_i is assigned to the sub-graph ϕ_j if the vertex represented by the $i-$th row of U' is assigned to the sub-graph ϕ_j in the graph partition $\phi = \phi_1, \phi_2, ..., \phi_K$. The edge weights are defined according to the application, for example similarity distances between color, textures, region descriptor

Algorithm 1: Normalized spectral clustering algorithm for image segmentation (SHI; MALIK, 2000).

Input : The input image I, the similarity matrix W, and the number K of desired clusters.

Output: K clusters of image pixels corresponding to the K distinct image regions.

1 Find the $N \times N$ matrix U containing along its columns the solutions u_i of the eigenvector problem $L_{rw}U = \lambda U$. Sort the column eigenvectors u_i by their associated eigenvalues λ_i, for $i = 1, \ldots, N$. The $i-$th row of U is the $1 \times N$ representation of the pixel i of the input image I in the space spanned by N column eigenvectors of U;

2 Select the sub-space spanned by the K column eigenvectors u_j of U having the K smaller eigenvalues λ_j, for $j = 1, \ldots, K$. These K smaller eigenvectors u_j are the columns of the $K \times N$ matrix $U' \subset U$. Now, the $i-$th row of U' is the $1 \times K$ representation of the pixel i of the input image I in the space spanned by K smallest column eigenvectors of U;

3 Cluster using K-means the $1 \times K$ rows of U' to find K clusters of pixels $\phi = \phi_1, \phi_2, ..., \phi_K$;

4 Return K pixel clusters.

or the join of them, and usually is expressed like to a Gaussian distribution to obtain a similarity values between 0 to 1.

2.4 Superpixels Quantitative Evaluation Criteria

Superpixels are groups of pixels sharing similar properties such as color or texture. In order to evaluate the generation of superpixels by the proposed methods (see Section 4), and compare it with other state-of-the-art approaches, we use two standard error metrics, namely, *boundary recall* and *under-segmentation error* (NEUBERT; PROTZEL, 2012). Boundary recall is an error metric widely used to evaluate image segmentation methods, but it tends to provide a limited evaluation of the superpixels over-segmentation. Therefore the under-segmentation error metric also is used in our experimental evaluation of the superpixels methods to compensate for the above mentioned limitation of the boundary recall metric. Next, we detail both metrics.

2.4.1 Boundary Recall (BR)

Boundary recall is the fraction of ground truth boundary pixels that fall within a certain distance d of at least one superpixel boundary location (as suggested in (NEUBERT; PROTZEL, 2012), $d = 2$) defines the boundary recall. Given a ground truth boundary image GT and the compared boundary image B, the boundary recall computation proceeds as follows:

$$BR = \frac{TP}{TP + FN}, \tag{2.11}$$

where TP is the number of boundary pixels in GT if exists a B boundary pixel within the d range, and FN is the number of boundary pixels in GT if does not exist a boundary pixel in B within the d range.

2.4.2 Under-segmentation Error (UE)

Considering that a ground truth segment divides a superpixel P into in and out parts, the under-segmentation error compares the segments areas and measures to what extent the superpixels flood over the ground truth segment borders (NEUBERT; PROTZEL, 2012), which is estimated as follows:

$$UE = \frac{1}{N}\left[\sum_{S \in GT}\left(\sum_{P:P \cap S \neq \emptyset} min(|P_{in}|, |P_{out}|)\right)\right], \tag{2.12}$$

where N is the number of pixels in the original image, S is a segment of the ground truth GT ($S \in GT$), P is a superpixel, and P_{in} and P_{out} are the superpixels parts that are inside of the segment S and outside of the segment S, respectively. The $|P_{in}|$ and $|P_{out}|$ are the number of pixels contained in each set.

2.5 Segmentation Quantitative Evaluation Criteria

The goal of image segmentation is to partition the original image into a set of regions, and it is desirable that image segments have good boundaries adherence to the boundaries of the objects in the image. In order to evaluate the segmentation results obtained by the proposed method (see Section 5), and compare them with the results ob-

tained by other state-of-the-art approaches, we use three known error measures, namely, segmentation covering (COV), probabilistic rand index (PRI) and Variation of Information (VoI) (ARBELAEZ et al., 2011).

2.5.1 Segmentation Covering (COV)

Segmentation covering of a segmentation S by a segmentation S' is defined as (MAIRE, 2009):

$$C(S' \rightarrow S) = \frac{1}{N} \sum_{R \in S} |R| * \max_{R' \in S'} O(R, R'), \tag{2.13}$$

where N denotes the total number of pixel in the image, and $O(R, R')$ is the overlap between pairs of regions R and R' and is defined as:

$$O(R, R') = \frac{|R \cap R'|}{|R \cup R'|}. \tag{2.14}$$

2.5.2 Probabilistic Rand Index (PRI)

The Probabilistic Rand Index (PRI) counts the fraction of pixels which labels are consistent comparing the computed segmentation and the ground truth, averaging across multiple ground truth segmentations to account for scale variation in human perception (YANG et al., 2008).

PRI is defined as:

$$PRI(S, G_r) = \frac{1}{T} \sum_{i<j} \left[c_{ij} p_{ij} + (1 - c_{ij})(1 - p_{ij}) \right], \tag{2.15}$$

where T is the total number of pixel pairs considered, p_{ij} is the probability of this event occurring, and c_{ij} is the event that pixels i and j have the same label (MAIRE, 2009).

Considering that c_{ij} provides information about the segmentation in form of binary numbers $c_{ij} = \mathbb{I}(l_i^S = l_j^S)$ for each pairs of pixels (x_i, x_j), where the label of point x_i is represented by l_i^S in segmentation S (UNNIKRISHNAN; PANTOFARU; HEBERT, 2005).

Given the manually labeled images, we can compute the empirical probability of the label relationship of a pixel pair x_i and xj simply as (UNNIKRISHNAN; PANTO-

FARU; HEBERT, 2005):

$$p_{ij}(l_i^S == l_j^S) = \frac{1}{K}\sum_{k=1}^{K}\mathbb{I}\left[l_i^k == l_j^k\right], \tag{2.16}$$

and

$$p_{ij}(l_i^S \neq l_j^S) = \frac{1}{K}\sum_{k=1}^{K}\mathbb{I}\left[l_i^k \neq l_j^k\right] = 1 - p_{ij}(l_i^S == l_j^S), \tag{2.17}$$

This measure takes values in $[0, 1]$, 0 when S and $\{S_1, S_2, ..., S_K\}$ have no similarities (i.e. when S consists of a single cluster and each segmentation in $\{S_1, S_2, ..., S_K\}$ consists of clusters containing single points, or vice versa) to 1 when all segmentations are identical (UNNIKRISHNAN; PANTOFARU; HEBERT, 2005).

2.5.3 Variation of Information (VoI)

The variation of information measure evaluates the distance between two segmentations by the average conditional entropy of one segmentation given the other, and thus roughly measures the amount of randomness in one segmentation which can not be explained by the other (YANG et al., 2008):

$$VoI(S, S') = H(S) + H(S') - 2R(S, S'), \tag{2.18}$$

where H and R represent respectively the entropies and mutual information between two segmentation results S and S' (MAIRE, 2009).

3 STATE-OF-THE-ART: SUPERPIXELS AND IMAGE SEGMENTATION

Superpixels generation is one of the most widely used pre-processing step for a computer vision task, ranging from image segmentation (LEI et al., 2018), saliency detection (YIFENG et al., 2020), classification (AVELAR et al., 2020) and object tracking (CHAN; ZHOU; CHEN, 2018).

The use of superpixels representations reduce greatly informational redundancy by creating a condensed representation of the scene, and it also enables greater scalability by significantly reducing the computational complexity required to perform computer vision task.

The superpixels generation methods can be grouped in two broad categories: i) *graph-based algorithms*, and ii) *gradient-ascent-based algorithms* (ACHANTA et al., 2012). Usually, the *graph-based algorithms* use a graph based structure to represent the spatial relationship among pixels. Some examples of the *graph-based algorithms* are: 1) normalized cuts algorithm (NCUTS) (SHI; MALIK, 2000), which applies the normalized cut recursively on a graph of image pixels; 2) minimum spanning tree methods (MST) (FELZENSZWALB; HUTTENLOCHER, 2004), that usually rely on the Kruskal algorithm and some heuristic to estimate the pixels similarities, and then divide the image in trees, with each tree representing a region; 3) ERS algorithm (LIU et al., 2011) that tries to maximize the entropy rate to determine which subgraphs of the image should be grouped/merged, and so on. Often, the *gradient-ascent-based algorithms* initially obtain a rough clustering of the image pixels, and then refine the obtained pixel clusters iteratively until some convergence criterion is met.

Some examples of *gradient-ascent-based algorithms* are: 1) mean shift algorithm (COMANICIU; MEER, 2002), which uses color and spatial features to search iteratively for the local maxima of an underlying density function; 2) watershed algorithm (VINCENT; SOILLE, 1991), which provides an alternative to obtain the initial image over-segmentation; 3) turbopixels (LEVINSHTEIN et al., 2009) and the SEEDS algorithm (BERGH et al., 2015), that can be used to obtain the initial superpixels partitioning of an image, and then refine the superpixels by modifying their boundaries while enforcing some superpixels color homogeneity criteria; 4) spatial-constrained watersheds (SCoW) (HU; ZOU; LI, 2015), that rely on watersheds with a set of evenly placed markers to cluster pixels and form superpixels; and 5) DBSCAN (SHEN et al., 2015), which evaluates the local pixel color similarities and geometric constraints to obtain the initial

set of superpixels.

Among the clustering methods performing some type of *gradient-ascent*, the Simple Linear Iterative Clustering (SLIC) method (ACHANTA et al., 2012) is popular for superpixels generation, and uses local pixels k-means clustering based on color and spatial pixels distances to evaluate the similarity among pixels, and obtain the clusters centroids. The SLIC method is simple, but often has problems to obtain good boundary adherence when the desired number of superpixels is small. Also, it shall be observed that SLIC was not designed to obtain an exact number of superpixels or an image representation based on multi-scale superpixels (i.e. a representation that preserves superpixels boundary correspondences at different scales). Some improved versions of the SLIC algorithm have been proposed, such as the Simple Non-Iterative Clustering (SNIC) (ACHANTA; SUSSTRUNK, 2017), that is an non-iterative method which enforces connectivity from the start; the Linear Spectral Clustering algorithm (LSC) (LI; CHEN, 2015), that can provide improved superpixels by using a ten-dimensional space to run a weighted K-means; and the FSLIC (WU; ZHANG L .AND ZHANG; YAN, 2019), that is a noise-robust superpixel method. These improved versions of SLIC still do not provide good boundary adherence to the image objects, specially for natural images that usually show noise, color and illumination variability, and/or weak boundary separation between image objects. Besides, these superpixels generation methods can not obtain the desired number of superpixels or multi-scale superpixels image representations. There are others interesting superpixels generation methods available, such as the Bayesian Adaptive Superpixel Segmentation (BASS) (UZIEL; RONEN; FREIFELD, 2019) and the Superpixel Hierarchy (SH) (WEI et al., 2018). In particular, BASS generates superpixels adaptively using an initial number of superpixels, that is refined until the desired number of superpixels (N_d) is reached. The BASS method is computationally more complex than most methods, it can run in parallel in $2s$ on a GPU, but its results are still less competitive in comparison with other algorithms that run in $1s$ with better results that do not use a dedicated hardware. The SH approach is faster and can generate multi-scale superpixels based on the Borůvka algorithm relying on the integration of color histograms and other image features, but it can not control some relevant superpixels characteristic like their size. Also, recently a CNN-based superpixel method was proposed in (TEPPEI, 2020) that generates superpixels via convolutional neural networks (CNN) by minimizing a proposed objective function, but it does not have control over the number of superpixels and its results only outperform slightly the SLIC method, in terms of boundary recall, when the number of

superpixels is small.

Table 3.1: Comparison of the properties of different superpixels generation methods.

Method	P1	P2	P3	P4	P5
Proposed	✓	✓	✓	✓	✓
NCUT (SHI; MALIK, 2000)	✓	-	-	-	-
MST (FELZENSZWALB; HUTTENLOCHER, 2004)	✓	-	-	-	✓
ERS (LIU et al., 2011)	✓	-	✓	-	✓
WSHED (VINCENT; SOILLE, 1991)	-	-	-	-	✓
MSHIFT (COMANICIU; MEER, 2002)	-	-	-	-	✓
SEEDS (BERGH et al., 2015)	-	-	✓	-	✓
TURBO (LEVINSHTEIN et al., 2009)	-	✓	✓	-	-
SLIC (ACHANTA et al., 2012)	-	-	✓	-	-
LSC (LI; CHEN, 2015)	-	-	✓	-	-
SCoW (HU; ZOU; LI, 2015)	-	-	✓	-	-
DBSCAN (SHEN et al., 2015)	-	-	✓	-	✓
SNIC (ACHANTA; SUSSTRUNK, 2017)	✓	-	✓	-	✓
Fuzzy-SLIC (WU; ZHANG L .AND ZHANG; YAN, 2019)	-	-	✓	✓	✓

Source: The Author.

Table 3.1 compares some relevant properties of the different superpixels approaches mentioned above. In Table 3.1, each method appears in a row and has the property with a check mark (✓). The properties (P) listed in Table 3.1 are: (P1) Spatial relationship, or if the method uses a graph structure to ensure connectivity; (P2) Multi-scale superpixels representation; (P3) Control of the number of superpixels by defining the number of superpixels as an input parameter; (P4) Ability to obtain the exact number of superpixels defined as an input parameter; some algorithms define an input parameter to comply with P3, but they do not reach the exact number of superpixels required; and (P5) Irregularity of the superpixel shape at coarser scales, or when a small number of superpixels is required.

In terms of the properties P1 and P2, different superpixels generation methods (ERS, SEEDS, TURBO, SLIC, LSC, SCoW and DBSCAN, SNIC, Fuzzy-SLIC) can obtain fine and coarse superpixels by using different parameter settings, but this scheme tends to result in different numbers of superpixels. Also, the structural relationship among fine and coarse superpixels is not emphasized by these methods (e.g. superpixels boundaries at fine and coarse scales may mismatch). In terms of the properties P3 and P4, usually the segmentation methods (e.g. NCUT, MST and MSHIFT) can not control the specific number of superpixels to be generated. Also, it shall be observed that irregular superpixels shapes tend to obtain better adherence to the image objects boundaries, and

lead to a better representation of texture regions (property P5) at coarser scales (or when the number of superpixels is small). Additionally, the proposed method tends to create irregularly shaped superpixels, with better boundary adherence than other superpixels methods.

Other image segmentation algorithms can be used to generate superpixels (e.g. Statistical Region Merging (NOCK; F., 2004) and Stochastic Region Merging (WONG; SCHARCANSKI; FIEGUTH, 2011)). However, these algorithms often do not allow an adequate control of the number of superpixels, despite providing good boundary adherence to the objects boundaries in an image.

3.1 Spectral Segmentation

Color image segmentation is a crucial step for several computer vision applications. The main color segmentation approaches can be categorized as (PAL; PAL, 1993): i) threshold-based, ii) edge-based, iii) region-based, iv) clustering-based, and v) hybrid methods.

Spectral Clustering is a cluster-based segmentation approach that uses an eigen-decomposition of an image similarity matrix to partition an image into regions. Usually, this is a pixel oriented method and due to the increasingly larger number of pixels in images, scalability may be challenging when dealing with larger images (SHI; MALIK, 2000). Some authors proposed to perform the eigen-decomposition on a downsized image, but the loss of finer image details may cause segmentation inaccuracies. Block-based spectral clustering has been proposed to segment larger images (TUNG; WONG; CLAUSI, 2010), (WANG; FEIPING; YU, 2017) and (LI et al., 2018), exchanging larger memory requirements for higher computational complexity to perform block-based spectral clustering. Also, multi-scale spectral clustering for image segmentation was proposed (ZHANG et al., 2018) and (ZHANG et al., 2019) to over-segment the image using tree-structured superpixels (e.g. quad-trees and kd-trees), and then obtaining k-dimensional eigenspaces for clustering superpixels into k clusters using k-means, and obtaining an image segmentation.

Recently, stochastic graph segmentation methods (WONG; SCHARCANSKI; FIEGUTH, 2011; VASQUEZ; SCHARCANSKI; WONG, 2015b) and graph signal processing methods (CHEUNG et al., 2018; FRACASTORO et al., 2015) were proposed. Most of these algorithms perform graph-based stochastic region merging at the pixel level, trying to

handle noise, color and illumination variations since these are common issues in natural images segmentation. Unfortunately such methods tend to over-segment the input image.

The theory about this algorithm is detailed in the Section 2.3 and we use our superpixels result to handle the scalability challenge to propose a new spectral image segmentation.

Others recent algorithms that use the superpixels segmentation as initialization step are the Superpixel-based Fast Fuzzy C-Means Clustering (SFFCM) (LEI et al., 2018) and the Improved Superpixel-based Fast Fuzzy C-Means Clustering (ISFFCM) (WU et al., 2019), both proposed methods uses a different superpixels method and a clustering Fuzzy C-Means step as last step for segmentation. SFFCM has the problem with the boundary definition due to the post-processing step related to the dilated image, ISFFCM uses the Fuzzy-SLIC algorithm to obtain superpixels and to improve the boundary adherence, but in noise-free environment it has worse performance than the SFFCM, because the last clustering step using Fuzzy C-Means does not have good results when the number of superpixels is big and can not to generate connected regions for only one label.

3.2 Discussion

The state-of-the-art in superpixels generation methods is growing every year adding new features like multi-scale representations (WEI et al., 2018; VASQUEZ; SCHARCANSKI, 2018), and these methods already outperform the SLIC method in terms of boundary adherence. The SLIC method continues as a very popular approach, but it has some issues related to the clustering approaches such as: i) defines an initial regular distribution grid of centroids for all superpixels, but the object regions do not have regular distribution (localization), and ii) the shapes of the objects regions in natural images are not regular and their boundaries are not as a straight line. So, when the algorithm produces boundaries like a straight line and very compact shapes, it tends to generate worse boundary adherence. In case of traditional graph-based methods is difficult to control the number of desired superpixels and for several cases is difficult to control the superpixel size. Also, is important mentioning that all popular methods just can produce single-scale superpixels and for all them is not possible generate multi-scale superpixels with boundary correspondence between the different scales.

(WEI et al., 2018) and (VASQUEZ; SCHARCANSKI, 2018) show the new trends of the superpixels algorithms, where the superpixels have better boundary adherence to

the objects in the image, also they have irregular shapes and size, and the superpixels have a multi-scale representation with boundaries correspondence among different scales.

Motivated to tackle the aforementioned superpixels generation key challenges, specially in natural images, this thesis proposes new superpixel generation methods that use stochastic strategy, with the purpose of obtain superpixels with better boundary adherence to image objects than comparable superpixels methods, and with complexity compatible with a pre-processing step of more complex computer vision processes, such as spectral image segmentation.

In the case of spectral segmentation, we identify some challenges, such as scalability problem of the eigen-decomposition task, when the number of elements of the graph is big. Also the variability of the color in natural images is a challenge to the initialization step of spectral segmentation, because it is needed to obtain a good boundary adherence.

4 PROPOSED METHODS FOR GENERATING STOCHASTIC SUPERPIXELS

We proposed three new superpixel generation methods from natural images. One single-scale method, namely *Stochastic graph Contraction* (SGC) (see section 4.2); and two multi-scale superpixels methods, namely *Iterative Hierarchical Stochastic Graph Contraction* (IHSGC) (see Section 4.1) and *Hierarchical Stochastic Graph Contraction* (HSGC) (see Section 4.2), where SGC and HSGC are improved versions of IHSGC.

The proposed IHSGC method uses a stochastic strategy to cluster visual features, and represent images by nested hierarchical graph structures (i.e. nested hierarchical superpixels) based on an unsupervised over-segmentation of the input image. The proposed approach creates multi-scale superpixels that are structured as nested trees, while preserving the boundary correspondences and the adjacency of the superpixels at finer and coarser scales (i.e., provides locality preservation at different scales). In the proposed approach, the multi-level relationship between finer and coarser superpixels (and their boundaries) is explicit (i.e. pixels are accessible at different scales).

The initialization step of the proposed scheme creates an over-segmentation by using multi-channel stochastic graph contractions, starting at the pixel level. This over-segmentation method allows to handle challenging issues often occurring in natural images, such as noise, color and illumination variability, as well as weak boundary separation between the objects. The second step is an iterative hierarchical stochastic graph contraction scheme, which processes the superpixels adjacency graphs at the finest scale, to obtain the superpixels at coarser scales, one scale at a time. Graph contractions are used to obtain the desired number of superpixels at each scale and to ensure graph connectivity.

The proposed methods SGC and HSGC are improved version of IHSGC method, considering to obtain results with single-scale and multi-scale superpixels. SGC is a fast single-scale version of stochastic superpixels, that uses simple features with control of the number of superpixels and size of them. HSGC is a multi-scale version of SGC that improve IHSGC results in term of time and boundary adherence.

The distinctive characteristics of the proposed multi-scale superpixels generation methods (with respect to other methods available in the literature) are : i) uses stochastic graph contractions to cluster visual features hierarchically, while preserving the local spatial relationships of the visual features at different scales (i.e. the proposed method preserves the boundary correspondences and local neighborhoods at different scales); ii) outputs a multi-scale hierarchical structure of nested graphs (i.e. hierarchical superpix-

els), that can represent the image contents with different levels of detail, which can be useful in visual information processing (e.g. in computer vision, image processing and pattern recognition applications).

4.1 Iterative Hierarchical Stochastic Graph Contraction Approach

The main steps of the proposed method for obtaining multi-scale superpixels are the following : i) Initialization step (see Fig.4.1(b)); and ii) Iterative Hierarchical Stochastic Graph Contraction step (IHSGC) (see Fig. 4.1(d)). The *Initialization* step performs a multi-channel stochastic over-segmentation process, and is applied at the pixel level (scale $s = 0$) to obtain the finest superpixel scale $s = 1$ of our hierarchical representation. As mentioned before, the superpixels T_i^s form a superpixel adjacency graph at scale s (i.e. $s = 1$ in this case), and iterative contraction operations of the superpixels adjacency graph (SAG) are used to obtain the next coarser image representation scales.

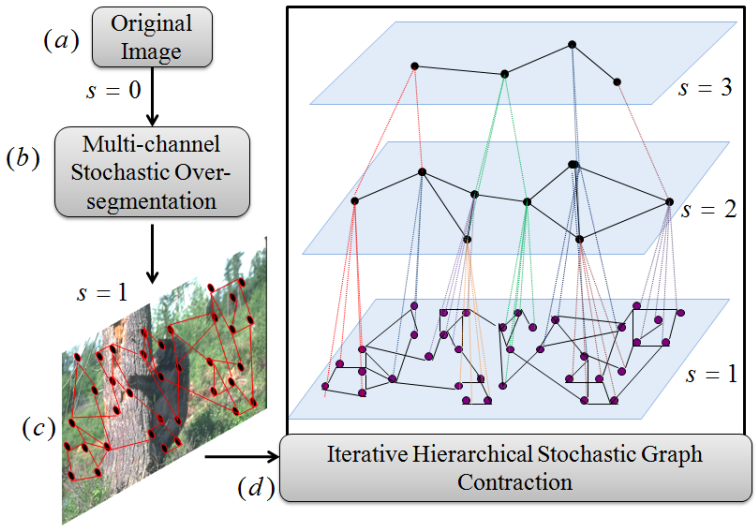
The proposed method is summarized as follows. At the *initialization* step, starting with the original image, an over-segmentation is applied at the pixel level or at $s = 0$ scale, as shown in Figs. 4.1(a) original image, and 4.1(b) initialization step that applies a multi-channel stochastic over-segmentation process at the pixel level ($s = 0$ scale) to obtain the superpixels at scale $s = 1$. The over-segmentation results in small image regions with similar color properties based on CIELAB color space, which represents the local image contents at the scale $s = 1$ with the SAG $G_1 = (V_1, E_1)$ as shown in the Fig. 4.1(c) image segmentation result after applying the initialization step which represents the image at scale $s = 1$ using the superpixel adjacency graph G_1; this *initialization* step tends to generate regions with a good boundary adherence at the finest scale (i.e., $s = 1$, see Section 2.4.1).

Afterwards, at the IHSGC iterative step, the SAG G_1 at the scale $s = 1$ is processed to obtain the multi-scale superpixels at coarser scales by using vertex contraction operations, as in Fig. 4.1(d) iterative hierarchical stochastic graph contraction step (see Secion 4.1.2), which is applied on the superpixel adjacency graph G_1, at scale $s = 1$, to create a hierarchical tree structure for each multi-scale superpixel at a coarse scale s, with $s > 1$.. At each iteration, the SAG G_s at scale s is updated based on the child superpixels $T_j^{s'}$ at finer scales (i.e. $T_j^{s'}$ are child of T_j^s, $s' < s$), and the multi-scale superpixels are used as building blocks to obtain the hierarchical tree structure that represents the image (through multi-scale superpixels). It shall be clarified that a superpixel $T_i^{k_1}$ at finer scale

$s = k_1$ with $k_1 \geq 1$ is included in only one superpixel $T_j^{k_2}$ at each coarser scale k_2, considering that $k_2 > k_1$. Next, the main steps of our proposed approach are presented in more detail.

Fig. 4.2 shows the algorithm in the form of a flow chart that describes dynamics of the proposed iterative hierarchical method. In this flow chart, an natural image is the input. This input image is over-segmented in the initialization step, and the over-segmented image is then represented iteratively at several scales. At each scale, the iterative superpixel generation process applies stochastic graph contraction operations on the graph edges to obtain the desired number of superpixels. As the number of superpixels is reached at each scale, a set of multi-scale superpixels is iteratively obtained at the subsequent scales, allowing to represent the image contents at finer and coarser scales, while the multi-scale superpixels boundaries correspond at different scales.

Figure 4.1: Block diagram of the proposed Iterative Hierarchical Stochastic Graph Contraction (IHSGC) approach. (a) original image, (b) initialization step that applies a multi-channel stochastic over-segmentation process at the pixel level ($s = 0$ scale) to obtain the superpixels at scale $s = 1$. (c) image segmentation result after applying the initialization step which represents the image at scale $s = 1$ using the superpixel adjacency graph G_1; (d) iterative hierarchical stochastic graph contraction step.

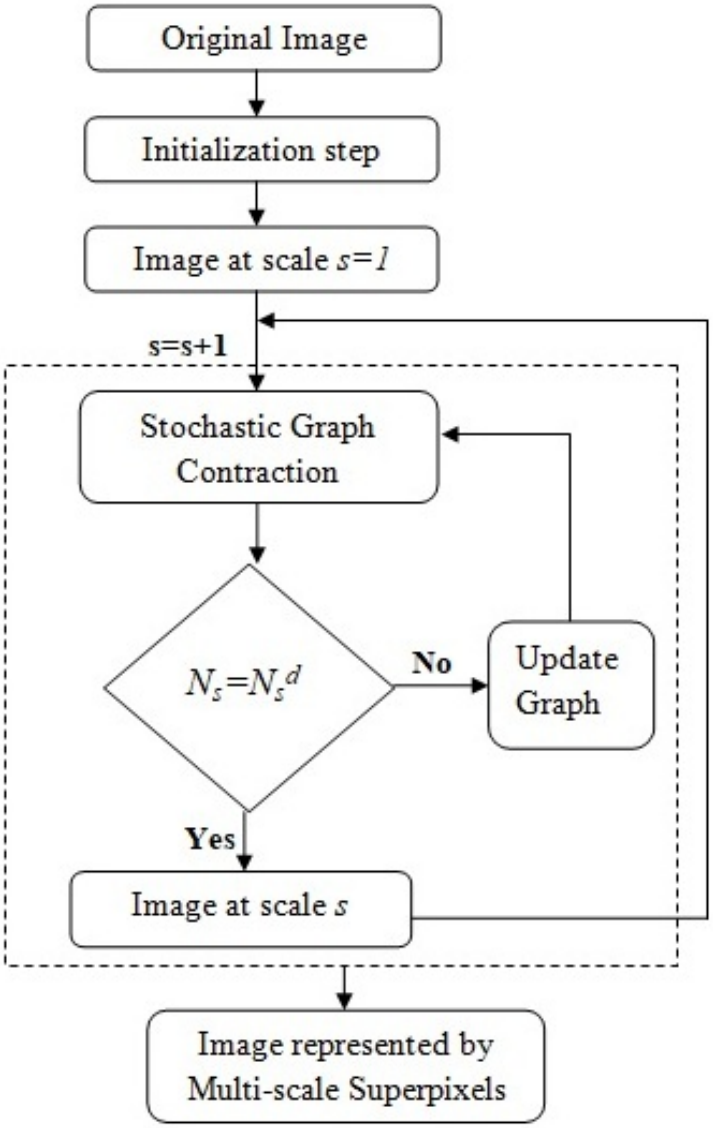


Source: The Author.

4.1.1 Initialization

To initialize the hierarchical image representation process, an unsupervised stochastic over-segmentation scheme is used at the pixels level (i.e., at $s = 0$ scale). This initialization step is the multi-channel algorithm proposed in (VASQUEZ; SCHARCANSKI;

Figure 4.2: Flow chart of the proposed Iterative Hierarchical Stochastic Graph Contraction (IHSGC) approach. Describes dynamics of the proposed iterative hierarchical method. In this flow chart, an natural image is the input. This input image is over-segmented in the initialization step, and the over-segmented image is then represented iteratively at several scales. At each scale, the iterative superpixel generation process applies stochastic graph contraction operations on the graph edges to obtain the desired number of superpixels.



Source: The Author.

WONG, 2015b). To obtain the superpixels at finest scale $s = 1$, the original RGB color image is converted to the CIELAB color space. An adjacency graph is built at the pixels level ($s = 0$) using the 4-connectivity discrete pixels lattice Z. The graph edges link each pixel to its four neighbors, and parallel edges are removed. This pixels adjacency graph is created as an undirected graph $G_0 = (V_0, E_0)$ with $v_i \in V_0$ vertices and edges $e_m \in E_0$, where $e_m = (v_p, v_q)$ links pairs of adjacent pixels assuming 4-connectivity on the discrete

lattice $Z(x, y)$ at the scale $s = 0$. Each pixel $Z(x, y)$ is a vertex, and its four neighbors $Z(x-1, y), Z(x+1, y), Z(x, y-1), Z(x, y+1)$ linked to $Z(x, y)$ define the adjacency graph edges linked to $Z(x, y)$. Each edge e_m has a weight w_m which represents the color dissimilarity between the adjacent pixels v_i and v_j, and this color dissimilarity is defined as the Euclidean distance between the CIELAB colors of a pair of adjacent pixels (v_i, v_j) as follows:

$$w_{ij} = \sqrt{(L_i - L_j)^2 + (b_i - b_j)^2 + (a_i - a_j)^2}. \tag{4.1}$$

In order to obtain the initial over-segmentation, the adjacency graph edges are sorted in ascending order of their edge weights w_{ij}, and the edges e_m are processed in the ascending order of their weights to evaluate what vertices will be contracted by using the following stochastic strategy:

$$C_{fs}(v_i, v_j) = \begin{cases} & \tau_L \leq \alpha_L(v_i, v_j) \text{ and }, \\ 1 \quad , \text{ if } & \tau_a \leq \alpha_a(v_i, v_j) \text{ and }, \\ & \tau_b \leq \alpha_b(v_i, v_j); \\ 0 \quad , \text{ otherwise,} & \end{cases} \tag{4.2}$$

where the three random numbers τ_L, τ_a and $\tau_b \subset [0, 1]$ are obtained from a uniform distribution, and $\alpha_a(v_i, v_j)$, $\alpha_b(v_i, v_j)$ and $\alpha_L(v_i, v_j)$ denote the vertex contraction likelihoods functions (see Eq. (4.3)), the random values τ_L, τ_a and τ_b are used to avoid local minima in the merging process. If $[\tau_a \leq \alpha_a(v_i, v_j)] \wedge [\tau_b \leq \alpha_b(v_i, v_j)] \wedge [\tau_L \leq \alpha_L(v_i, v_j)]$ then the adjacency graph vertices v_i and v_j are contracted (i.e. their associated image regions are merged), otherwise the vertices are not contracted (i.e. their associated image regions are not merged).

We extend the approach in (WONG; SCHARCANSKI; FIEGUTH, 2011) to obtain a vertex contraction predicate (see Eq. 4.2), which decides if the vertices v_i and v_j should be contracted by evaluating the contraction likelihood functions $\alpha_{ch}(v_i, v_j)$ for each pair of adjacent vertices v_i and v_j. In our multi-channel stochastic over-segmentation scheme, each site $Z(x, y)$ is associated with a unique vertex $v_i \subset V_0 = \{v_1, ..., v_{M \times N}\}$, and the adjacency graph edges E_0 connect each vertex (v_i) to its adjacent graph vertices.

As mentioned before, the initialization process is based on the evaluation of the graph edges in ascending order of their edges weights. For each edge $e_m = (v_i, v_j)$ that links the vertices v_i and v_j, the proposed vertex contraction likelihood $\alpha_{ch}(v_i, v_j)$ is calculated for each channel $ch = \{L, a, b\}$ at the scale $s = 0$ as indicated in Eq. (4.3),

and then used in Eq. (4.2).

If a vertex contraction should occur based on the vertex contraction predicate in Eq. (4.2), a new superpixel is obtained at scale $s = 1$ with color equal to the weighted mean color of the vertices v_i and v_j, considering their sizes in terms of number of pixels inside of each vertex. The evaluation of the remainder of the edges in the adjacency graph will continue following the initial ascending order of their edge weights, until all edges at E_0 of the graph $G_0 = (V_0, E_0)$ have been processed. During the process, the vertex labels are updated with the new label for the vertex v_{ij}, because the priority queue has incident edges to the new contracted vertex; these incident edges have their vertices updated and the edges are processed in the initial sorting, but the likelihood function considers the new data for the vertex v_{ij}.

Essentially, the vertex contraction likelihood function $\alpha_{ch}(v_i, v_j)$ estimates the dissimilarity between the neighboring vertices v_i and v_j as follows :

$$\alpha_{ch}(v_i, v_j) = \exp\left(\frac{-(\Psi_{ch}(v_i) - \Psi_{ch}(v_j))^2}{\Lambda(v_i, v_j)}\right), \tag{4.3}$$

where $\Psi_{ch}(v_i)$ is the mean color value of the pixels belonging to the vertex v_i of the adjacency graph at scale $s = 0$, ch represents a channel in $\{L, a, b\}$, and Λ is a statistical contraction penalty function defined as follows :

$$\Lambda(v_i, v_j) = \frac{D_I^2 \ln(N_0)}{Q}\left(\frac{1}{|v_i|} + \frac{1}{|v_j|}\right), \tag{4.4}$$

where N_0 is the number of pixels at the scale $s = 0$ (or the number of image pixels), $|v_i|$ represents the number of elements (pixels) in the adjacency graph vertex v_i, D_I represents the dynamic range of I ($D_I = max\{L, a, b\}$), and Q is a regularization term that controls the vertex contraction rate, because high values of Q ensure low contraction rate and low value of Q tends to generate high contraction rate depending of the image content (e.g. homogeneous or textured content). Using a high value for Q decreases α_k (in our experiments $Q = 2000$). Finally, a refinement step is applied to merge small regions containing less than δ pixels to their most similar big adjacent region (in our experiments $\delta = 7$).

The adjacency superpixel graph $G_1 = (V_1, E_1)$ at the scale $s = 1$ is obtained by vertices contractions (i.e. vertices representing the image pixels at $s = 0$), which leads to superpixels containing pixels with similar colors. Since the obtained adjacency superpixel graph $G_1 = (V_1, E_1)$ tends to contain less superpixels (i.e. vertices) than pixels in the

input image, the obtained adjacency superpixels graph actually is a simplified version of the initial pixels adjacency graph $G_0 = (V_0, E_0)$ (which represents the adjacency of the pixels in the lattice Z at the scale $s = 0$).

The pseudo-code for the initialization step used for generating the finest scale $s = 1$ is presented in Algorithm 2. The initial pixel adjacency graph is created in the steps 1 and 2 by assuming 4-connectivity on the discrete lattice Z. Then, the edge weights are calculated in step 3, and these edge weights are used to decide whether the graph edges should be contracted based on Eq. (4.2).

Let N_0 be the number of pixels in the original image at $s = 0$, the complexity of this initialization step is proportional to the processing of the graph edges in ascending order. The number of the edges at the scale $s = 0$ is smaller than twice the number of pixels, and is approximately $|E_0| \leq 2N_0$, so the worst-case running time is $\mathcal{O}(2N) = \mathcal{O}(N)$. The bottleneck is the ascending order sorting of the graph edges. Then the final worst-case running time is $\mathcal{O}(N \log N)$ since it depends on the sorting algorithm used (e.g. the radix sort algorithm has linear dependence in the image size).

Algorithm 2: Initialization Step

Input : Image I in CIELAB color space and Q parameter.
Output: Graph G_1 that represents the over-segmentation map.

1 Construct an initial pixel adjacency graph $G_0 = (V_0, E_0)$, where each vertex represents a pixel z with four (neighbors) adjacent vertices.
2 Assign a vertex label to each pixel $Z(x, y)$ in the image I.
3 Place all adjacent vertex pairs $e_m = (v_i, v_j)$ (edges) into a priority queue P based on ascending color CIELAB Euclidean distance (see Eq. (4.1)) as its weight w_m.
4 Assign $G_1 = G_0$.
5 **repeat**
6 Remove the first edge of vertex pair v_i and v_j from the priority queue;
7 Generate τ_L, τ_a and $\tau_b \subset [0, 1]$ from an uniform distribution;
8 Calculate the vertex contraction likelihood for each channel α_L, α_a and α_b based on the function in Eq. (4.3);
9 Evaluate the vertices v_i and v_j using the vertex contraction predicate in Eq. (4.2);
10 If a vertex contraction occurs, update the adjacency graph G_1 and the color value of the new vertex v_{ij}, respectively, and then remove the self-loops and parallel edges. Also, the vertex labels are updated to handle the consistence of the edges in P.
11 **until** *Priority queue is empty.*;
12 Return G_1.

4.1.2 Iterative Hierarchical Stochastic Graph Contraction (IHSGC) for Obtaining Coarser Scales

In this step, superpixels are merged iteratively, starting with the initial superpixels obtained during the initialization step (see Section 4.1.1), at the finest scale $s = 1$. The initial superpixels are denoted by T_i^1, since they are the basic components that form the scale $s = 1$ of the tree model, which provides a hierarchical structure for the pixels available at the scale $s = 0$ and the image regions. The superpixels are merged by vertex contractions, based on their edge weights and vertices contraction likelihoods. It shall be observed that the edge weights w_{ij} (see Eq. (4.1)) and the vertex contraction likelihoods $\alpha_a(v_i, v_j)$, $\alpha_b(v_i, v_j)$ and $\alpha_L(v_i, v_j)$ (see Eq. (4.3)) are computed in a different way in the IHSGC step. This alternative way of computing edges weights and vertices contraction likelihoods is used to account for larger sets of pixels than those available at the finer scales, and will be used in the processing of the superpixels adjacency graph at the scales $s > 1$.

Let $G_s = (V_s, E_s)$ be a superpixels adjacency graph (SAG) at a coarser scale s ($s > 1$), with the vertices being (hierarchical) superpixels $V_s = \{T_1^s, T_2^s, T_3^s, ..., T_{N_s}^s\}$, and the edges E_s connecting adjacent superpixels where $E_s = \{e(T_i^s, T_j^s) \,/\, T_i^s \text{ and } T_j^s \text{ are adjacent}\}$. The SAGs $G_2, G_3, ..., G_{N_G}$ are obtained when each new coarser scale $s = 2, 3, ..., N_G$ is obtained, and the number of vertices (superpixels) in each SAG G_s ($s > 1$) tends to be progressively smaller than in the finer scales due to the vertex contraction operation applied to SAG $G_{(s-1)}$ to merge vertices and generate a new SAG G_s. Of course, the vertices (superpixels) at coarser scales $s > 1$ inherit the hierarchical relationship with the SAGs vertices at the finer scales $0 \leq s \leq 1$, since these vertices at finer and coarser scales are related by hierarchical trees, and superpixels at a scale s contain superpixels at a finer scale s' ($s' < s$) (see Section 2.2).

Therefore, at the IHSGC step, the iterative vertex contractions are applied at the SAG (superpixels adjacency graph) G_s at the scale s to obtain a new SAG $G_{s+1} = (V_{s+1}, E_{s+1})$ at scale $s + 1$. As in the initialization step (see Section 4.1.1), at each iteration of the IHSGC step, all edges $e_m = (T_i^s, T_j^s)$ are sorted in ascending order of their weights w_m, and then they are processed in this ascending order. A vertex contraction likelihood $\beta(T_i^s, T_j^s)$ between T_i^s and T_j^s is calculated to decide if these vertices should be contracted. If $\beta(T_i^s, T_j^s) > \tau$, with $\tau \subset [0, 1]$ and drawn from an uniform distribution, the superpixels adjacency graph is updated by merging T_i^s and T_j^s, and then removing

self-loops and parallel edges. At the IHSGC step, there are more pixels associated to each vertex than in the finest scale $s = 1$ (since the superpixels are hierarchical and contain lower scale superpixels and the image pixels at $s = 0$).

Consequently, we handle these larger quantities of pixels by color histograms, that are used as descriptors of the superpixels contents. Also, it shall be observed that 3D CIELAB color histograms H_i^s provide a richer appearance description of the superpixels contents than the local mean colors (GRUNDMANN et al., 2010) used in the initialization step. For instance, textured regions will have flatter histograms while homogeneous regions will have peaks. We use B bins for each dimension of the 3D CIELAB histograns (in our experiments $B = 16$), and each dimension represents a channel of the CIELAB color space (L, a, b).

In the IHSGC step, initially the original image is converted to CIELAB color space, and the superpixels adjacency graph (SAG) $G_s = (V_s, E_s)$ is obtained with the Algorithm 3 to build a new scale $s + 1$. In this algorithm, each superpixel at the finer scale s is described by a 3D CIELAB color histogram (tests on RGB color space obtained worse results compared with those in the CIELAB color space), assuming independent color channels (L, a and b). Finally, the edge weights w_m of the SAG (G_s) are computed based on the chi-square distance $\chi^2(H_i^s, H_j^s)$ between the 3D CIELAB color histograms H_i^s and H_j^s, representing the dissimilarity between a pair of adjacent superpixels T_i^s and T_j^s.

We chose the chi-square distance between 3D CIELAB histograms (χ^2) because normalized histograms with the same number of bins are used, and the chi-square distance takes into account cross-bin relationships while reducing the influence of large bins. Additionally, the chi-square distance has been successfully used in in local texture discrimination problems (PELE; WERMAN, 2010). The chi-square distance (χ^2) between 3D CIELAB histogram H_i^s and H_j^s is calculated as follows:

$$\chi^2(H_i^s, H_j^s) = \frac{1}{2} \sum_{l=1}^{L} \sum_{m=1}^{M} \sum_{n=1}^{N} \frac{\left[H_i^s(l,m,n) - H_j^s(l,m,n)\right]^2}{H_i^s(l,m,n) + H_j^s(l,m,n)}, \tag{4.5}$$

where H_i^s and H_j^s represent two histograms describing the contents of the vertices (superpixels) T_i^s and T_j^s at the scale s, respectively. The χ^2 distance is assigned to the edge weights of the SAG $G_s = (V_s, E_s)$ at scale s.

As in the initialization step (see Section 4.1.1), a vertex contraction likelihood function $\beta(T_i^s, T_j^s)$ is used to decide when adjacent superpixels should be merged, and

consequently the SAG should be refined to obtain the coarser scales, as detailed next. To decide if the two vertices T_i^s and T_j^s of the edge $e_m(T_i^s, T_j^s)$ should be contracted, we use the vertex contraction likelihood $\beta(T_i^s, T_j^s)$ (see Eq.(4.6)) to evaluate whether vertices should be merged based on their similarity. The $\beta(T_i^s, T_j^s)$ function is an extension of $\alpha(v_i, v_j)$ in Section 4.1.1 (see Eq. (4.3)), which was modified to include the χ^2 distance between 3D histograms describing adjacent superpixels (instead of the mean color values of local patches, as in Eq. (4.3)). Therefore, the vertices contraction likelihood $\beta(T_i^s, T_j^s)$ is expressed as follows:

$$\beta(T_i^s, T_j^s) = \exp\left(\frac{-(\chi^2(H_i^s, H_j^s)^2}{\Lambda(T_i^s, T_j^s)}\right), \tag{4.6}$$

where χ^2 is the chi-square distance between the histograms H_i^s and H_j^s associated to the superpixels T_i^s and T_j^s at the scale s, respectively, and Λ is a statistical contraction penalty function defined as follows:

$$\Lambda(T_i^s, T_j^s) = \frac{9\ln(N_0)}{4Q_s}\left(\frac{1}{|T_i^s|} + \frac{1}{|T_j^s|}\right), \tag{4.7}$$

where N_0 represents the number of pixels in the original image (at scale $s = 0$), $|T_i^s|$ represents the number of elements (pixels) in the vertex T_i^s, and Q_s is a regularization parameter that controls the merging process of the superpixels at the scale s ($s > 1$). To obtain the scale $s = 2$, and for each new scale $s + 1$, we use $Q_{s+1} = Q_s/2$ ($Q_2 = 500$ in our experiments). The SAG edges at scale s are processed in ascending order of their edge weights, and the SAG G_s is updated whenever adjacent vertices are contracted. The final SAG $G_{s+1} = (V_{s+1}, E_{s+1})$ represents the image contents at the newly obtained scale $s + 1$. In the Equation 4.4 the parameter D_I^2 represents the dynamic range of the image values, and in the case of Equation 4.7 $D_I^2 = 9/4$ or approximately 1.5 which represents a maximum value of the histogram distance. The regularization parameter $Q_{s+1} = Q_s/2$ varies slightly and may decrease to favor the contraction process.

Also, similar to the initialization step (see Section 4.1.1), during the graph edges processing in ascending order of their weights (i.e. adjacent superpixels dissimilarity), if $\tau \leq \beta(T_i^s, T_j^s)$, where $\tau \in [0, 1]$ is a random number, the vertices T_i^s and T_j^s are contracted, otherwise T_i^s and T_j^s are not contracted. This vertex contraction criterion is

expressed by the contraction predicate $C_{cs}(T_i^s, T_j^s)$ bellow :

$$C_{cs}(T_i^s, T_j^s) = \begin{cases} 1 & , \text{ if } \tau \leq \beta(T_i^s, T_j^s); \\ 0 & , \text{ otherwise .} \end{cases} \tag{4.8}$$

Therefore, the SAG generation methods used in the initialization step (see Section 4.1.1) and in the IHSGC step differ. Initially, the goal is to obtain a large number of superpixels T_i^1 containing pixels with similar color properties. However, in the IHSGC step, at coarser scales, the variability of pixels within each superpixel tends to increase and the scale s increases, and richer descriptions of the superpixels T_i^s contents are obtained with 3D color histograms, and an adequate vertices likelihood contraction function $\beta(T_i^s, T_j^s)$ is used for superpixels similarity evaluation and contraction decisions.

Let N_0 the number of pixels of the original image and N_1 the number of superpixels obtained in the initialization step, where $N_1 < N_0$. Let N_s be the number of superpixels at the scale s ($s > 1$). The complexity of the IHSGC step is proportional to the number of graph edges processed $e_m = (T_i^s, T_j^s)$. Usually, $N_s << N_0$, and if the graph edges are processed in ascending order of their weights (i.e. χ^2 distances among histograms), the worst-case running time for the edges processing is $\mathcal{O}\left(N_s \left(\log N_s + (N_0/N_s)\right)\right)$ for each new coarser scale obtained, which already includes expensive computational tasks such as sorting and histogram generation, where bin_L, bin_a and bin_b are the number of bins in each dimension of the 3D color histograms of the L, a and b channels, respectively.

Fig. 4.3 illustrates the results obtained in the IHSGC step. In the first row of Fig. 4.3, from left to right, the results for 2500, 1000 and 500 superpixels are shown in Fig. 4.3(a), Fig. 4.3(b) and Fig. 4.3(c), respectively. In the second row are shown the results for the 250, 100 and 50 superpixels in Fig. 4.3(d), Fig. 4.3(e) and Fig. 4.3(f), respectively. The boundaries (green lines) in Figs. 4.3(a) to 4.3(f) show the evolution of the superpixels contraction for the foreground object (i.e. the bear). Fig. 4.3(a) shows the foreground object with several smaller superpixels, and the foreground object appears in Fig. 4.3(f) with just one superpixel. At each iteration all edges of the superpixels adjacency graph are processed until the desired number of superpixels at each scale is reached.

Algorithm 3: Iterative Hierarchical Stochastic Graph Contraction (IHSGC)

Input : Image I in CIELAB color space, SAG at the finer scale s, $G_s = (V_s, E_s)$ and the desired number of superpixels $N_{desired}$.

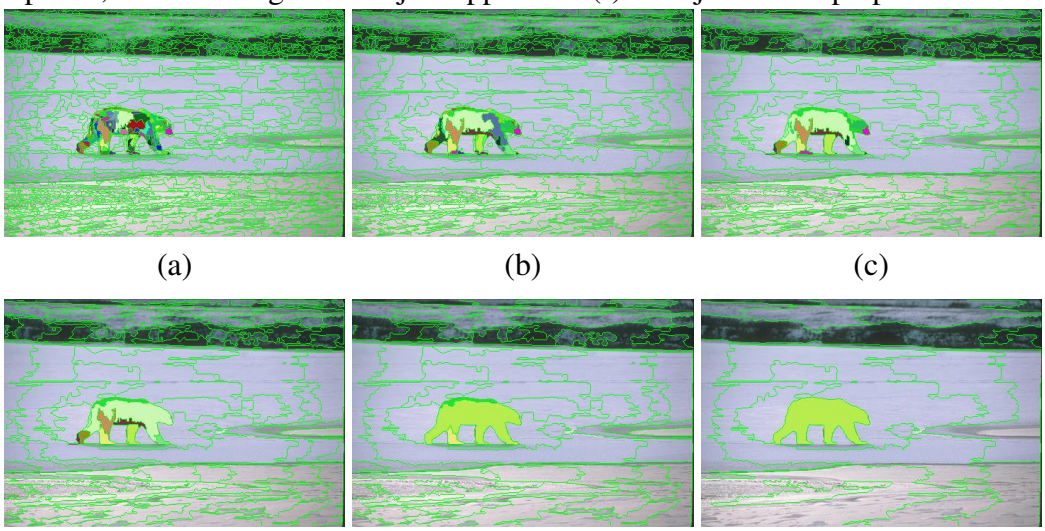
Output: SAG $G_{s+1} = (V_{s+1}, E_{s+1})$ at the coarser scale $s+1$ that represents coarse over-segmentation of the original image.

Assign $Q_2 = 500$.
repeat
  Assign to each vertex in $G_s = (V_s, E_s)$ an unique superpixel label;
  Create the N_s superpixels T_i^s, $i = 1, ..., N_s$, at the scale s;
  Calculate for each superpixel T_i^s a 3D CIELAB color histogram H_i^s;
  Place all pairs of adjacent superpixels $\left(T_i^s, T_j^s\right)$ that define edges e_m in $E_s = \left\{e_m\left(T_i^s, T_j^s\right) \mid T_i^s \text{ and } T_j^s \text{ are adjacent}\right\}$ into a priority queue PQ based on the ascending order of their $\chi^2(H_i^s, H_j^s)$ distances of histograms H_i^s and H_j^s as in the Eq. (4.5) to define the edge weights w_m;
  Assign $G_{s+1} = G_s$;
  repeat
    Remove the first superpixels pair $\left(T_i^{s+1}, T_j^{s+1}\right)$ from priority queue PQ;
    Generate $\tau \subset [0, 1]$ from an uniform distribution;
    Calculate the vertex contraction likelihood $\beta\left(T_i^{s+1}, T_j^{s+1}\right)$ based on the function in Eq. (4.6);
    Decide if the vertices T_i^{s+1} and T_j^{s+1} are going to be contracted using the vertex contraction predicate in Eq. (4.8);
    If the vertex contraction occurs, update the superpixels adjacency graph G_{s+1} at scale $s+1$ by merging vertices T_i^{s+1} and T_j^{s+1}, and remove self-loops and parallel edges. Update the histogram descriptors H_{ij}^{s+1} for the new superpixel T_{ij}^{s+1} at scale $s+1$, and decrease the number of superpixels $N_{s+1} = N_{s+1} - 1$. Also, the vertex label of the new contracted vertex is updated for its incident edges in PQ;
  until *the priority queue PQ is empty, or* $N_{s+1} == N_{desired}$;
  Assign $Q_{s+1} = Q_s/2$;
until $N_s == N_{desired}$;
Return G_{s+1}.

4.2 Improved Method for Generating Hierarchical Stochastic Superpixels

We proposed two improved approaches of IHSGC, namely, *Stochastic Graph Contraction* (SGC) and *Hierarchical Stochastic Graph Contraction* (HSGC) approaches which are an unsupervised stochastic superpixels generation schemes. These methods outperforms the IHSGC approach in term of connectivity using 8-neighbors for pixels level connectivity and simple features such as the weighted mean CIELAB color for each superpixel. Also, these methods SGC and HSGC, generate superpixels at single-scale

Figure 4.3: Example of the iterative hierarchical stochastic graph contraction step for an image of the BSDS500 dataset. In the first row, from left to right, the results for 2500, 1000 and 500 superpixels are shown in (a), (b) and (c), respectively. In the second row are shown the results for the 250, 100 and 50 superpixels in (d), (e) and (f), respectively. The boundaries (green lines) in (a) to (f) show the evolution of the superpixels contraction for the foreground object (i.e. the bear). (a) shows the foreground object with several smaller superpixels, and the foreground object appears in (f) with just one superpixel.



(a) (b) (c)

(d) (e) (f)

Source: The Author.

and multi-scale superpixels, respectively, with high boundary adherence and boundaries correspondence among scales.

Other issues of the IHSGC method are the runtime to obtain several scales of representation ($320s$ for 7 scales) in comparison with HSGC method ($5s$ for 7 scales). Also the IHSGC method shows small superpixels when the image has textured regions, this occurs because we are using just a color descriptor; and we are not using some control of the superpixel size (see Fig. 4.3). Then, all of these issues are solved with the new proposed methods SGC and HSGC (see Fig. 4.4 and 4.5). The SGC method was proposed to approach the single scale superpixels problem with a reasonable runtime. To approach the multiscale superpixels problem and to add some control of superpixel size, the HSGC method was proposed as an extension of the SGC method for hierarchical image processing and representation.

Stochastic Graph Contraction *(SGC)* starts its processing at the pixels level and generates a single-scale superpixels over-segmentation. The original RGB color image is converted to the CIELAB color space, and an adjacency graph is built at the pixels level using the 8-connectivity discrete pixels lattice Z, so the graph edges linking each pixel to its eight neighbors. This pixels adjacency graph is created as an undirected graph $G_0 =$

(V_0, E_0) with $v_i \in V_0$ vertices and edges $e_m \in E_0$, where $e_m = (v_p, v_q)$ links pairs of adjacent pixels assuming 8-connectivity on the discrete lattice $Z(x, y)$ at the scale $s = 0$. Each pixel $Z(x, y)$ is a vertex, and its eight neighbors $Z(x-1, y), Z(x+1, y), Z(x, y-1), Z(x, y+1), Z(x-1, y-1), Z(x+1, y-1), Z(x-1, y+1), Z(x+1, y+1)$ define the adjacency graph edges linked to $Z(x, y)$.

Each edge $e_m = (v_i, v_j)$ has a weight w_m which represents the color dissimilarity between the adjacent pixels v_i and v_j, and the color dissimilarity is defined as the Euclidean distance between the CIELAB colors of a pair of adjacent pixels (v_i, v_j) using the Eq. 4.1. The adjacency graph edges are sorted in ascending order of their edge weights w_{ij}, and the edges e_m are processed in the ascending order of their weights to evaluate what vertices will be contracted if $C_{fs}(v_i, v_j) = 1$, where $C_{fs}(v_i, v_j)$ values are obtained by using the stochastic strategy using Eq. 4.2 and the vertex contraction likelihood $\alpha_{ch}(v_i, v_j)$, $ch = \{L, a, b\}$ is calculated for each channel as indicated in Eq. (4.3).

In our SGC scheme, each site $Z(x, y)$ is associated with a unique vertex $v_i \subset V = \{v_1, ..., v_{M \times N}\}$, and the adjacency graph edges E connect each vertex (v_i) to its adjacent graph vertices. The vertex contraction should occur based on the vertex contraction predicate in Eq.(4.2), until a maximum superpixel size is reached, and then for each new contracted vertex the new color assigned to the superpixel is the weighted mean color of the vertices v_i and v_j, considering their superpixel sizes in term of number of pixels.

The superpixels size is defined as $TAM = N_0/N_d$, where N_0 is the number of pixels in the image, and N_d is the desired number of superpixels. The evaluation of the remainder of the edges in the adjacency graph will continue according to the the initial ascending order of their edge weights, until all edges E of the graph $G = (V, E)$ have been processed, but the labels of the vertices in the incident edges to the new contracted vertex (v_{ij}) are updated, so the contraction likelihood α_{ch} will use the new color for v_{ij} (in our experiments $Q = 200$).

The outcome of SGC is an adjacency superpixels graph which represents the superpixels finer scale can be defined as $G_1 = (V_1, E_1)$ at the scale $s = 1$. In this case G_1 tends to contain less vertices than the number of pixels in the input image, and it leads to a simplified version of the initial pixels adjacency graph $G_0 = (V_0, E_0)$.

The pseudo-code for our SGC method proposed for generating stochastic superpixels at $s = 1$ is in Algorithm 2. The initialization and the initial pixel adjacency graph are in steps 1 to 3, assuming 8-connectivity in the discrete lattice Z. Then, the edge weights are calculated in step 5, and in steps 6 to 11 these edge weights are sorted and

used to decide whether the graph edges should be contracted based on Eq. (4.2). If the current number of superpixels still is larger than the desired number of superpixels ($N_v > N_{sp}$), the variable TAM is increased (in our experiments TAM is increased by 0.5%) to reach the desired number of superpixels N_{sp}, as indicated in steps 12 and 13.

Let N_0 be the number of pixels in the original image, the complexity of this proposed SGC is proportional to the processing of the graph edges in ascending order of the edges weights. The number of the edges E_0 in the initial graph is approximately $|E_0| \leq 4N_0$, so the worst-case running time is $\mathcal{O}(4N_0) = \mathcal{O}(N_0)$. The bottleneck is the ascending order sorting of the graph edges. Then the final worst-case running time is $\mathcal{O}(N_0 \log N_0)$ since it depends on the sorting algorithm used.

Algorithm 4: Stochastic Graph Contraction (SGC) Algorithm.

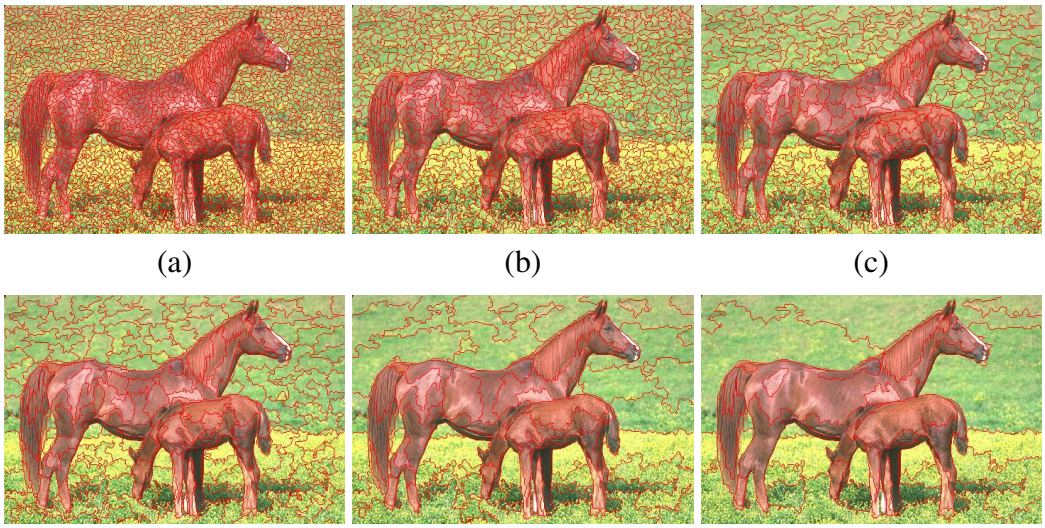
Input : Image I in CIELAB color space and the desired number of superpixels N_{sp}.
Output: Superpixels Adjacency Graph (SAG) G_s, $s = 1$.

1 Build the initial pixels adjacency graph $G_0 = (V_0, E_0)$, where the pixels $Z(x, y)$ are vertices, each vertex with 8 adjacent vertices;
2 Assign a vertex label to each $Z(x, y)$ in the image I;
3 Assign $G_1 = G_0$, $Q = 200$, $rateTAM = 0.5 * TAM$, number of vertices $N_v = N_0$, maximum superpixels size $TAM = N_0/N_{sp}$;
4 **repeat**
5 Place all edges $e_m = (v_i, v_j) \in G_s$ in a priority queue P in ascending order of their color CIELAB Euclidean distances (see Eq. (4.1));
6 **while** $P \neq \emptyset$ **do**
7 Remove the edge $e_m = (v_i, v_j)$ and its associated vertices v_i and v_j from the top of P;
8 Compute $\{\tau_L, \tau_a, \tau_b\} \in [0, 1]$ for the $\{L, a, b\}$ channels using Eq. 4.2;
9 Calculate the vertex contraction likelihood $\alpha(v_i, v_j)$ using Eq. (4.3);
10 If the vertices contraction predicate in Eq. (4.2) $C_{fs}(v_i, v_j) = 1$ then v_i and v_j are contracted, $(size(v_i) + size(v_j)) \leq TAM$ and $N_v > N_{sp}$, update the SAG G_s: replace the contracted vertices v_i and v_j by a new superpixel v_{ij}^s; remove self-loops and parallel edges; update the weigthed mean color for v_{ij}^s as the mean CIELAB color of v_i and v_j; decrease the number of superpixels to process in scale s to $N_v = N_{v-1}$;
11 **end**
12 Assign $TAM = TAM + rateTAM$ and $Q = Q/2$;
13 **until** *($N_v == N_{sp}$)*;
14 Return G_1.

Fig. 4.4 illustrates the results obtained using SGC method. In the first row of Fig. 4.4, from left to right, the results for 2500, 1000 and 500 superpixels are shown in Fig. 4.4(a), Fig. 4.4(b) and Fig. 4.4(c), respectively. In the second row are shown the

results for the 250, 100 and 50 superpixels in Fig. 4.4(d), Fig. 4.4(e) and Fig. 4.4(f), respectively. The boundaries (red lines) in Figs. 4.4(a) to 4.4(f) show the evolution of the superpixels contraction. At each iteration all edges of the superpixels adjacency graph are process until the desired number of superpixels at each scale is reached.

Figure 4.4: Illustrations of the stochastic graph contraction (SGC) method to obtain the superpixels for an image of the BSDS500 dataset. In the first row, from left to right, the results for 2500, 1000 and 500 superpixels are shown in (a), (b) and (c), respectively. In the second row are shown the results for the 250, 100 and 50 superpixels in (d), (e) and (f), respectively. The boundaries (red lines) in (a) to (f) show the evolution of the superpixels contraction.



(a) (b) (c)

(d) (e) (f)

Source: The Author.

A simple extension of SGC to obtain multi-scale superpixels is proposed, namely, ***Hierarchical Stochastic Graph Contraction*** *(HSGC)*, we use the *hierarchical tree model* defined in (VASQUEZ; SCHARCANSKI, 2018) to represent the relationship between adjacent superpixels (T_i^s) at finer and coarser scales (see Fig. 2.4). At the end of the multi-scale superpixels generation process, the obtained hierarchical trees (T) describe the multi-scale superpixels, and can be used to represent the image contents at different levels of detail in multiple scales. This hierarchical approach is a natural complement of the proposed stochastic graph contraction (SGC) method, and it is initially applied at the superpixel level $s = 1$ to generate coarser representations of the image containing larger superpixels that exhibit highly homogeneous properties and strong adherence to object boundaries within the scene, even in the presence of uncertainty due to non-ideal scene conditions.

As mentioned before, the multi-scale superpixels T_i^s form a superpixels adjacency

graph at scale s, and iterative stochastic contraction operations of the superpixels adjacency graph (SAG) are used to obtain the coarser image representation scales (i.e. in this case if $s > 1$, the scale previous s is the input). In each scale $s > 1$ a new superpixel size is defined as $TAM_s = N_0/N_d$ according to the number of superpixels specified for that scale s. So, the proposed method is summarized as follows.

Recall that the initial stochastic superpixel generation stage works with the original image pixels, since SGC is applied at the pixel level. Consequently, $G_1 = (V_1, E_1)$ at the finest superpixels scale $s = 1$ tends to have smaller size superpixels, and these superpixels tend to be homogeneous in terms of CIELAB color properties. An iteration of the HSGC method obtains the superpixels at scale $s' > s, s > 0$ based on the children superpixels T_j^s available at the finer scale s (i.e. if $s' > s$, then $T_j^{s'}$ is a parent superpixel of T_j^s), as well as the SAG G_s' [1].

Therefore, in the hierarchical relationship between the multi-scale stochastic superpixels generated by the proposed HSGC method, the multi-scale representations are nested. In other words, superpixels at finer scales $T_i^{s'}$, $s' > 0$, are contained within the superpixels at coarser scales T_i^s, $s > s'$, and the superpixels T_i^s at the same scale s are disjoint sets of superpixels and pixels. Also, for each scale $s > 0$, each superpixel i, T_i^s, is a root of a rooted tree and can represent a rooted tree itself if $s > 1$. Each superpixel T_i^s at scale s is part of an adjacency graph of multi-scale superpixels (G_s) at scale s.

The proposed HSGC algorithm for generating multi-scale stochastic superpixels is described in detail in Algorithm 5. The proposed method is initialized with the finest superpixels scale SAG $G_1 = (V_1, E_1)$ (which has N_1 superpixels). Recall that SAG $G_1 = (V_1, E_1)$ was obtained by Algorithm 4 (see Section 4.2), given the CIELAB input image I. The additional HSGC parameters are the number of new nested scales k^d to be obtained (so, the total number of scales is $k^d + 1$), and the desired number of superpixels N_s^d at each additional scale s (i.e., $[N_2^d, N_3^d, ..., N_{k+1}^d]$). The HSGC method provides the set of nested SAGs for the additional k^d coarser scales. In other words, the obtained SAGs are hierarchical, and superpixels (i.e., SAGs vertices) at finer scales are entirely contained in coarser scales superpixels that cover them [2]. Consequently, there is boundary correspondence between the superpixels in a SAG G_s at a coarser scale s and the superpixels in the SAG G_{s-1} at finer scales $s - 1$. The weights w_m of the edges e_m in

[1] When building the superpixels hierarchy, superpixels at finer scales T_j^s are used to obtain the coarser scales superpixels $T_j^{s'}$, where $s < s'$. But, obtained the superpixels hierarchical tree structure, $T_j^{s'}$ is a parent superpixel of T_j^s if $s' > s$.

[2] A superpixel T_i^s spatially cover the finer scales child superpixels T_j^{s-1} if all pixels in T_j^{s-1} ($\cup_{x,y} Z(x, y) \in T_j^{s-1}$) also are pixels of T_i^s ($\cup_{child} T_j^{s-1} \in T_j^{s-1}$).

$E_s = \{e_m(T_i^s, T_j^s) | T_i^s \text{ and } T_j^s \text{ are adjacent}\}$ are given by the Euclidean distances between the mean CIELAB colors of their associated superpixels T_i^s and T_j^s (see Eq. 4.1). The edges e_m and their associated superpixels T_i^s and T_j^s are placed in a priority queue P in the ascending order or their weights w_m, so adjacent superpixels similar in color are placed at the top of P and processed first. Next, the SAG edges e_m of G_s at the top of P, and their associated superpixels T_i^s and T_j^s, are removed from P to test if T_i^s and T_j^s should be contracted to generate an additional superpixel T_{ij}^s, replacing T_i^s and T_j^s in G_s. This contraction decision requires the stochastic predicate in Eq. 4.2 to be $C_{fs}(T_i^s, T_j^s) = 1$, and also that the current size of the new superpixel T_{ij}^s is not higher than the superpixel size TAM_s defined for the scale s. If the contraction of T_i^s and T_j^s takes place, the weighted mean CIELAB color of T_i^s and T_j^s becomes the mean color of T_{ij}^s, and the number of superpixels at scale s is decreased to $N_s = N_s - 1$. The superpixels contractions tests continue following the order in P until the desired number of superpixels N_s^d for the scale s is reached.

Let N_1 be the number of superpixels obtained in the SGC step, where $N_1 < N_0$. Let N_s be the number of superpixels at the scale s ($s > 1$). The complexity of the HSGC step is proportional to the number of graph edges processed $e_m = (T_i^s, T_j^s)$. Usually, $N_s << N_0$, and if the graph edges are processed in ascending order of their weights (i.e. using Euclidean distances of average colors), the worst-case running time for the edges processing is $\mathcal{O}(N_0(\log N_0) + \sum_{s=1}^{M} N_s(\log N_s))$ for each new coarser scale obtained.

Fig. 4.5 illustrates the results obtained in the HSGC method. In the first row of Fig. 4.5, from left to right, the results for 2500, 1000 and 500 superpixels are shown in Fig. 4.5(a), Fig. 4.5(b) and Fig. 4.5(c), respectively. In the second row are shown the results for the 250, 100 and 50 superpixels in Fig. 4.5(d), Fig. 4.5(e) and Fig. 4.5(f), respectively. The boundaries (red lines) in Figs. 4.5(a) to 4.5(f) show the evolution of the superpixels contraction. At each iteration all edges of the superpixels adjacency graph are process until the desired number of superpixels at each scale is reached.

Fig. 4.6 illustrates the results obtained by the HSGC method for an image of the BSDS500 dataset. In the left column are shown the results for 4.6(a) 300, 4.6(c) 100 and 4.6(e) 10 superpixels. In the right column are shown the results for a multi-scale superpixel of the foreground object (i.e. the bird) using the hierarchical tree representation from coarser to fine scales (see 4.6(b), 4.6(c) and 4.6(f)). The boundaries (green lines) in Figs. 4.6(a), 4.6(c) and 4.6(e) show the evolution of the superpixels contraction. Figs.4.6(b), 4.6(d) and 4.6(f) shows the foreground object with several smaller superpixels with dif-

Algorithm 5: Hierarchical Stochastic Graph Contraction (HSGC) Algorithm.

Input : CIELAB color image I, SAG $G_1 = (V_1, E_1)$ at scale $s = 1$;
CIELAB colors of $T_i^s \in V_1$; k^d additional scales;
$\left[N_2^d, N_3^d, ..., N_{k^d+1}^d\right]$ superpixels in each scale.

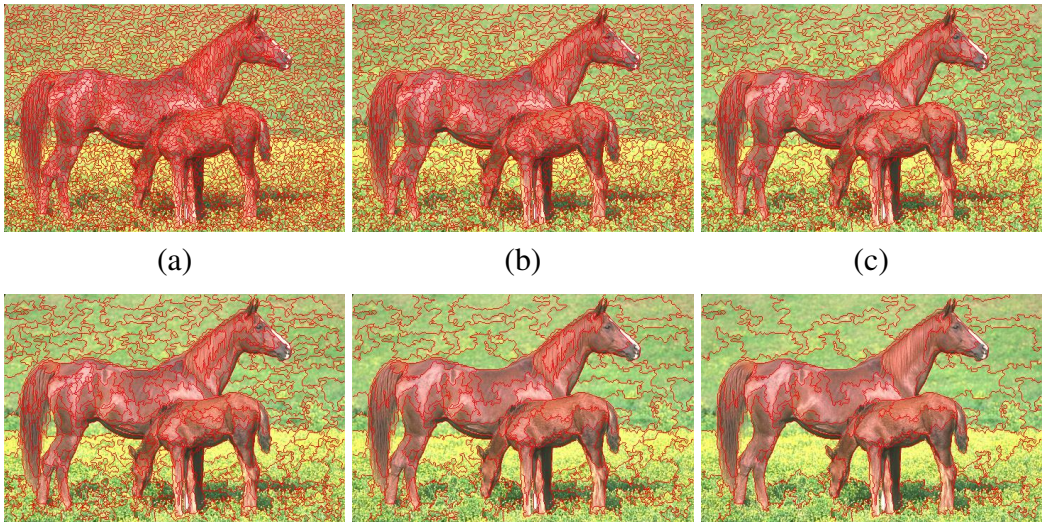
Output: All SAG's $G_s = (V_s, E_s)$ where $s = [2, .., k^d + 1]$.

Assign $s = 2$, $Q = 200$.
repeat
  Assign $G_s = G_{s-1}$ and $TAM_s = N_0/N_s^d$;
  Assign a label to each superpixel (vertex) T_i^s, $(i = 1, ..., N_s)$ in the SAG $G_s = (V_s, E_s)$; G_s has N_s superpixels (vertices);
  repeat
    Place all edges e_m in $E_s = \left\{e_m\left(T_i^s, T_j^s\right) | T_i^s \text{ and } T_j^s \text{ are adjacent vertices}\right\}$ in a priority queue P in ascending order of their color CIELAB Euclidean distances (see Eq. (4.1));
    while $P \neq \emptyset$ **do**
      Remove the edge $e_m = \left(T_i^s, T_j^s\right)$ and its associated superpixels T_i^s and T_j^s from the priority queue P;
      Compute $\{\tau_L, \tau_a, \tau_b\} \in [0, 1]$ for the $\{L, a, b\}$ channels using Eq. 4.2;
      Calculate the vertex contraction likelihood $\alpha\left(T_i^s, T_j^s\right)$ for T_i^s and T_j^s using Eq. (4.3);
      The vertices T_i^s and T_j^s are contracted if the vertices contraction predicate in Eq. (4.2) $C_{fs}(T_i^s, T_j^s) = 1$ and $size(T_i^s \cup T_j^s) \leq TAM_s$;
      If T_i^s and T_j^s are contracted, update the SAG G_s: replace the contracted vertices T_i^s and T_j^s by the new superpixel T_{ij}^s; remove self-loops and parallel edges; update the weighted mean color for T_{ij}^s as the mean CIELAB color of T_i^s and T_j^s; decrease the number of superpixels to process in scale s to $N_s = N_s - 1$;
    end
  until $(N_s == N_s^d)$;
  Save $SAG\ G_s$;
  Assign $s = s + 1$, $Q = Q/2$.
until $s == k^d + 1$;
Return all saved SAG $G_s = (V_s, E_s)$ for all scales $s = [2, .., k^d + 1]$.

ferent color representation, and the foreground object appears in Fig. 4.6(b) with just one superpixel. In the right column is important to notice the boundary correspondence and the nested scales representation by the hierarchical tree, where the superpixels in finer scales are contained in the superpixels at coarser scales.

Fig. 4.7 illustrates the results for the proposed stochastic methods (a) IHSGC, (b) SGC, and (c) HSGC for an image of the BSDS500 dataset from the finer to coarser

Figure 4.5: Example of the hierarchical stochastic graph contraction (HSGC) method for an image of the BSDS500 dataset. In the first row, from left to right, the results for 2500, 1000 and 500 superpixels are shown in (a), (b) and (c), respectively. In the second row are shown the results for the 250, 100 and 50 superpixels in (d), (e) and (f), respectively. The boundaries (red lines) in Figs. (a) to (f) show the evolution of the superpixels contraction.



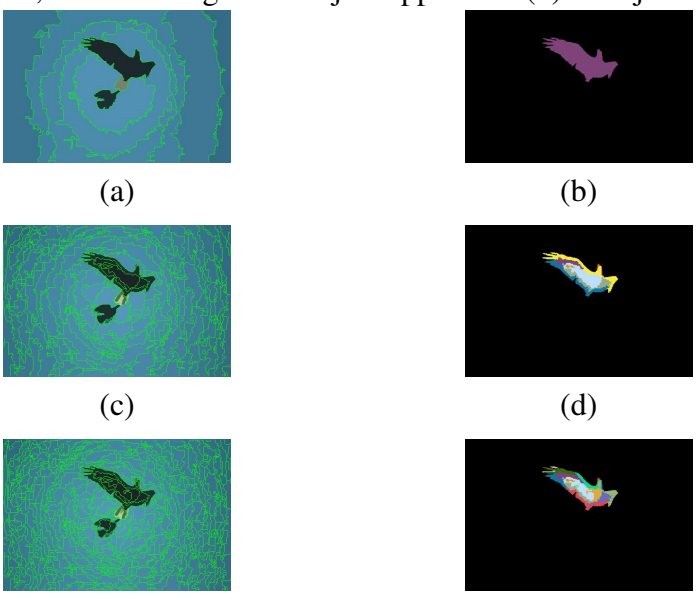
(a) (b) (c)

(d) (e) (f)

Source: The Author.

scales. From top to bottom are shown the results for 500, 1000 and 2500 superpixels. In this figure is possible to see the issues of the first proposed IHSGC method, which is very slow and generates small superpixels according to the texture of the regions in the image (rocks and sand in the figure), so these issues are solved using the SGC method to control the superpixel size and to generate more uniform superpixels in term of size with similar color properties. A problem with the SGC method is missing boundary correspondences between superpixels at different scales, which is solved by the HSGC method that generates several scales of representation with better runtime in comparison to IHSGC.

Figure 4.6: Illustration of the hierarchical representation obtained by the HSGC method for an image of the BSDS500 dataset from the finer to coarser scales. In the left column are shown the results for (a) 300, (c) 100 and (e) 10 superpixels. In the right column are shown the results for a multi-scale superpixel of the foreground object (i.e. the bird) using the hierarchical tree representation from coarser to fine scales. The boundaries (green lines) in Figs. (a), (c) and (e) show the evolution of the superpixels contraction. Figs. (b), (d) and (f) shows the foreground object with several smaller superpixels with different color representation, and the foreground object appears in (b) with just one superpixel.



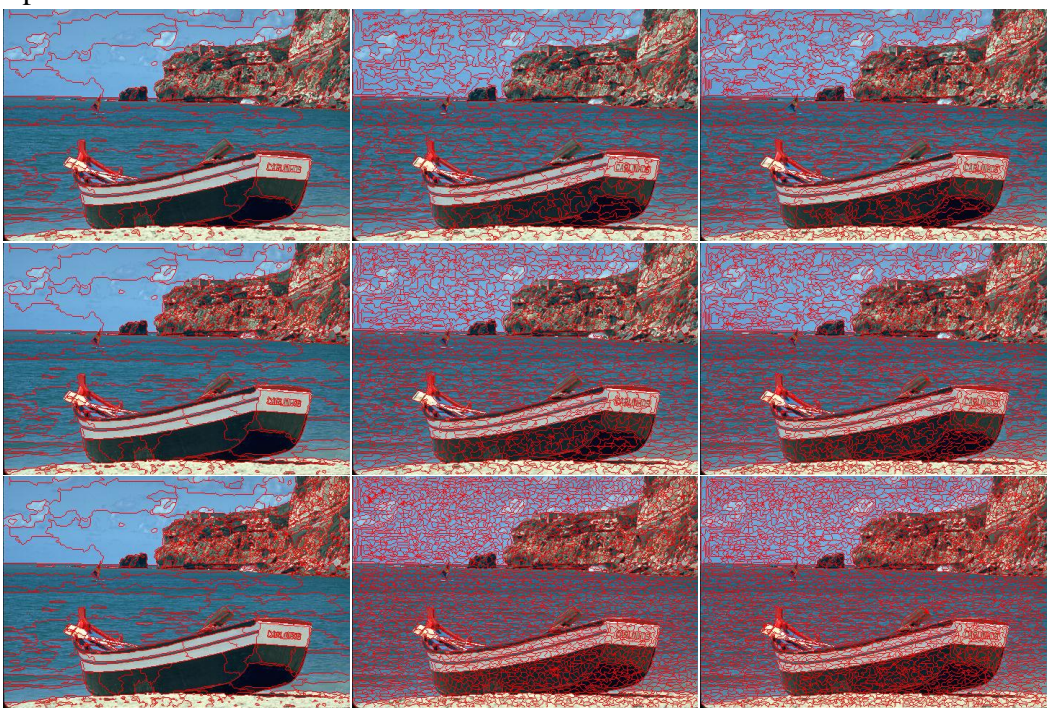
(a) (b)

(c) (d)

(e) (f)

Source: The Author.

Figure 4.7: Illustration of the hierarchical representation obtained by the stochastic methods: (a) IHSGC, (b) SGC, and (c) HSGC for an image of the BSDS500 dataset from the finer to coarser scales. From top to bottom are shown the results for 500, 1000 and 2500 superpixels.



(a) IHSGC (b) SGC (c) HSGC

Source: The Author.

5 PROPOSED METHODS FOR STOCHASTIC SPECTRAL SEGMENTATION

In this chapter, a spectral graph contraction (SSGC) approach is proposed, to perform the spectral image segmentation using eigen-decomposition of the superpixels-based normalized Laplacian matrix by applying a stochastic strategy.

The stochastic single-scale superpixel method SGC (4.2) was used as pre-processing step to help us to handle the scalability problem, and the similarity matrix of superpixels adjacency graph is built using a 3D normalized CIELAB histogram as local descriptor for each superpixel. Finally, a stochastic graph contraction strategy is applied over the K-eigenvectors of the superpixels-based normalized Laplacian matrix.

The distinctive characteristics of the proposed spectral image segmentation method are: i) uses stochastic superpixels as a pre-processing step; ii) uses simple data to compute the similarity matrix; and iii) defines stochastic graph contraction operations on an eigenvectors spanned sub-space, built with the superpixels spatial adjacency graph.

5.1 Stochastic Spectral Segmentation Approach Based on Graph Contractions

The proposed image segmentation approach, namely Stochastic Spectral Graph Contraction (SSGC) approach, uses stochastic superpixels as pre-processing step. These superpixels are obtained using the SGC method (see Section 4.2) and the initial superpixels are denoted by G, so they are the basic components for our spectral analysis.

Stochastic superpixels are obtained as a pre-processing step for the the proposed image segmentation approach, namely Stochastic Spectral Graph Contraction (SSGC) approach. These superpixels are obtained using the SGC method (see Section 4.2), and are the basic components of the SSGC image segmentation method.

Let $G = (V, E)$ be the obtained superpixels adjacency graph (SAG), where the set of vertices are superpixels $V = \{v_1, v_2, v_3, ..., v_{N_{sp}}\}$, and each edge $e(v_i, v_j)$ of the set of edges $E = \{e(v_i, v_j)|v_i, v_j \in V\}$ connect the superpixels v_i and v_j that are adjacent on the SAG G.

The SSGC method segments an input image by successively contracting the superpixels adjacency graph (SAG) $G = (V, E)$. In order to compute these SAG contractions, the SAG vertices data are projected into the space spanned by the eigenvectors of the normalized Laplacian matrix of the obtained SAG, which is updated at each contraction iteration.

In the SSGC method, the original image I is initially converted to CIELAB color space, and the SAG $G = (V, E)$ is obtained using the Algorithm 4. Next, each SAG vertex v_i (i.e. superpixel) is described by a 3D CIELAB normalized color histogram, and the similarities w_{ij} between all spatially adjacent SAG vertices v_i and v_j are computed to obtain the vertex similarity matrix W (see Equation 5.1). The spatial neighborhood of a superpixel consists of all spatially adjacent superpixels on the SAG, and $w_{ij} = 0$ is assigned to the non-adjacent superpixels.

The similarity between spatially adjacent superpixels is estimated based on the similarity of their pixels color distributions, and each superpixel is represented by a 3D CIELAB normalized color histogram due to its simplicity, and robustness to small scale changes (i.e. identical histograms are obtained for the same region in different scales). It shall be observed that a 3D CIELAB normalized color histogram H_i provides a rich appearance description for the superpixels contents as compared to local mean colors (GRUNDMANN et al., 2010). We use B bins for each dimension of the 3D CIELAB normalized color histograms (in our experiments $B = 24$), and each dimension represents a channel of the CIELAB color space. Given two normalized color histograms H_i and H_j for the superpixels v_i and v_j, respectively, their similarity w_{ij} is computed as follows :

$$w_{ij} = \exp\left(-\frac{(\chi^2(H_i, H_j))^2}{0.05}\right), \tag{5.1}$$

where H_i and H_j represent two histograms describing the contents of the SAG vertices (superpixels) v_i and v_j, respectively, and $\chi^2(H_i, H_j)$ is the chi-square distance between 3D CIELAB histograms calculated as follows:

$$\chi^2(H_i, H_j) = \frac{1}{2}\sum_{r=1}^{R}\sum_{m=1}^{M}\sum_{n=1}^{N}\frac{[H_i(r,m,n) - H_j(r,m,n)]^2}{H_i(r,m,n) + H_j(r,m,n)}, \tag{5.2}$$

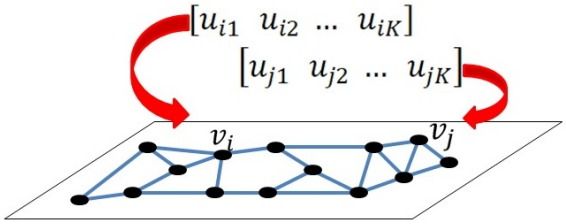
.

The similarities between the superpixel v_i and all other N_{sp} superpixels v_j, $i \neq j$, is a row $\{w_{ij}|j = 1, \ldots, N_{sp}\}$ of W (an N_{sp}-vector). Obtained the SAG similarity matrix W as indicated before, the Laplacian matrix normalized with respect to the graph vertices degrees d_{ii}, $L_{rw} = D^{-1}L = I - D^{-1}W$, is computed (See Section 2.3). Next, the eigen-decomposition $L_{rw}U = \lambda U$ is computed, where the eigenvectors are columns $\{u_{ij}|i = 1, \ldots, N_{sp}\}$ of the matrix U and λ_i are their corresponding eigenvalues, and the K smaller eigenvectors $\{u_{ij}|i = 1, \ldots, K\}$ span the $N_{sp} \times K$ sub-space in which the superpixels are represented and clustered. Figure 5.1 shows the spectral description of the

superpixels (i.e. SAG $G(V, E)$ vertices), where each SAG vertex $v_i \in V$ is described by a vector of spectral features $\{u_{i1}, ..., u_{iK}\}$ with $i = 1, .., N_{sp}$ corresponding to the $i - th$ row of the matrix U (see Figure 5.1).

As mentioned before, in order to segment an input image I, stochastic spectral contraction operations are applied on the SAG (superpixels adjacency graph) $G(V, E)$ to iteratively obtain new, contracted, versions of the SAG. At each contraction iteration of a SAG, a new SAG $G' = (V', E')$ is obtained with less vertices, until the desired K vertices are obtained in the final SAG. The final number of vertices K coincides with the number of dimensions of the sub-space used for spectral clustering (i.e. K corresponds to the number of eigenvectors spanning that sub-space) (SHI; MALIK, 2000). Also, it shall be observed that the final number K of superpixels (i.e. SAG vertices) coincides with the final number of regions obtained in the image segmentation.

Figure 5.1: Spectral representation for each superpixel as the vertex v_i of the SAG $G(V, E)$. Shows the spectral description of the superpixels (i.e. SAG $G(V, E)$ vertices), where each SAG vertex $v_i \in V$ is described by a vector of spectral features $\{u_{i1}, ..., u_{iK}\}$ with $i = 1, .., N_{sp}$ corresponding to the $i - th$ row of the matrix U.



As in the SGC superpixels generation step (see Section 4.2), a vertex contraction likelihood function $\beta(v_i, v_j)$ also is used here to decide if the superpixels v_i and v_j adjacent on the SAG should be contracted, refining the SAG $G(V, E)$ to obtain a new SAG $G'(V', E')$ with less vertices. As mentioned before, SSGC makes iterative graph contraction until the desired $K-$vertices (image regions) are obtained, as detailed next. The vertex contraction likelihood function $\beta(v_i, v_j)$ is evaluated to decide if the vertices v_i and v_j of the edge $e_m(v_i, v_j)$ should be contracted :

$$\beta(v_i, v_j) = \exp\left(-\frac{\sum_{k=1}^{k=K}(u_{ik} - u_{jk})^2}{\Lambda(v_i, v_j)}\right), \tag{5.3}$$

where Λ is a contraction penalty function defined as follows :

$$\Lambda(v_i, v_j) = \frac{\ln(N_0)}{Q}\left(\frac{1}{|v_i|} + \frac{1}{|v_j|}\right), \tag{5.4}$$

where N_0 is the number of pixels in the original image, $|v_i|$ is the number of elements

(pixels) in the vertex v_i, and Q is a regularization parameter that controls the superpixels clustering process. Larger values of Q results is more over-segmentation, and smaller values of Q results is more under-segmented. In our experiments, the initial value $Q = 200$ and it is updated using $Q = Q/2$ when is not possible contract the graph with previous Q values. The edges $e_m(v_i, v_j)$ with more similar vertices v_i and v_j are processed first, so the edges of the SAG $G(V, E)$ are processed in the ascending order of their weights w_{ij} (see Eq. 5.5), and whenever adjacent vertices are contracted the SAG $G(V, E)$ and the spectral representation of the superpixels U is updated. The edge weights w_{ij} used to sort the edges $e_m(v_i, v_j)$ in the ascending order for the stochastic spectral contraction operations, and the weights w_{ij} are computed based on the spectral features obtained for each vertex $\{u_{i1}, ..., u_{iK}\}$, $i = 1, .., N_{sp}$, as follows:

$$w_{ij} = \sqrt{\sum_{k=1}^{k=K} (u_{ik} - u_{jk})^2}. \tag{5.5}$$

It shall be observed that elements w_{ij} of W are computed as described in Eq. 5.5, so rigorously W is a dissimilarity matrix. However, the edges $e_m(v_i, v_j)$ are sorted in ascending of their weights w_{ij}, and the edges ordering goes from the most to the least similar edges, following a similarity order.

If $\tau \leq \beta(v_i, v_j)$, where $\tau \in [0, 1]$ is a random number, the vertices v_i and v_j are contracted, otherwise v_i and v_j are not contracted. This vertex contraction criterion is expressed by the contraction predicate $C_{spectral}(v_i, v_j)$ bellow :

$$C_{spectral}(v_i, v_j) = \left\{ \begin{array}{ll} 1 & , \text{ if } \beta(v_i, v_j) \geq \tau; \\ 0 & , \text{ otherwise .} \end{array} \right\}, \tag{5.6}$$

where the vertex contraction likelihood $\beta(v_i, v_j)$ is used to decide if the vertices should be contracted (See Eq. 5.3). If $\beta(v_i, v_j) \geq \tau$, with $\tau \subset [0, 1]$ and drawn from an uniform distribution, the vertices v_i and v_j are contracted, and the SAG $G(V, E)$ and the spectral representation of the superpixels U are updated, as mentioned before.

Algorithm 6 shows the pseudo-code of the SSGC algorithm used for color image segmentation. Its input is the SAG obtained by SGC algorithm (see Algorithm 4). Steps 2-20 perform the iterative SAG contractions. The superpixels-based similarity and degree matrices are obtained in step 3; the SAG normalized Laplacian matrix is computed in step 3; followed by the eigen-decomposition of L_{rw} in step 4. Each SAG vertex (superpixel) is described by its spectral values in the sub-space spanned by the K smaller eigenvectors

Algorithm 6: Stochastic Spectral Graph Contraction Algorithm (SSGC).

Input : Image I in CIELAB color space, superpixels adjacency graph $G = (V, E)$ (obtained by SGC superpixels method) and the desired number of image regions K.

Output: Superpixels Adjacency Graph (SAG) G' with K vertices and the image I' with K regions.

Assign $Q = 200$, $G'(V', E') = G(V, E)$ and let $N' = N_{sp}$;
while $N' > K$ **do**
  Compute the vertex similarity matrix $W' \in \mathbb{R}^{N_{sp} \times N_{sp}}$ (see Eq. 5.1) and the vertex degree matrix D' for the SAG G' with N' vertices;
  Compute the normalized Laplacian matrix L_{rw} of the SAG G';
  Find the K smaller eigenvectors of $L_{rw}U = \lambda U$, where the eigenvectors are columns $\{u_{ij} | i = 1, \ldots, K\}$ of $U \in \mathbb{R}^{N_{sp} \times K}$;
  Describe each vertex v_i with K spectral features $\{u_{i1}, ..., u_{iK}\}$ with $i = 1, .., N'$ (vector corresponding to the $i - th$ row of U);
  Place all edges $e'_m(v'_i, v'_j) \in E'$ in the priority queue P in the ascending order of their weights w'_{ij} (Eq. 5.5); only keep in P for further processing the edges with the 50% smaller weights w'_{ij};
  Define the empty set $R = \emptyset$ to contain the new SAG vertices obtained by SAG contraction;
  while $P \neq \emptyset$ **do**
    Remove the $e'_m(v'_i, v'_j)$ from the priority queue P;
    if $N' > K$ *and* $v'_i \notin R$ *and* $v'_j \notin R$ **then**
      if $\beta\left(v'_i, v'_j\right) \geq \tau$ **then**
        Update the SAG $G'(V', E')$ by defining a new SAG vertex v_{ij} and connect its edges to the edges linking v'_i and v'_j to their neighbors;
        Decrease the number of vertices $N' = N' - 1$;
        Update P to include v_{ij} and $R = R \cup \{v_{ij}\}$.
      end
    end
  end
  If $R = \emptyset$, then assign $Q = Q/2$.
end
Return $G'(V', E')$ with $N' = K$ vertices, and the image I' with K regions (K vertices of $G'(V', E')$).

in steps 5 and 6. In steps 7 and 8 the priority queue P and variables are set, and the SAG $G'(V', E')$ is contracted in the steps 9 to 18. The vertex contraction only occurs if the vertex contraction predicate in Equation 5.6 is satisfied, and this only occurs if the product of all K likelihood functions $\beta(v_i, v_j) \geq \tau$, where τ is a random number. In this case the spectral features are updated to include the new SAG vertex v_{ij}. Therefore, the proposed spectral image segmentation method can be interpreted as performing a random walk on the SAG, and merging the vertices v_i to v_j according to the transition probability matrix $P = (p_{ij}) = D^{-1}W$, for $i, j = 1, ..., N'$ (HASTIE; TIBSHIRANI; FRIEDMAN, 2009).

Let N_0 the number of pixels of the original image and N_{sp} the number of superpixels obtained by the SGC method, where $N_0 > N_{sp}$. The complexity of the SSGC step is proportional to the sum of the complexities to obtain the superpixels, the complexity of the eigen-decomposition for the SAG with N_{sp} superpixels, and the complexity to perform the stochastic spectral graph contractions. Usually $N_0 >> N_{sp}$, and if the graph edges are processed in ascending order of their weights (i.e. using Euclidean distances of their K-dimensional representations), then worst-case running time for the SSGC algorithm is $\mathcal{O}(N_0(\log N_0))$ + $\mathcal{O}(r * N_{sp}^3)$, where r is a number of iteration to rebuilt the similarity matrix including the new vertices (in our experiments, starting with 120 superpixels approximately 10 iterations are needed to obtain the image segmentation with a number of image regions between 5 and 20).

An efficient approximation of the SSGC method, less computational intensive, namely $SSGC_E$, is presented in Section 5.2. Nevertheless, its improved computational efficiency comes at the expenses of slightly worse performance or the BSDS300 and BSDS500 datasets, as indicated by the segmentation quality measures.

5.2 A Computationally Efficient Approximation of the SSGC Method ($SSGC_E$)

A computationally efficient approximation of the SSGC algorithm, namely $SSGC_E$, used for color image segmentation is presented in Algorithm 7. Its input is the SAG obtained by SGC algorithm (see Algorithm 4). Then, the superpixels-based similarity and degree matrices are obtained in step 1; the SAG normalized Laplacian matrix is computed in step 2; followed by the eigen-decomposition of L_{rw} in step 3. Each SAG vertex (superpixel) is described by its spectral values in the sub-space spanned by the K smaller eigenvectors in step 4. In steps 5 some variables are defined, and the SAG $G'(V', E')$ is

contracted in the steps 6 to 16. The vertex contraction only occurs if the vertex contraction predicate in Eq. 5.6 is satisfied, and this condition is satisfied if the product of all K likelihood functions $\beta_c(v_i, v_j) \geq \tau$, where τ is a random number, and $C_{spectral}$ and β_k are defined as follows :

$$C_{spectral}(v_i, v_j) = \begin{cases} 1 & \text{, if } \prod_{k=1}^{k=K} \beta_k(v_i, v_j) \geq \tau; \\ 0 & \text{, otherwise ,} \end{cases} \tag{5.7}$$

and,

$$\beta_k(v_i, v_j) = \exp\left(\frac{-(u_{ik} - u_{jk})^2}{\Lambda(v_i, v_j)}\right), \tag{5.8}$$

where Λ is defined as in Eq. 5.4.

In this computationally efficient version of the SSGC algorithm (i.e. $SSGC_E$) the spectral features $u_{(ij)k}$ are updated after each SAG contraction when the new vertex v'_{ij} was obtained by computing the average of u_{ik} and u_{jk} weighted by the respective vertices sizes $size(v_i)$ and $size(v_j)$. Therefore, the spectral values of the new vertex v'_{ij} are obtained as indicated in Eq. 5.9, and afterwards are normalized as shown in Eq. 5.10.

$$u_{(ij)k} = \frac{size(v_i) \cdot u_{ik} + size(v_j) \cdot u_{jk}}{size(v_i) + size(v_j)}, \tag{5.9}$$

with $k = 1, .., K$. The spectral features are normalized as follows :

$$u_{(ij)k} = \frac{u_{(ij)k}}{\sum_{k=1}^{k=K}(u_{(ij)k})}, \tag{5.10}$$

and the new edges $e'_m v'_{ij}, v'_n$ weights $w'_{ij}n$ also are updated based on the new vertex estimated spectral features as follows :

$$w'_{(ij)n} = \sqrt{\sum_{k=1}^{k=K} \left(u_{(ij)k} - u_{nk}\right)^2}. \tag{5.11}$$

For example, let us suppose the vertices v_i and v_j of the edge $e_1 = (v_i, v_j)$ are merged generating the new vertex v_{ij}, and the edges still remaining in the priority queue P ordered for further processing are $P = \{e_2 = (v_p, v_q), e_3 = (v_s, v_i), e_4 = (v_t, v_x), e_5 = (v_z, v_j), e_6 = (v_j, v_n)\}$. Therefore, the remaining edges in priority queue P are updated to include the new vertex v_{ij} to $P = \{e_2 = (v_p, v_q), e_3 = (v_s, v_{ij}), e_4 = (v_t, v_x), e_5 = (v_z, v_{ij}), e_6 = (v_{ij}, v_n)\}$, and the new edges weights are updated using Eq. 5.11, and edges continue being processed based on their ordering in P. Therefore, in the example above the next in P edge to be processed would be $e_2 = (v_p, v_q)$, then $e_3 = (v_s, v_{ij})$, and so on.

Consequently, when evaluating if the vertices of

$$e_2 = (v_p, v_q)$$

should be merged, the up to date spectral features u_{ip} and u_{iq} are used to compute the vertices similarity (see Eq. 5.5), which is used in the evaluation of vertices contraction predicate $C_{spectral}(v_p, v_q)$ (see Equation (5.6)).

While the priority queue P is not empty the SAG edges $e_m v_i, v_j$ are processed in the ascending order of their current weights. However, when the priority queue P is empty and the number of image regions K has not been reached yet, the normalized Laplacian matrix L_{rw} is re-computed for the new SAG G', obtaining the new vertices spectral features, and the edges in the priority queue P are processed as described before. These iterations will continue until the number of image regions K is reached.

Let N_0 the number of pixels of the original image and N_{sp} the number of superpixels obtained by the SGC method, where $N > N_{sp}$. The complexity of the $SSGC_E$ method is proportional to the sum of the complexities to obtain the superpixels, the complexity of the eigen-decomposition for the SAG with N_{sp} superpixels, and the complexity to perform the stochastic spectral graph contractions. Usually $N_0 >> N_{sp}$, and if the graph edges are processed in ascending order of their weights (i.e. using Euclidean distances of their K-dimensional representations), then worst-case running time for the $SSGC_E$ algorithm is $\mathcal{O}(N_0(\log N_0))$ + $\mathcal{O}(N_{sp}^3)$.

Algorithm 7: Computationally Efficient Stochastic Spectral Graph Contraction Algorithm ($SSGC_E$).

Input : Image I in CIELAB color space, superpixels adjacency graph $G = (V, E)$ (obtained by SGC superpixels method) and the desired number of image regions K.

Output: Superpixels Adjacency Graph (SAG) G' with K vertices and the image I' with K regions.

Let N_{sp} the number of vertices in the SAG $G(V, E)$, compute the vertex similarity matrix $W \in \mathbb{R}^{N_{sp} \times N_{sp}}$ (see Equation 5.1) and vertex degree matrix D;

Compute the normalized Laplacian matrix L_{rw} of the SAG $G(V, E)$;

Find the K smaller eigenvectors of $L_{rw}U = \lambda U$, where the eigenvectors are columns $\{u_{ij} | i = 1, \ldots, K\}$ of the matrix $U \in \mathbb{R}^{N_{sp} \times N_{sp}}$;

Describe each superpixel v_i with K spectral features $\{u_{i1}, ..., u_{iK}\}$ with $i = 1, .., N_{sp}$ (vector corresponding to the $i - th$ row of U);

Assign $Q = 200$, $G'(V', E') = G(V, E)$ and let $N' = N_{sp}$;

repeat

  Place all edges $e'_m(v'_i, v'_j) \in E'$ in the priority queue P based on the ascending order of their weights w'_{ij} (descending order of the similarity between v'_i and v'_j, see Equation 5.5;

  while $P \neq \emptyset$ **do**

    Remove the $e'_m(v'_i, v'_j)$ from the priority queue P;

    Generate the value $\tau \subset [0, 1]$ from an uniform distribution;

    Calculate the vertex contraction likelihoods $\beta_k \left(v'_i, v'_j\right)$ using Equation 5.8;

    Decide if the vertices v'_i and v'_j should be contracted using the predicate $C_{spectral}(v_i, v_j)$ (see Equation (5.7));

    If the vertices v'_i and v'_j are contracted obtaining v_{ij}, and update the SAG $G'(V', E')$. Update the spectral features for the new vertex v'_{ij} using Eq. 5.9, normalize the row sums of U to 1 using Eq. 5.10, update the weight of w'_{ij} using Eq. 5.11 and decrease the number of regions $N' = N' - 1$.

  end

  Assign $Q = Q/2$;

until $N' == K$;

Return $G'(V', E')$ with $N' = K$ vertices, and the image I' with K regions (K vertices of $G'(V', E')$).

6 EXPERIMENTAL RESULTS

In this chapter, our stochastic proposed methods for superpixels generation and spectral segmentation are evaluated qualitatively and quantitatively using standard benchmarks.

6.1 Experimental Results: Stochastic Superpixels

Our proposed superpixels methods IHSGC (see Section 4.1), SGC (see Section 4.2) and HSGC (see Section 4.2) are evaluated qualitatively and quantitatively, and compared with other superpixels approaches that are representative of the state-of-the-art. The comparative methods are: Simple Linear Iterative Clustering Superpixels (SLIC) (ACHANTA et al., 2012), Linear Spectral Clustering superpixels (LSC) (LI; CHEN, 2015), Watershed superpixels (SCoW) (HU; ZOU; LI, 2015), Simple Non-Iterative Clustering (SNIC) (ACHANTA; SUSSTRUNK, 2017), Fuzzy simple linear iterative clustering (FSLIC) (WU; ZHANG L .AND ZHANG; YAN, 2019) and Superpixel Hierarchy (SH) (WEI et al., 2018). These algorithms were selected because they are quite popular, provide low computational complexity, and perform well in terms of boundary adherence and under-segmentation error.

This method comparison is based on the benchmark proposed in (NEUBERT; PROTZEL, 2012), which uses the publicly available datasets BSDS300 and BSDS500 (ARBELAEZ et al., 2011) to measures the boundary recall and under-segmentation error from the superpixels algorithms (see Section 2.4. The BSDS300 dataset consists of 300 natural color images with 481×321 or 321×481 pixels each, divided into 200 images for training and 100 images for testing. The BSDS500 dataset consists of 500 natural color images with 481×321 or 321×481 pixels each, divided into 200 images for training, 100 images for evaluation and 200 images for testing, and this dataset is more complex than BSDS300. The experimental results are obtained by using only the test images set provided in both datasets.

The proposed method IHSGC was implemented in MATLAB on a PC compatible computer, with an Intel Core i7 3.33 GHz processor and 16 GB RAM. The average runtime to obtain the superpixels for the images in BSDS datasets is about $320s$ for 7 scales (i.e. scales with $25, 50, 100, 250, 500, 1000$ and 2500 superpixels). When running IHSGC, most of the processing time is spent on the 3D histogram construction and sorting of the

SAG edges.

The proposed methods SGC and HSGC were implemented in MATLAB on a PC compatible computer, with an Intel Core i3 2.13 GHz processor and 8 GB RAM. The average runtime to obtain the superpixels for the images in BSDS datasets is about $2s$ for single scale (SGC) or $6s$ for 7 scales using the HSGC (i.e. scales with $25, 50, 100, 250, 500, 1000$ and 2500 superpixels).

Figures 6.1 and 6.2 illustrate some results obtained by the proposed single scale superpixels method SGC and by comparative approaches for 3 images of the BSDS300 and the BSDS500 data sets, when the number of superpixels is changed. In Figures 6.1 and 6.2 the green lines overlaid on the original images indicate the superpixels boundaries, and the columns show the results obtained by the compared methods, left to right : SLIC (the parameter controlling the compactness of superpixels was set to $m = 20$) (ACHANTA et al., 2012); LSC (the ratio parameter $r_c = C_s/C_c$ was set to 0.075, where C_s and C_c are the spatial and color similarities) (LI; CHEN, 2015); SCoW (the parameter controlling the spatial constraint was set to $\lambda = 0.75$) (HU; ZOU; LI, 2015); SNIC (its main parameter was set to $m = 20$) (ACHANTA; SUSSTRUNK, 2017); Fuzzy SLIC (FSLIC) (its main parameter was set to $m = 20$) (WU; ZHANG L .AND ZHANG; YAN, 2019); and the proposed SGC (see Section 4.2), respectively. Each image appears twice, in two rows, and the top three rows show the results obtained with approximately 1000 superpixels for each tested method, and the three bottom rows show the results obtained with approximately 100 superpixels for each compared method.

Figs. 6.3 and 6.4 show the quantitative results obtained for boundary recall (left) and under-segmentation error (right) using the BSDS300 and BSDS500 datasets, respectively. Also Figure 6.5 shows a comparison of the runtimes in seconds obtained for the BSDS300 (left) and the BSDS500 (right) data sets. The methods were compared using the following parameter settings : $m = \{10, 20\}$ for SLIC; $r = 0.075$ for LSC; $\lambda = 0.75$ for SCoW, $m = \{10, 20\}$ for SNIC; $m = \{10, 20\}$ for FSLIC; and the proposed SGC used the settings specified in Section 4.2. From the coarser to the finer scales, the number of superpixels per scale used in this runtime comparison are approximately (or equal) to : $25, 50, 100, 250, 500, 1000$ and 2500 superpixels. The authors codes in C++ were used to run the comparative methods, and the proposed method was implemented and tested in MATLAB.

The multi-scale superpixels obtained for 3 images of the BSDS300 data set are illustrated in Figs. 6.6, 6.8 and 6.10. Also, the multi-scale superpixels obtained for 3

images of the BSD500 data set are illustrated in Figs. 6.7, 6.9 and 6.11. These results were obtained for different methods, and compared for different numbers of superpixels per image, and the results for 25, 50, 100, 250, 500, 1000 and 2500 superpixels are shown from the top to the bottom of the image plates, respectively. The superpixels boundaries are shown in the red lines overlaid on the original images, and the results obtained for the following compared methods and parameter settings are shown along the columns, left to right: LSC (with the ratio parameter $r = C_s/C_c = 0.075$, where C_s and C_c are the spatial and color similarities) (LI; CHEN, 2015); SNIC (with the parameter $m = 5$) (ACHANTA; SUSSTRUNK, 2017), Fuzzy SLIC (FSLIC) (with parameter $m = 5$) (WU; ZHANG L .AND ZHANG; YAN, 2019), SH (with the parameter that balances between color and edge features set to 0.5) (WEI et al., 2018), and the proposed IHSGC and HSGC, respectively.

The comparative quantitative evaluation of the proposed methods for the BSDS300 data set are shown in Figure 6.12 using the boundary recall (left) and under-segmentation error (right), respectively. This comparison is based on 7 scales, and includes multi-scale methods like IHSGC, SH, HSGC, and single-scale methods that were configured to obtain their best boundary adherence in the 7 scales. The number of superpixels per scale (exact or approximate, depending on the method), are the following from the coarser to the finer scales : $25, 50, 100, 250, 500, 1000$ and 2500. The methods compared and their parameters settings are : the proposed SGC, HSGC (see Section 4.2) and IHSGC; SLIC with the parameter $m = 5$; SCoW with parameter $\lambda = 0.75$; LSC with parameter $r = 0.075$, SNIC with parameter $m = 5$, SH with the default parameter 0.5, and FSLIC with parameter $m = 5$, respectively. Similarly, the quantitative evaluation of the compared methods in terms of boundary recall and under-segmentation error obtained for the BSDS500 data set are shown in Figure 6.13. It can be seen that HSGC tends to provide the best boundary recall results, followed by SCG, with a relatively small under-segmentation error, also followed by SCG, for all 7 scales in the tested data sets.

It shall be observed that the boundary recall and under-segmentation error curves in the Figures 6.12, and 6.13 have 7 points in each curve, and each point represents a scale containing a number of superpixels approximately equal to $25, 50, 100, 250, 500, 1000$ and 2500, from the coarser to the finer scales. The four methods SGC, HSGC, IHSGC and SH obtained the exact number of superpixels per scale indicated above, but SLIC, SCoW, LSC, SNIC and FSLIC obtained an approximation of specified number of superpixels per scale (see the number of superpixels in the horizontal axis). The multi-scale methods

HSGC, IHSGC and SH obtained the 7 scales in one run. Each one of the single-scale methods (SGC, SLIC, SCoW, LSC, SNIC and FSLIC) needed 7 distinct runs with the same parameters settings (e.g. m, λ, r) to obtain the 7 scales, each run with the specified number of superpixels for that particular scale. These experiments indicate that the runtimes of SGC and HSGC tend to decrease with the number of superpixels, since less iterations are needed to obtain smaller superpixels (or yet, a larger number of superpixels, but the implementations in MATLAB were not optimized leading to higher runtimes.

Figs. 6.14, show the quantitative results obtained for runtime metric in seconds using the BSDS300 (left) and BSDS500 (right) datasets, including multi-scale methods for seven scales results ($25, 50, 100, 250, 500, 1000$ and 2500). These Figs. include the tested methods: SLIC with parameter $m = 5$, LSC with $r = 0.075$, SCoW with $\lambda = 0.75$, SNIC with $m = 5$, FSLIC with $m = 5$, SH with default parameter 0.5, and the proposed methods SGC, HSGC (see Section 4.2) and IHSGC (see Section 4.1), respectively. For the state-of-the-art tested method we use the author's code in C++, and our proposed methods were developed in MATLAB.

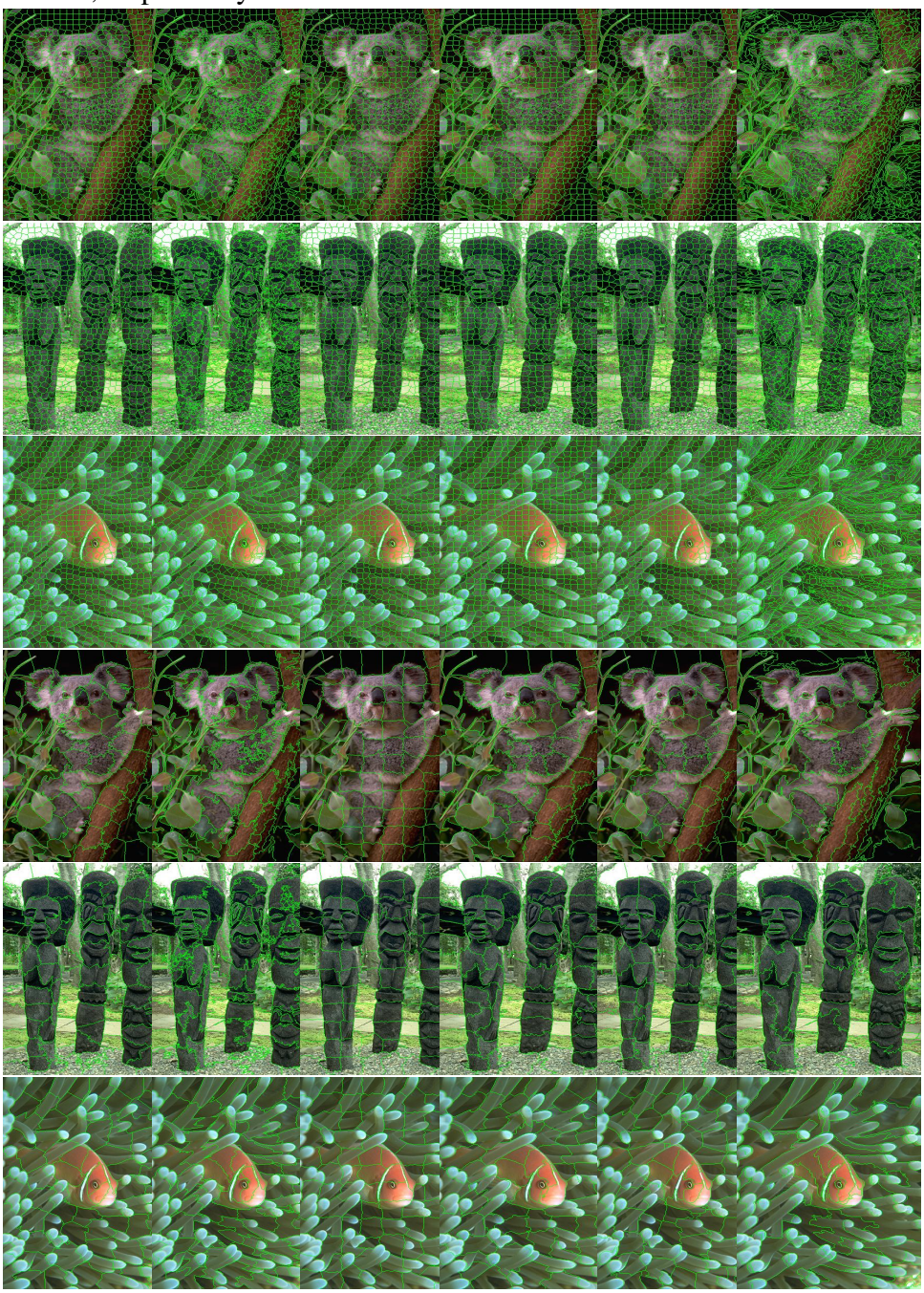
6.1.1 Discussion and Ablation

As mentioned before, the quantitative evaluation uses two standard superpixels metrics as criteria to evaluate the comparative methods, namely, boundary recall and under-segmentation error (NEUBERT; PROTZEL, 2012) (see Section 2.4). Using these evaluation metrics, we can know how the superpixels boundaries are aligned with the image objects boundaries. The objects boundaries are represented in the ground truth, and were obtained by human labeled segmentation.

The performances of all compared methods were evaluated based on the same number of superpixels, within the range $[25, 2500]$ superpixels per image/scale. It can be verified that the proposed approaches SGC and HSGC tend to perform better than the comparative state-of-the-art superpixels methods. Also, the proposed approaches IHSGC and HSGC can provide a multi-scale representation that describes the image contents at different levels of detail. Additionally, the proposed methods can get the desired number of superpixels by controlling the vertex contraction operations.

Also, it can be seen in Figures 6.1 and 6.2 that the proposed single-scale SGC method obtains superpixels with better boundary adherence and more homogeneous in color than the comparative methods : SLIC (ACHANTA et al., 2012), LSC (LI; CHEN,

Figure 6.1: Obtained Superpixels for three images of the BSDS300 dataset. The green lines overlaid on the original images indicate the superpixels boundaries, and the columns show the results obtained by the compared methods, left to right : (a) SLIC ($m = 20$) (ACHANTA et al., 2012); (b)LSC ($r_c = 0.075$) (LI; CHEN, 2015); (c) SCoW ($\lambda = 0.75$) (HU; ZOU; LI, 2015); (d) SNIC ($m = 20$) (ACHANTA; SUSSTRUNK, 2017); (e) Fuzzy SLIC (FSLIC) ($m = 20$) (WU; ZHANG L .AND ZHANG; YAN, 2019); and (f) the proposed SGC, respectively.

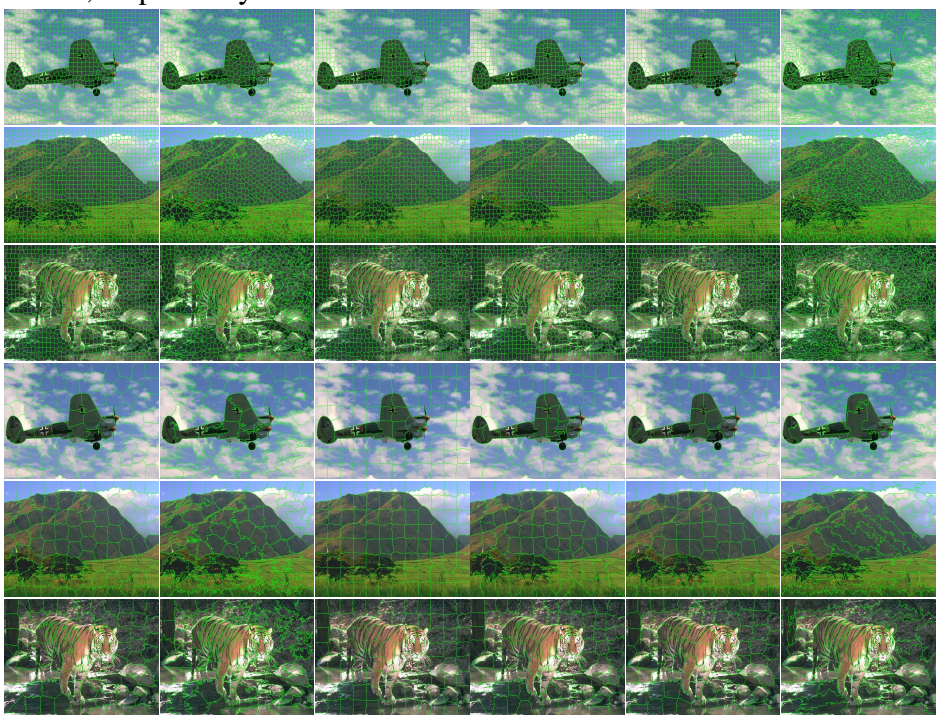


(a) SLIC (b) LSC (c) SCoW (d) SNIC (e) FSLIC (f) SGC

Source: The Author.

2015), SCoW (HU; ZOU; LI, 2015), SNIC (ACHANTA; SUSSTRUNK, 2017) and FSLIC (WU; ZHANG L .AND ZHANG; YAN, 2019). Possibly because SGC tends to

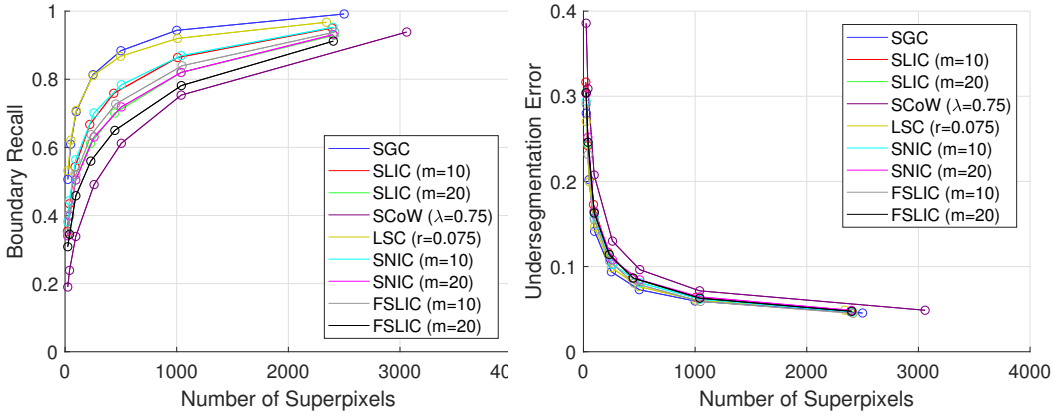
Figure 6.2: Obtained Superpixels for three images of the BSDS500 dataset. The green lines overlaid on the original images indicate the superpixels boundaries, and the columns show the results obtained by the compared methods, left to right : (a) SLIC ($m = 20$) (ACHANTA et al., 2012); (b)LSC ($r_c = 0.075$) (LI; CHEN, 2015); (c) SCoW ($\lambda = 0.75$) (HU; ZOU; LI, 2015); (d) SNIC ($m = 20$) (ACHANTA; SUSSTRUNK, 2017); (e) Fuzzy SLIC (FSLIC) ($m = 20$) (WU; ZHANG L .AND ZHANG; YAN, 2019); and (f) the proposed SGC, respectively.



(a) SLIC (b) LSC (c) SCoW (d) SNIC (e) FSLIC (f) SGC

Source: The Author.

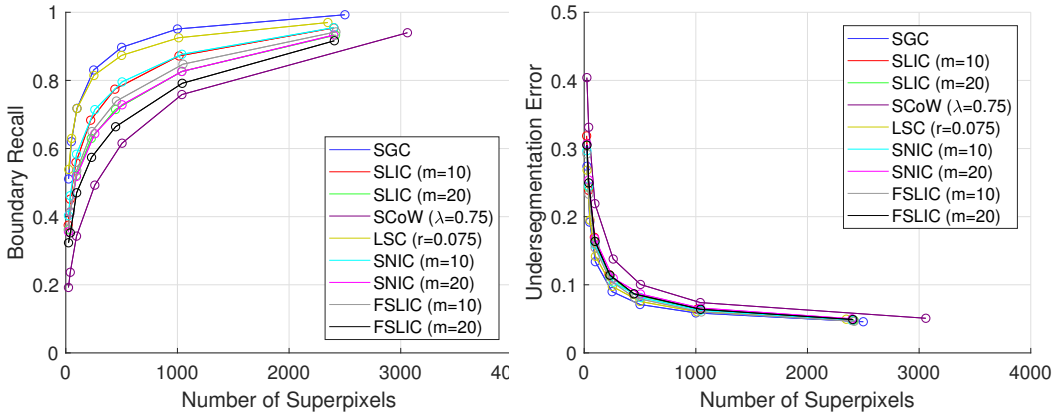
Figure 6.3: Benchmark metric values obtained for SGC method and superpixels comparative methods using (a) Boundary Recall and (b) Under-segmentation error for the test set images of the BSD300 dataset.



(a) BR (b) UE

Source: The Author.

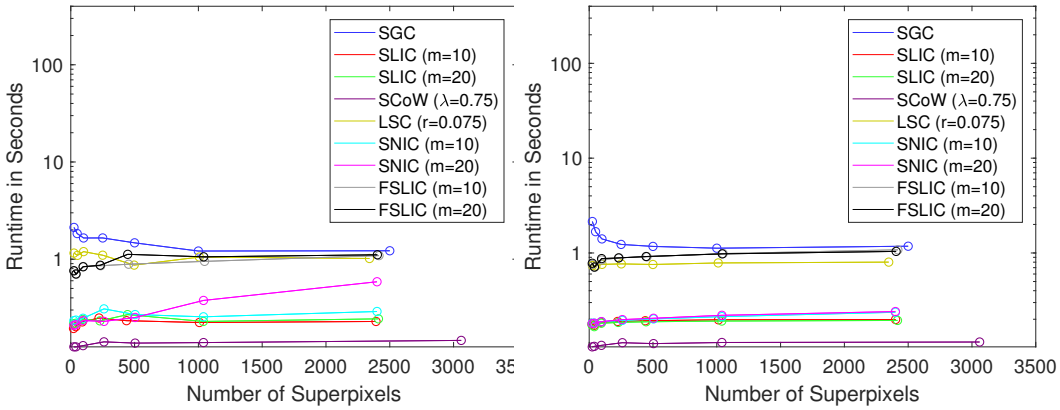
Figure 6.4: Benchmark metric values obtained for SGC method and superpixels comparative methods using (a) Boundary Recall and (b) Under-segmentation error for the test set images of the BSDS500 dataset.



(a) BR (b) UE

Source: The Author.

Figure 6.5: Runtime metric for SGC method and superpixels comparative methods using (a) BSD300 and (b) BSDS500 datasets.



(a) Runtime BSDS300 (b) Runtime BSDS500

Source: The Author.

Figure 6.6: Multi-scale superpixels results for one image of the BSDS300 dataset with seven scales for each tested method. These results were obtained for different numbers of superpixels and are shown from the top to the bottom of the image plates: 25, 50, 100, 250, 500, 1000 and 2500 superpixels, respectively. The superpixels boundaries are shown in the red lines overlaid on the original images, and the results obtained for the following compared methods: (a) proposed IHSGC, (b) proposed HSGC, (c) SH (WEI et al., 2018), (d) LSC (LI; CHEN, 2015), (e) SNIC (ACHANTA; SUSSTRUNK, 2017), (e) Fuzzy SLIC (FSLIC) (WU; ZHANG L .AND ZHANG; YAN, 2019), respectively.

(a) IHSGC(b) HSGC (c) SH (d) LSC (e) SNIC (f) FSLIC

Source: The Author.

generate more irregular superpixels, which facilitates obtaining an improved boundary adherence than the comparative methods even at coarser scales, which may be more relevant when a small number of superpixels is needed. Figures 6.3(left) and 6.4(left) show that SGC tends to achieve better boundary recall than the comparative methods. Also, Figures 6.3(right) and 6.4(right) show that SGC tends to achieve a better under-segmentation error than the comparative methods.

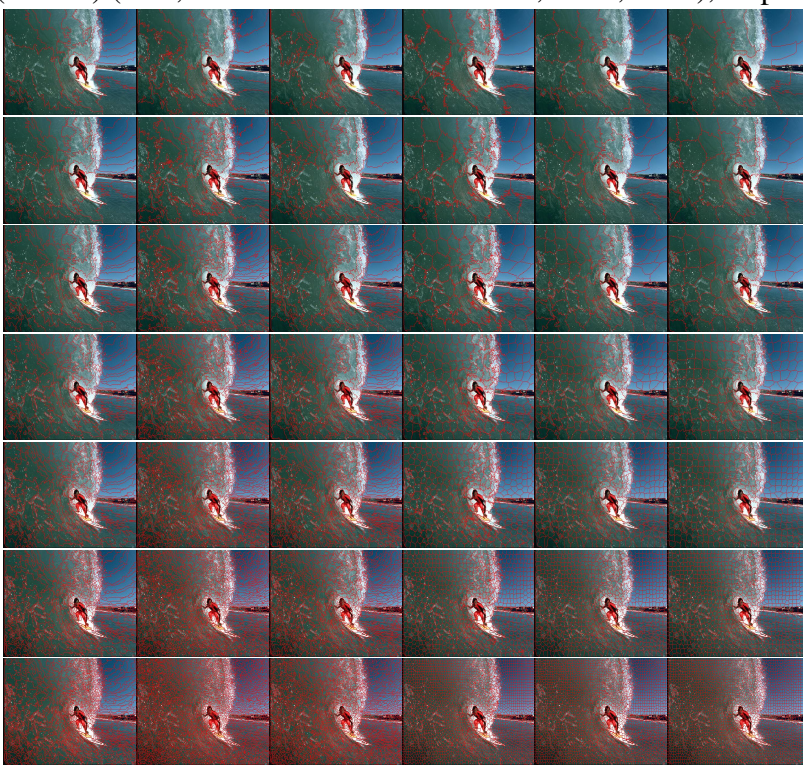
The proposed multi-scale IHSGC and HSGC methods provide nested hierarchical superpixels. In this case, the pixels in each superpixel at a scale s also belong to a parent superpixel at scale $s + 1$, forming hierarchical trees of superpixels. Therefore, the relationship between multi-scale superpixels is known, and images can be represented hierarchically using these multi-scale superpixels, as illustrated in Figures 6.6, 6.8, 6.10, 6.7, 6.9 and 6.11. It shall be observed that most superpixels methods available are sin-

Figure 6.7: Multi-scale superpixels results for one image of the BSDS500 dataset with seven scales for each tested method. These results were obtained for different numbers of superpixels and are shown from the top to the bottom of the image plates: 25, 50, 100, 250, 500, 1000 and 2500 superpixels, respectively. The superpixels boundaries are shown in the red lines overlaid on the original images, and the results obtained for the following compared methods: (a) proposed IHSGC, (b) proposed HSGC, (c) SH (WEI et al., 2018), (d) LSC (LI; CHEN, 2015), (e) SNIC (ACHANTA; SUSSTRUNK, 2017), (e) Fuzzy SLIC (FSLIC)) (WU; ZHANG L .AND ZHANG; YAN, 2019), respectively.

(a) IHSGC(b) HSGC (c) SH (d) LSC (e) SNIC (f) FSLIC

Source: The Author.

Figure 6.8: Multi-scale superpixels results for one image of the BSDS300 dataset with seven scales for each tested method. These results were obtained for different numbers of superpixels and are shown from the top to the bottom of the image plates: 25, 50, 100, 250, 500, 1000 and 2500 superpixels, respectively. The superpixels boundaries are shown in the red lines overlaid on the original images, and the results obtained for the following compared methods: (a) proposed IHSGC, (b) proposed HSGC, (c) SH (WEI et al., 2018), (d) LSC (LI; CHEN, 2015), (e) SNIC (ACHANTA; SUSSTRUNK, 2017), (e) Fuzzy SLIC (FSLIC) (WU; ZHANG L .AND ZHANG; YAN, 2019), respectively.



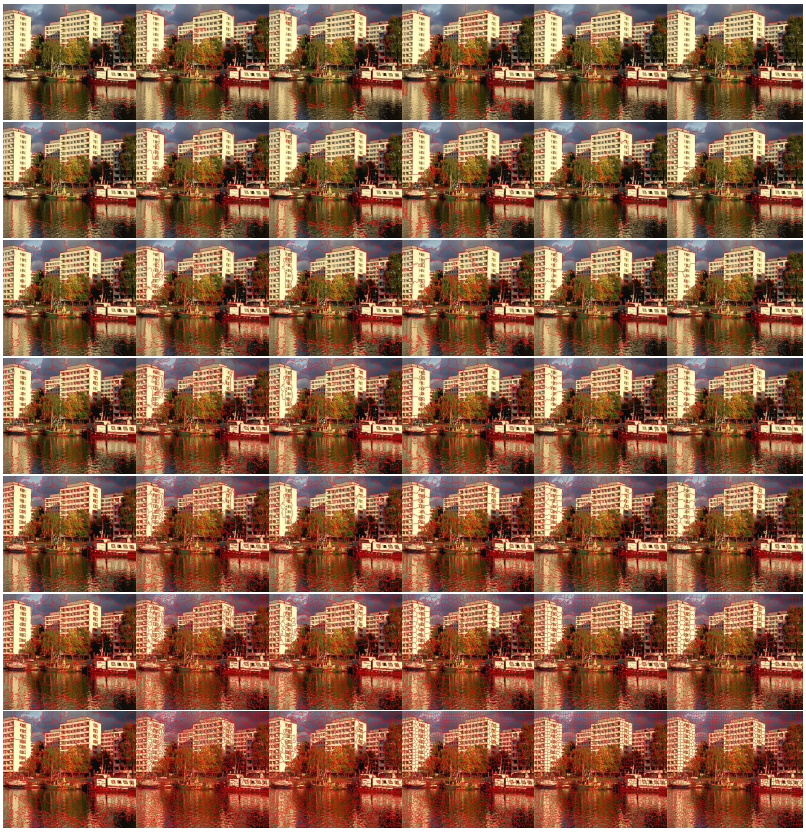
(a) IHSGC(b) HSGC (c) SH (d) LSC (e) SNIC (f) FSLIC

Source: The Author.

gle scale, and their parameter settings must be modified to obtain superpixels at different scales (with different number of superpixels per scale). Consequently, the correspondence between the superpixels boundaries at different scales tends to be worse than the boundary correspondences across different scales provided by the proposed HSGC. In the case of SLIC, LSC, SCoW, SNIC and FSLIC, an iterative process can be used to update the pixels clusters, without necessarily obtaining a good correspondence between the superpixels boundaries in adjacent scales.

Therefore, the proposed stochastic superpixels methods potentially allows to represent the image contents in multiple scales with good boundary adherence and the exact number of homogeneous superpixels at all scales, improving on state-of-the-art. The above mentioned properties were validated qualitatively (i.e. using visual analysis) and quantitatively (i.e. using two standard objective criteria, namely, boundary recall and

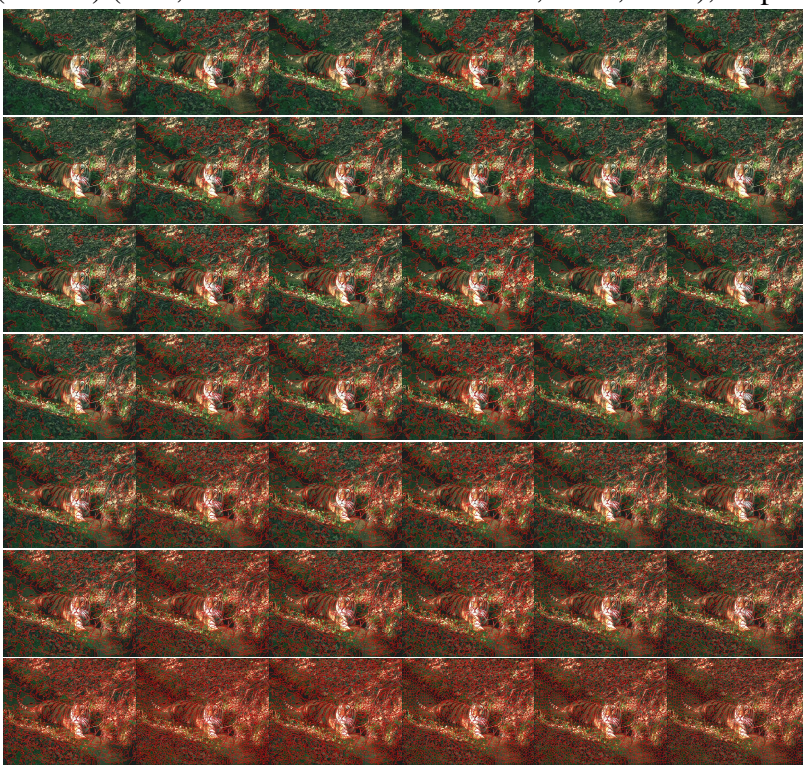
Figure 6.9: Multi-scale superpixels results for one image of the BSDS500 dataset with seven scales for each tested method. These results were obtained for different numbers of superpixels and are shown from the top to the bottom of the image plates: 25, 50, 100, 250, 500, 1000 and 2500 superpixels, respectively. The superpixels boundaries are shown in the red lines overlaid on the original images, and the results obtained for the following compared methods: (a) proposed IHSGC, (b) proposed HSGC, (c) SH (WEI et al., 2018), (d) LSC (LI; CHEN, 2015), (e) SNIC (ACHANTA; SUSSTRUNK, 2017), (e) Fuzzy SLIC (FSLIC) (WU; ZHANG L .AND ZHANG; YAN, 2019), respectively.



(a) IHSGC(b) HSGC (c) SH (d) LSC (e) SNIC (f) FSLIC

Source: The Author.

Figure 6.10: Multi-scale superpixels results for one image of the BSDS300 dataset with seven scales for each tested method. These results were obtained for different numbers of superpixels and are shown from the top to the bottom of the image plates: 25, 50, 100, 250, 500, 1000 and 2500 superpixels, respectively. The superpixels boundaries are shown in the red lines overlaid on the original images, and the results obtained for the following compared methods: (a) proposed IHSGC, (b) proposed HSGC, (c) SH (WEI et al., 2018), (d) LSC (LI; CHEN, 2015), (e) SNIC (ACHANTA; SUSSTRUNK, 2017), (e) Fuzzy SLIC (FSLIC) (WU; ZHANG L .AND ZHANG; YAN, 2019), respectively.



(a) IHSGC(b) HSGC (c) SH (d) LSC (e) SNIC (f) FSLIC

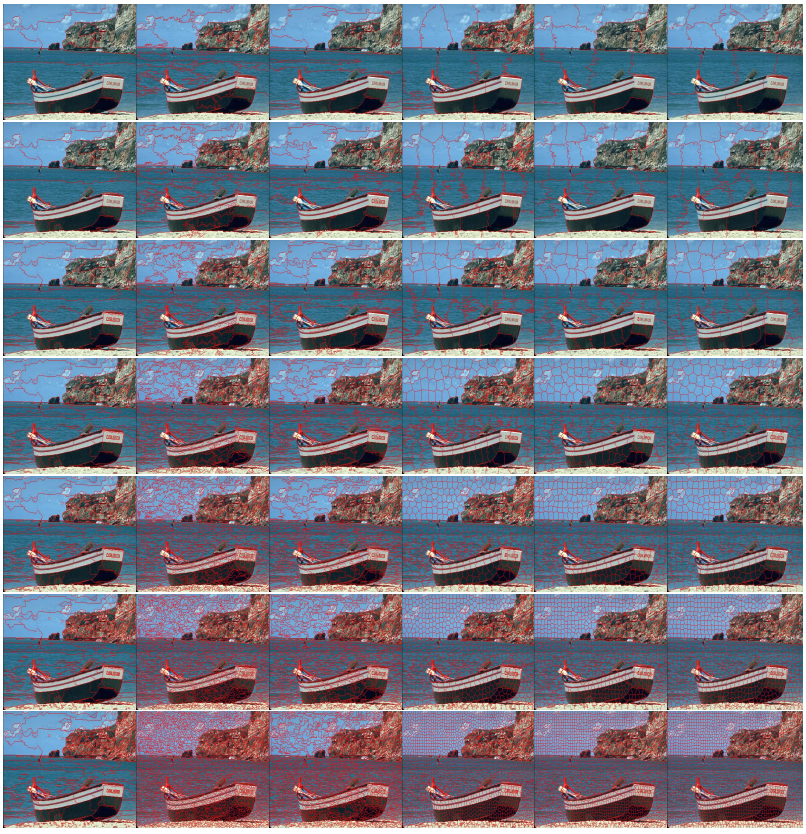
Source: The Author.

under-segmentation error) (NEUBERT; PROTZEL, 2012).

The proposed SGC, HSGC and IHSGC methods have distinctive features when compared to other methods available in the literature (see Section 3). The proposed approaches generate more homogeneous superpixels with better boundary adherence than the compared methods. Also, the two-step approach HSGC generates a hierarchical tree of nested superpixels, with the exact number of homogeneous superpixels at all scales, while preserving the superpixels boundary correspondences across all scales, improving on state-of-the-art. Also, the proposed stochastic graph contraction strategy helps handling the color and illumination variability that often occurs in natural images.

Is important to mentioning that we do not evaluate the compactness of superpixels, because the objects in natural images have irregular shape and boundaries (WEI et al., 2018) and (VASQUEZ; SCHARCANSKI, 2018).

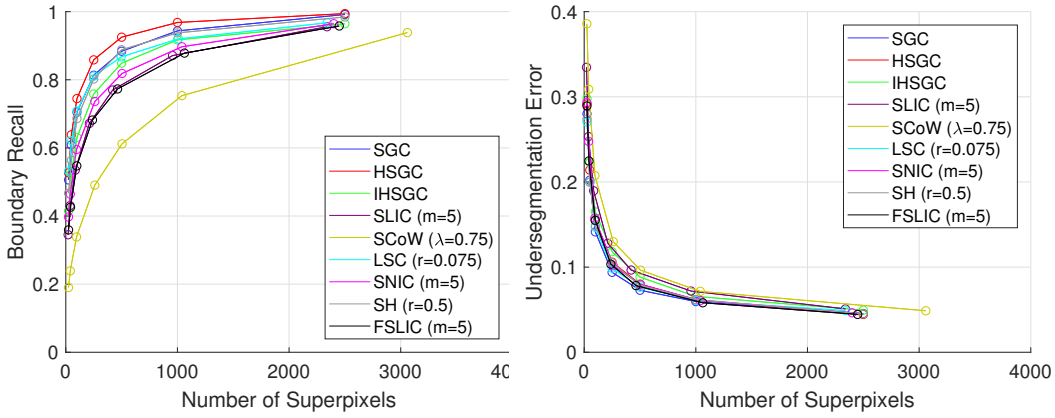
Figure 6.11: Multi-scale superpixels results for one image of the BSDS500 dataset with seven scales for each tested method. These results were obtained for different numbers of superpixels and are shown from the top to the bottom of the image plates: 25, 50, 100, 250, 500, 1000 and 2500 superpixels, respectively. The superpixels boundaries are shown in the red lines overlaid on the original images, and the results obtained for the following compared methods: (a) proposed IHSGC, (b) proposed HSGC, (c) SH (WEI et al., 2018), (d) LSC (LI; CHEN, 2015), (e) SNIC (ACHANTA; SUSSTRUNK, 2017), (e) Fuzzy SLIC (FSLIC) (WU; ZHANG L .AND ZHANG; YAN, 2019), respectively.



(a) IHSGC(b) HSGC (c) SH (d) LSC (e) SNIC (f) FSLIC

Source: The Author.

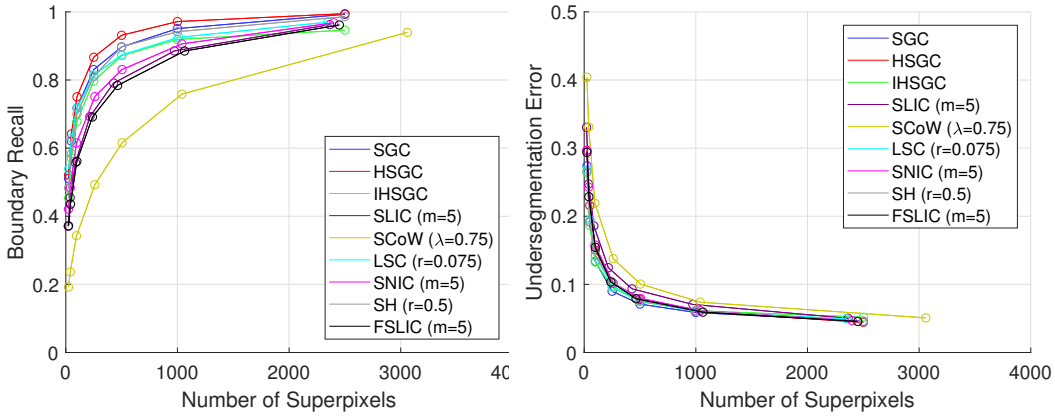
Figure 6.12: Benchmark metric values obtained for IHSGC, HSGC method and superpixels comparative methods using (a) Boundary Recall and (b) Under-segmentation error for the test set images of the BSD300 dataset.



(a) BR (b) UE

Source: The Author.

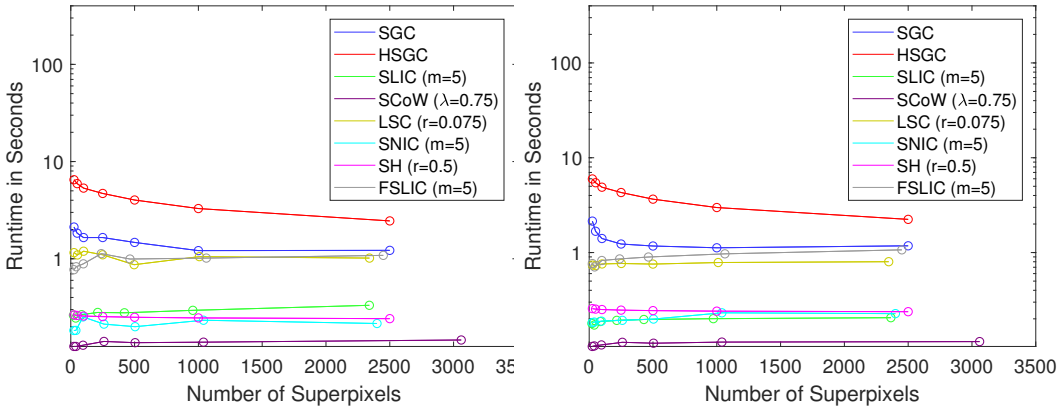
Figure 6.13: Benchmark metric values obtained for IHSGC, HSGC method and superpixels comparative methods using (a) Boundary Recall and (b) Under-segmentation error for the test set images of the BSD500 dataset.



(a) BR (b) UE

Source: The Author.

Figure 6.14: Runtime metric for IHSGC, HSGC method and superpixels comparative methods using (a) BSD300 and (b) BSDS500 datasets.



(a) Runtime BSDS300 (b) Runtime BSDS500

Source: The Author.

Finally, it shall be observed that the proposed methods are stochastic and the superpixels results varies slightly between runs, but the superpixels tend to be consistent with the object boundaries in an image. The variability of the SGC and HSGC results obtained in 10 runs of each method, and for 7 different number of superpixels, is discussed in Section 6.1.1.1.

6.1.1.1 Variability of the Superpixels Results

To evaluate the variability of the proposed SGC and HSGC stochastic methods, these methods were executed 10 times each to obtain the results for different number of superpixels. The results for 7 scales with the number of superpixels equal to 25, 50, 100, 250, 500, 1000 and , 2500, respectively, are shown in the Tables 6.1 and 6.2 for the SGC method using the test images from the BSDS500 dataset; and the results for the HSGC method using the test images of the BSDS500 dataset are in the Tables 6.3 and 6.4.

Figures 6.15 and 6.16 show the overlayed results obtained by running 10 times SGC and HSGC for the BSDS500 dataset, respectively. It can be verified in Figures 6.15 and 6.16 that the variability tends to be small for boundary recall (left) and for under-segmentation error (right), and the quantitative results confirming the visual analysis appear in the Tables 6.1, 6.2, 6.3 and 6.4.

Also, to provide an visual evaluation of the obtained superpixels in different runs, the same image was represented in 4 scales by HSGC and with different number of superpixels per image by SGC (with 50, 250, 500 and 1000 superpixels for the 4 scales

of HSGC and for the SGC image representations, respectively). Figures 6.17 and 6.18 shown 3 different runs of SGC and HSGC in each row, allowing to evaluate visually the boundary variability of the superpixels obtained by SGC and HSGC. These results suggest that quantitatively the proposed methods tend to present a small variability, while the visual results may differ from run to run of SGC and HSGC since the obtained superpixels tend to be different (i.e. to include different pixels sets). However, the SGC and HSGC methods consistently obtain homogeneous superpixels with their boundaries aligned with the image objects boundaries, which may help performing some computer vision tasks (e.g. image segmentation).

Table 6.1: Boundary recall statistics obtained by running 10 times the SGC method for the test set images of the BSDS500 dataset.

No. Superpixels	μ	σ^2	σ
25	$0,5114$	$5,44489E-06$	$0,002333429$
50	$0,6195$	$2,55789E-06$	$0,00159934$
100	$0,7179$	$3,54333E-07$	$0,000595259$
250	$0,8295$	$1,24844E-06$	$0,001117338$
500	$0,8966$	$2,25444E-07$	$0,00047481$
1000	$0,9510$	$5,21111E-08$	$0,000228279$
2500	$0,9927$	$4,55556E-09$	$6,74949E-05$

Table 6.2: Under-segmentation error statistics obtained by running 10 times the SGC method for the test set images of the BSDS500 dataset.

No. Superpixels	μ	σ^2	σ
25	$0,2713$	$5,26933E-06$	$0,002295503$
50	$0,1909$	$6,72889E-07$	$0,000820298$
100	$0,1354$	$6,96111E-07$	$0,000834333$
250	$0,0900$	$4,93333E-08$	$0,000222111$
500	$0,0710$	$1,34444E-08$	$0,00011595$
1000	$0,0587$	$4,55556E-09$	$6,74949E-05$
2500	$0,0457$	$2,33333E-09$	$4,83046E-05$

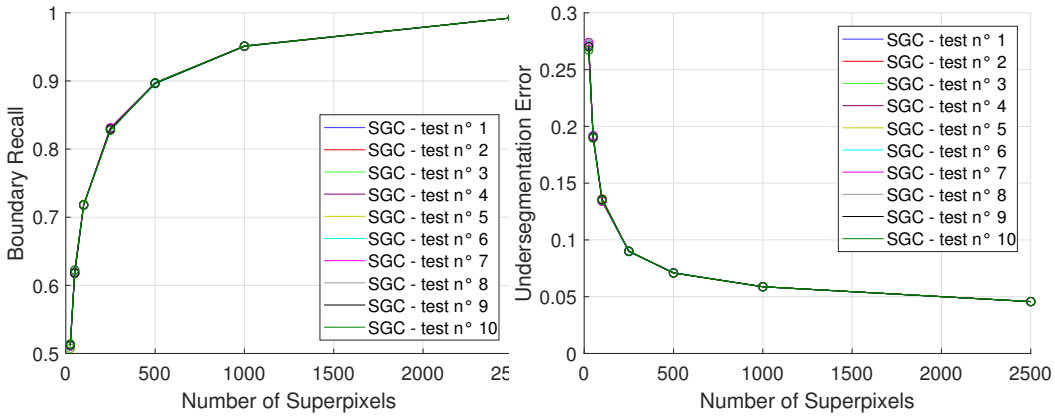
Table 6.3: Boundary recall statistics obtained by running 10 times the HSGC method for the test set images of the BSDS500 dataset.

No. Superpixels	μ	σ^2	σ
25	$0,5206$	$1,09151E-05$	$0,003303803$
50	$0,6418$	$3,33556E-06$	$0,00182635$
100	$0,7506$	$9,33778E-07$	$0,000966322$
250	$0,8660$	$7,81778E-07$	$0,000884182$
500	$0,9303$	$3,29889E-07$	$0,00057436$
1000	$0,9713$	$3,21111E-08$	$0,000179196$
2500	$0,9948$	$4E-09$	$6,32456E-05$

Table 6.4: Under-segmentation error statistics obtained by running 10 times the HSGC method for the test set images of the BSDS500 dataset.

No. Superpixels	μ	σ^2	σ
25	$0,2984$	$4,86011E-06$	$0,002204566$
50	$0,2127$	$3,461E-06$	$0,001860376$
100	$0,1517$	$8,35111E-07$	$0,000913844$
250	$0,1033$	$1,07111E-07$	$0,000327278$
500	$0,0781$	$6,67778E-08$	$0,000258414$
1000	$0,0607$	$1,82222E-08$	$0,00013499$
2500	$0,0442$	$7,66667E-09$	$8,75595E-05$

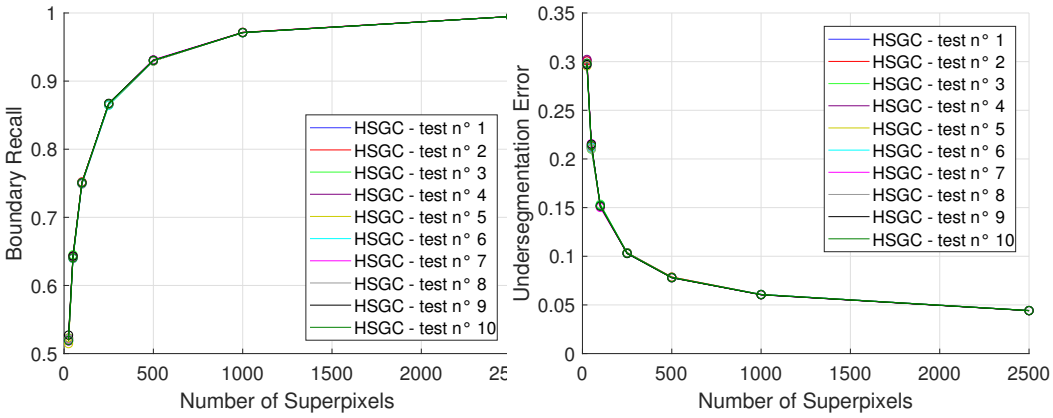
Figure 6.15: Variability of (a) boundary recall and (b) under-segmentation error values obtained by running 10 times the SGC method for the test set images of the BSDS500 dataset.



(a) BR (b) UE

Source: The Author.

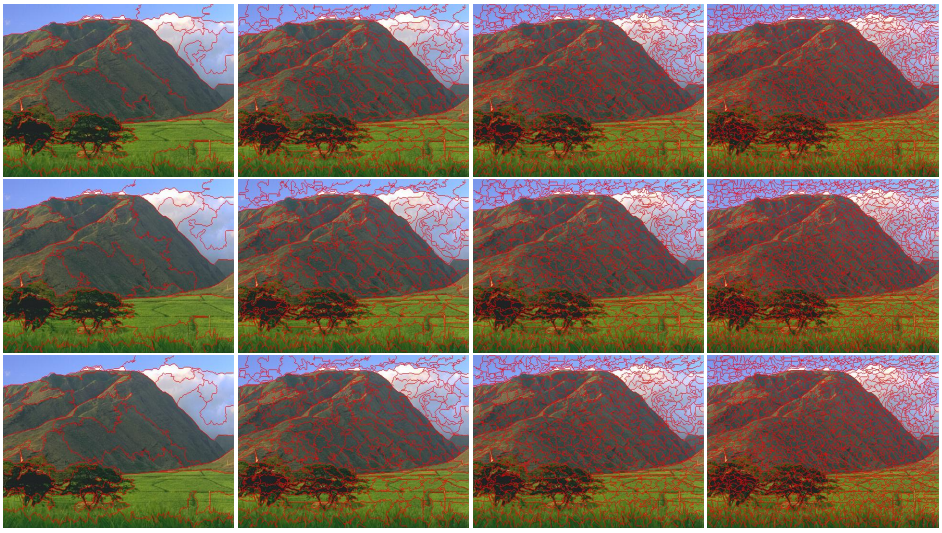
Figure 6.16: Variability of (a) boundary recall and (b) under-segmentation error values obtained by running 10 times the HSGC method for the test set images of the BSDS500 dataset.



(a) BR (b) UE

Source: The Author.

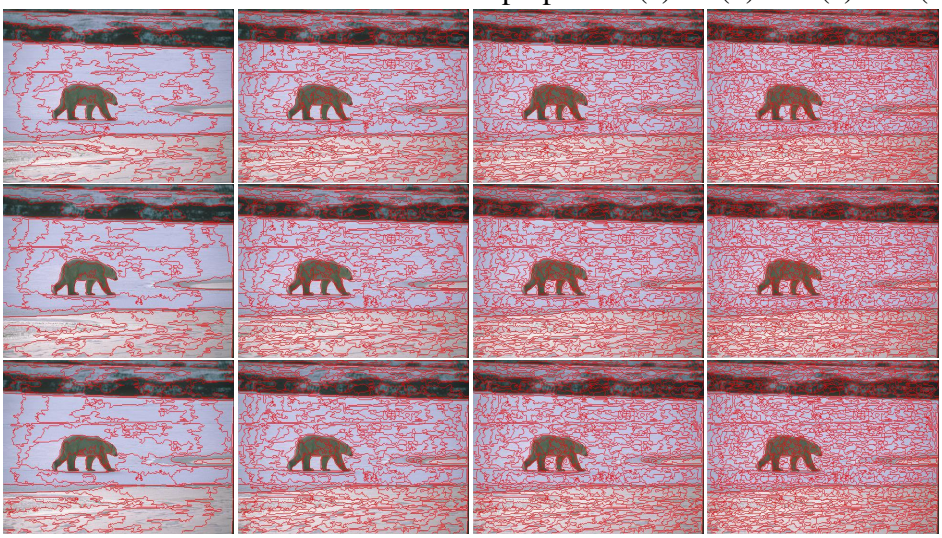
Figure 6.17: Superpixels results for the same image of the BSDS500 dataset to evaluate the variability of our stochastic method. Rows show the results for three different runs of the SGC method to obtain the same number of superpixels: (a) 50 (b) 250 (c) 500 (d) 1000.



(a) N=50 (b) N=250 (c) N=500 (d) N=1000

Source: The Author.

Figure 6.18: Superpixels results for the same image of the BSDS500 dataset to evaluate the variability of the HSGC method. Rows show the results for three different runs of the HSGC method to obtain the same number of superpixels: (a) 50 (b) 250 (c) 500 (d) 1000.



(a) N=50 (b) N=250 (c) N=500 (d) N=1000

Source: The Author.

6.2 Experimental Results: Stochastic Spectral Segmentation

Next, our proposed SSGC method for spectral image segmentation is evaluated qualitatively and quantitatively, and compared with other superpixels-based color image

segmentation approaches that are representative of the state-of-the-art. The comparative methods are: Normalized Spectral Clustering (NCUT) (SHI; MALIK, 2000), Stochastic Region Merging (SRM) (WONG; SCHARCANSKI; FIEGUTH, 2011), Superpixel-based Fast Fuzzy C-Means Clustering (SFFCM) (LEI et al., 2018), Improved Superpixel-based Fast Fuzzy C-Means Clustering (ISFFCM) (WU et al., 2019).

This method comparison is based on the benchmark proposed in (ARBELAEZ et al., 2011), which uses two publicly available datasets BSDS300 and BSDS500. The BSDS300 dataset consists of 300 natural color images with 481×321 or 321×481 pixels each, divided into 200 images for training and 100 images for testing. The BSDS500 dataset consists of 500 natural color images with 481×321 and 321×481 pixels each, divided into 200 images for training, 100 images for evaluation and 200 images for testing. The experimental results are based on the test images set provided in both datasets.

The proposed method was implemented in MATLAB on a PC compatible computer, with an Intel Core i3 2.13 GHz processor and 8GB RAM. The average runtime to obtain the superpixels for 481×321, or 321×481, images is about $7s$ for an input of 30 superpixels.

The comparative image segmentation methods are superpixel-based approaches, or were configured to use SLIC superpixels as input (ACHANTA et al., 2012). The NCUT and SRM methods were configured to work on color images and to receive as initialization a set of SLIC superpixels. NCUT describes each superpixels using 3D CIELAB color histograms, and chi-square distance are used to build the superpixels similarity matrix. SRM describes each superpixel using the mean CIELAB color, and uses the Euclidean distance as a (dis)similarity measure.

All methods were configured to receive a number of superpixels equal to, or approximately equal to, 30, 60 and 120 superpixels. Also, the result of all of comparison methods were configured to generate 5 image segmentations with the number of image regions ranging between 5 to 20, approximately. The NCUT and SRM methods use as input SLIC superpixels with a compactness parameter $m = 5$ to initialize with superpixels with good boundary adherence to the objects in the image. Additionally, the regularization parameter was set to was configured with $Q = 10$ in the SRM method since higher Q values tend to generate over-segmentation. The ISFFCM method uses the Fuzzy-SLIC superpixels (WU; ZHANG L .AND ZHANG; YAN, 2019) with a compactness parameter set to $m = 5$ to obtain good boundary adherence to the image objects. Finally, the SFFCM method uses its own superpixels method and does not allow to configure the number

of superpixels to input, and outputs 20 to 60 superpixels.

Figs. 6.19 and 6.20 illustrate visually some results obtained by the SSGC and $SSGC_E$ methods for six images of the BSDS300 and 3 images of the BSDS500 datasets, respectively, with the number of final segmented regions equal to $K = 20$. In Figs. 6.19 and 6.20, the columns represent the tested method (left to right): original Image, proposed SSGC method, proposed $SSGC_E$ method, Stochastic Region Merging (SRM) initialized by SLIC superpixels, NCUT (NG; JORDAN; WEISS, 2002), Superpixel-based Fast Fuzzy C-Means Clustering (SFFCM) (LEI et al., 2018), Improved Superpixel-based Fast Fuzzy C-Means Clustering (ISFFCM) (WU et al., 2019), respectively. It shall be observed that SSGC and $SSGC_E$ tend to segment the images into regions that are more consistent visually.

6.2.1 Quantitative Evaluation Results

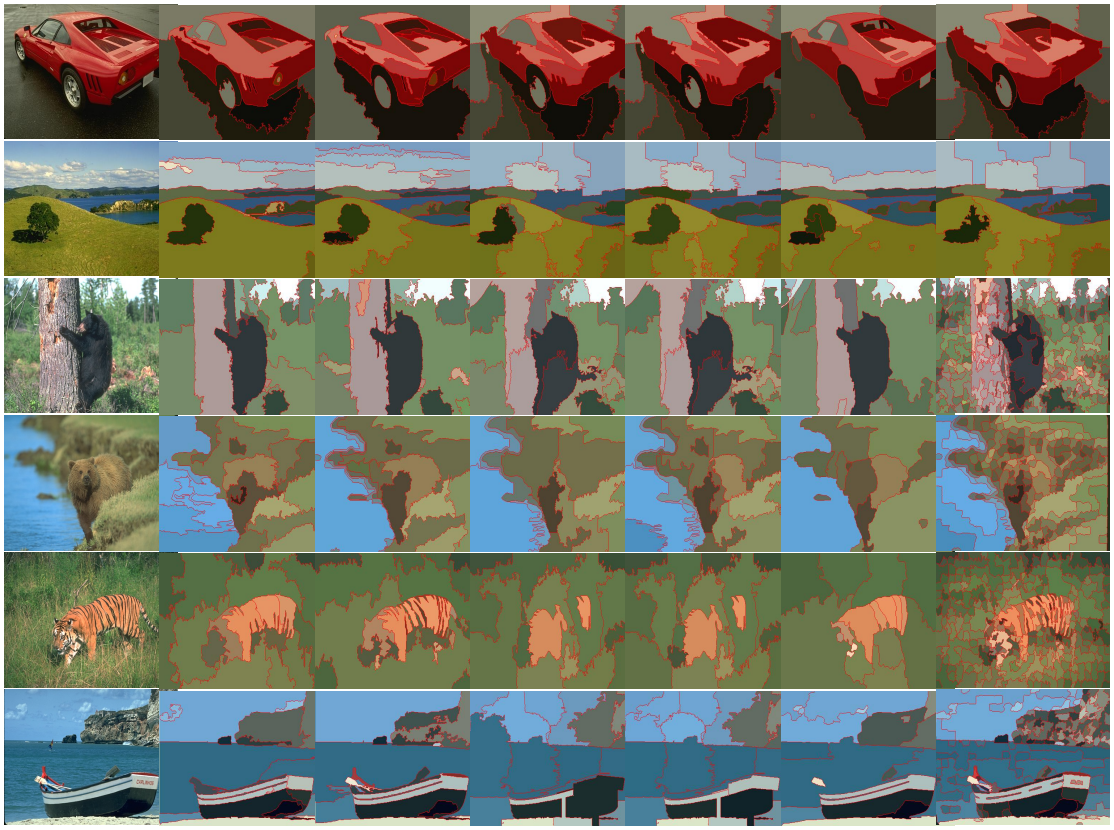
The Tables 6.5, 6.6 and 6.7 show the segmentation covering (COV), Probabilistic rand index (PRI) and volume of information (VoI) results for the BSDS300 and BSDS500 datasets, initializing the methods with 30, 60 and 120 input superpixels, respectively. The quantitative results include the proposed methods SSGC and $SSGC_E$ and four other comparative methods, namely, Stochastic Region Merging initialized by SLIC superpixels (SRM) (WONG; SCHARCANSKI; FIEGUTH, 2011), Normalized Spectral Clustering (NCUT) (SHI; MALIK, 2000), Superpixel-based Fast Fuzzy C-Means Clustering (SFFCM) (LEI et al., 2018), Improved Superpixel-based Fast Fuzzy C-Means Clustering (ISFFCM) (WU et al., 2019). All these methods have superpixels as a pre-processing step, and are based on clustering or on the data eigen-decomposition. It shall be observed that the method SFFCM uses its own superpixel initialization step, named multi-scale morphological gradient reconstruction (MMGR) (LEI et al., 2018), and this method also generates a small number of superpixels automatically. In these Tables, the up arrow ($\uparrow$) symbol indicates that a higher value is better and the down arrow ($\downarrow$) indicates the opposite. The bold values in the Tables indicate the best result for the respective metric. Additionally, it is shown the computation time (in seconds) to obtain the 5 image segmentations with 5, 7, 10, 15 and 20 image regions, approximately.

The Tables 6.8, 6.9 and 6.10 show the quantitative results for the proposed method $SSGC_E$ and the comparative methods SFFCM, ISFFCM, NCUT and SRM using 250, 500 and 1000 superpixels as initializations.

Figure 6.19: Obtained segmentation results for six images of the BSDS300 dataset. All methods were initialized with approximately 30 superpixels, and each BSDS300 image was segmented into 20 regions. The columns represent the tested method (left to right): (a) original Image, (b) proposed SSGC method, (c) proposed $SSGC_E$ method, (d) Stochastic Region Merging (SRM), (e) NCUT (NG; JORDAN; WEISS, 2002), (f) Superpixel-based Fast Fuzzy C-Means Clustering (SFFCM) (LEI et al., 2018), (g) Improved Superpixel-based Fast Fuzzy C-Means Clustering (ISFFCM) (WU et al., 2019), respectively.

(a) Original (b) SSGC (c) $SSGC_E$ (d) SRM (e) NCUT (f) SFFCM (g) ISFFCM

Figure 6.20: Obtained segmentation results for six images of the BSDS500 dataset. All methods were initialized with approximately 30 superpixels, and each BSDS500 image was segmented into 20 regions. The columns represent the tested method (left to right): (a) original Image, (b) proposed SSGC method, (c) proposed $SSGC_E$ method, (d) Stochastic Region Merging (SRM), (e) NCUT (NG; JORDAN; WEISS, 2002), (f) Superpixel-based Fast Fuzzy C-Means Clustering (SFFCM) (LEI et al., 2018), (g) Improved Superpixel-based Fast Fuzzy C-Means Clustering (ISFFCM) (WU et al., 2019), respectively.



(a) Original (b) SSGC (c) $SSGC_E$ (d) SRM (e) NCUT (f) SFFCM (g) ISFFCM

Table 6.5: Comparison of superpixels-based color image segmentation algorithms for the BSDS300 and BSDS500 datasets using 30 superpixels as an initialization step.

Method	BSDS300			BSDS500			Time (s)
	COV (↑)	PRI (↑)	VoI (↓)	COV (↑)	PRI (↑)	VoI (↓)	
SFFCM	0.55	0.78	2.07	0.57	**0.80**	1.98	5.10
ISFFCM	0.48	0.76	2.41	0.49	0.78	2.37	3.00
NCUT	0.51	0.76	2.17	0.54	0.78	2.12	2.10
SRM	0.50	0.76	2.29	0.50	0.77	2.29	**1.80**
SSGC	**0.61**	**0.79**	**1.76**	**0.62**	**0.80**	**1.76**	7.20
$SSGC_E$	0.59	0.78	1.88	0.61	**0.80**	1.84	5.40

Table 6.6: Comparison of superpixels-based color image segmentation algorithms for the BSDS300 and BSDS500 datasets using 60 superpixels as an initialization step.

Method	BSDS300			BSDS500			Time (s)
	COV (↑)	PRI (↑)	VoI (↓)	COV (↑)	PRI (↑)	VoI (↓)	
SFFCM	0.55	0.78	2.07	0.57	**0.80**	1.98	5.10
ISFFCM	0.46	0.75	2.62	0.49	0.78	2.49	4.50
NCUT	0.54	0.77	2.08	0.55	0.78	2.06	3.30
SRM	0.52	0.77	2.24	0.52	0.77	2.29	**2.10**
SSGC	0.59	0.77	**1.84**	0.59	0.77	**1.82**	15.00
$SSGC_E$	**0.60**	**0.79**	1.90	**0.60**	**0.80**	1.88	7.80

Table 6.7: Comparison of superpixels-based color image segmentation algorithms for the BSDS300 and BSDS500 datasets using 120 superpixels as an initialization step.

Method	BSDS300			BSDS500			Time (s)
	COV (↑)	PRI (↑)	VoI (↓)	COV (↑)	PRI (↑)	VoI (↓)	
SFFCM	0.55	0.78	2.07	0.57	0.80	1.98	5.10
ISFFCM	0.45	0.75	2.78	0.49	0.78	2.62	6.90
NCUT	0.58	0.78	1.93	0.59	0.80	1.94	3.00
SRM	0.54	0.78	2.12	0.54	0.78	2.21	**2.40**
SSGC	0.58	0.78	1.90	0.59	0.78	1.87	31.80
$SSGC_E$	**0.61**	**0.79**	**1.85**	**0.63**	**0.81**	**1.78**	10.80

Table 6.8: Comparison of superpixels-based color image segmentation algorithms for the BSDS300 and BSDS500 datasets using 250 superpixels as an initialization step.

Method	BSDS300			BSDS500			Time (s)
	COV (↑)	PRI (↑)	VoI (↓)	COV (↑)	PRI (↑)	VoI (↓)	
SFFCM	0.55	0.78	2.07	0.57	0.80	1.98	5.10
ISFFCM	0.46	0.75	2.87	0.49	0.78	2.71	10.20
NCUT	0.60	0.79	1.87	0.61	0.81	1.85	**3.30**
SRM	0.56	0.78	2.06	0.56	0.79	2.09	3.60
$SSGC_E$	**0.63**	**0.80**	**1.79**	**0.65**	**0.82**	**1.73**	14.40

6.2.2 Discussion and Ablation

As mentioned before, 3 well known quantitative segmentation measures were used as criteria to compare the SSCG and $SSGC_E$ methods with other methods that are representative of the state-of-the-art, namely segmentation covering (COV), probabilistic

Table 6.9: Comparison of superpixels-based color image segmentation algorithms for the BSDS300 and BSDS500 datasets using 500 superpixels as an initialization step.

Method	BSDS300			BSDS500			Time (s)
	COV (↑)	PRI (↑)	VoI (↓)	COV (↑)	PRI (↑)	VoI (↓)	
SFFCM	0.55	0.78	2.07	0.57	0.80	1.98	**5.10**
ISFFCM	0.44	0.75	3.05	0.48	0.78	2.86	19.20
NCUT	0.61	0.79	1.80	0.61	0.80	1.81	5.40
SRM	0.59	0.80	1.94	0.58	0.80	2.01	5.40
$SSGC_E$	**0.63**	**0.80**	**1.72**	**0.64**	**0.82**	**1.74**	28.80

Table 6.10: Comparison of superpixels-based color image segmentation algorithms for the BSDS300 and BSDS500 datasets using 1000 superpixels as an initialization step.

Method	BSDS300			BSDS500			Time (s)
	COV (↑)	PRI (↑)	VoI (↓)	COV (↑)	PRI (↑)	VoI (↓)	
SFFCM	0.55	0.78	2.07	0.57	0.80	1.98	**5.10**
ISFFCM	0.44	0.75	3.25	0.46	0.77	3.06	33.00
NCUT	0.60	0.78	1.77	0.60	0.79	1.81	12.60
SRM	0.60	0.79	1.87	0.60	0.80	1.91	9.00
$SSGC_E$	**0.63**	**0.80**	**1.70**	**0.63**	**0.81**	**1.74**	103.50

rand index (PRI) and Variation of Information (VoI) (ARBELAEZ et al., 2011). The comparative methods are the following: Stochastic Region Merging initialized by SPIC superpixels (SRM) (WONG; SCHARCANSKI; FIEGUTH, 2011), Normalized Spectral Clustering (NCUT) (SHI; MALIK, 2000), Superpixel-based Fast Fuzzy C-Means Clustering (SFFCM) (LEI et al., 2018), and Improved Superpixel-based Fast Fuzzy C-Means Clustering (ISFFCM) (WU et al., 2019). All these methods receive as input a superpixel representation of the original color image. As images references for these comparisons (ground truth), the human segmented images of the BSDS300 and BSDS500 datasets were used.

As it can be observed in Figs. 6.19 and 6.20, the proposed methods SSGC and $SSGC_E$ produce connected and homogeneous regions with better boundary adherence than the comparative methods.

Also, it is possible to verify in Tables 6.5, 6.6 and 6.7 that the proposed approaches SSGC and $SSGC_E$ tend to obtain better segmentation covering (COV), probabilistic rand index (PRI) and variation of Information (VoI) measures than the comparative state-of-the-art methods. The evaluated segmentation measures tend to improve when less superpixels are used in the initialization step. It shall be observed specially the reduction in terms of computational time as the number of initialization superpixels is decreased. The segmentation results in these Tables were obtained using 30, 60 and 120 superpixels as a pre-processing step. Figs. 6.19 and 6.20 show that the proposed method (SSGC) produces

connected and homogeneous regions with better boundary adherence than the comparative methods. NCUT and SRM are faster when using superpixels initialization, but do not provide acceptable visual results for small numbers of output segmented regions. Also, SFFCM and ISFFCM generate disconnected regions with the same segmentation label, and tend to obtain smoothed segmented regions boundaries. It can be verified in Figs. 6.19 and 6.20 that the SSGC method can generate more color homogeneous regions, and it also can obtain the desired number of image segments (image regions).

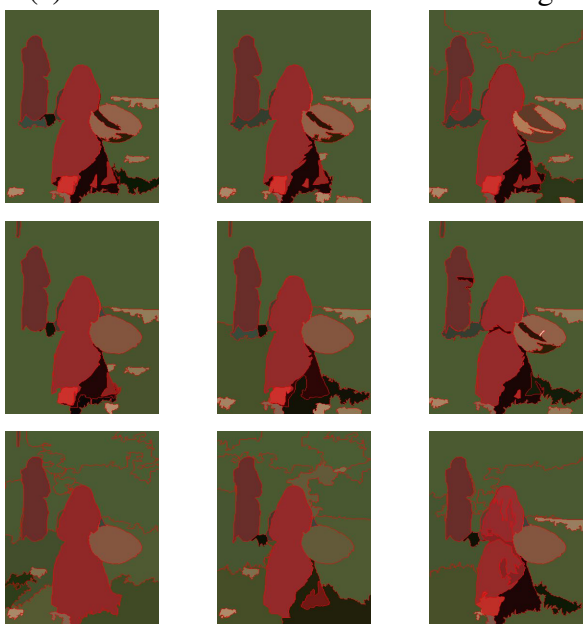
Therefore, the proposed stochastic spectral graph contraction method potentially can improve on state-of-the-art segmentation methods, since it allows to represent the image contents more accurately including the aforementioned challenges. The proposed method SSGC has distinctive features in comparison to other methods available in the literature. The SSGC approach generates more homogeneous regions with better connectivity to cover a ground truth segmentation. Due to stochastic superpixels method, the results have better boundary adherence to objects in the images. Also, the proposed stochastic spectral graph contraction strategy tends to generate meaningful visual data clusters, which helps handling the color and illumination variability that often occurs in natural images.

6.2.2.1 Variability of the Segmentation Results

To evaluate the variability of the segmentations produced by the proposed stochastic methods, SSGC and $SSGC_E$ were run 10 times and the averages of the different segmentation measures (i.e. COV, PRI and VoI) were calculated. Tables 6.11 and 6.12 show the averages and standard deviations for each measure computed for a subset of 50 images of the BSDS500 dataset, and the results are presented for 3 different number of superpixels used as initialization (30, 60 and 120).

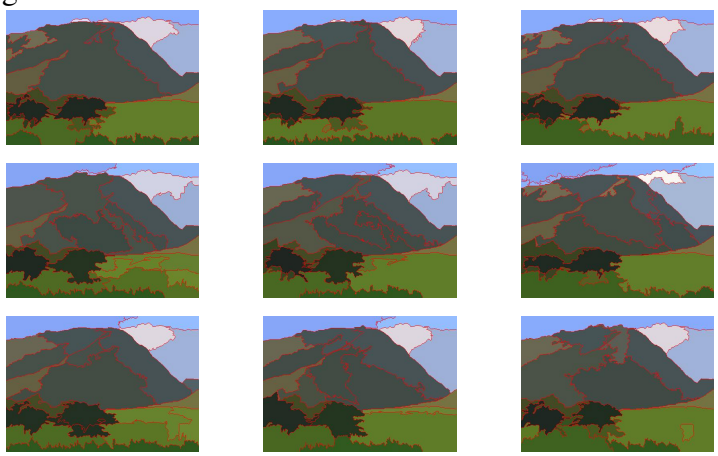
Figs. 6.21 and 6.22 show the results of 3 different run tests of the proposed SSGC and $SSGC_E$ methods using 3 different numbers of superpixels as an initialization step (each row shows the results for 30, 60 and 120 superpixels), based on an image from BSDS500 dataset. These results suggest that the proposed SSGC and $SSGC_E$ methods have a small variability in the obtained segmentations, and the visual segmentation results differ slightly but are consistent, as indicated by the quantitative results shown in the Tables 6.11 and 6.12.

Figure 6.21: Segmentation results for the same image of the BSDS500 dataset to evaluate the variability of the segmentations obtained with the proposed SSGC stochastic segmentation method. In (a) to (c) three different tests for the same image can be seen.



(a) Test 1 (b) Test 2 (c) Test 3

Figure 6.22: Segmentation results for the same image of the BSDS500 dataset to evaluate visually the variability of the $SSGC_E$ stochastic method. In (a) to (c) three different tests for the same image can be seen.



(a) Test 1 (b) Test 2 (c) Test 3

Table 6.11: Average and standard deviation of the COV, PRI and VoI segmentation measures obtained when the SSGC method is run 10 times for a subset of 50 images of the BSDS500 test dataset. The SSGC method is initialized with 30, 60 and 120 superpixels.

Number of Superpixels	COV (↑)	PRI (↑)	VoI (↓)
30	0.63 ± 0.006	0.81 ± 0.006	1.69 ± 0.025
60	0.60 ± 0.006	0.78 ± 0.004	1.76 ± 0.022
120	0.61 ± 0.009	0.80 ± 0.007	1.78 ± 0.029

Table 6.12: Average and standard deviation of the measures used to evaluate the segmentation results when the $SSGC_E$ method is run 10 times using images of the BSDS500 dataset. The $SSGC_E$ method is initialized with three different number of superpixels 30, 60 and 120.

Number of Superpixels	COV (↑)	PRI (↑)	VoI (↓)
30	0.61 ± 0.004	0.80 ± 0.003	1.82 ± 0.018
60	0.61 ± 0.007	0.81 ± 0.005	1.82 ± 0.027
120	0.64 ± 0.006	0.82 ± 0.004	1.74 ± 0.014

7 CONCLUSIONS AND FUTURE WORK

In this work, novel stochastic superpixels generation methods designed for single-scale and multi-scale representation of image patches were proposed. The proposed stochastic graph contraction method (SGC) (see Section 4.2) uses an unsupervised and stochastic strategy to generate superpixels. The SGC method starts at the pixel level and it can handle challenging issues such as noise, color and illumination variability, as well as weak boundary separation between the objects in an image.

Also, the hierarchical approaches were proposed (IHSGC and HSGC), where the superpixels boundaries correspond at finer and coarser scales, and the multi-level relationship between finer and coarser superpixels elements boundaries is explicit. The proposed IHSGC (see Section 4.1) and HSGC (see Section 4.2) scheme use one over-segmentation or SGC method as initialization step and apply a hierarchical stochastic graph contraction scheme, which evaluates a superpixels adjacency graph at a finer scale, and obtains the coarser scales iteratively. The proposed graph contraction scheme is flexible and allows to obtain the desired number of superpixels at each scale. Finally, the image contents are represented by hierarchical trees that contain the superpixels adjacency graph at different scales and can be used to describe the image content using several levels of detail.

A new stochastic spectral segmentation approach SSGC was proposed for color images. The proposed methods rely on the stochastic processing of the eigen-decomposition of a superpixels adjacency graph representing the input image. The superpixels adjacency graph undergoes stochastic contractions iteratively until the desired number of image regions is obtained as a final image segmentation. The computational complexity of the proposed segmentation method can be reduced while obtaining a comparable performance, and the $SSGC_E$ has been introduced as an efficient approximation of the SSGC approach.

Experiments were conducted on the BSDS300 and BSDS500 databases in Section 6, that are popular and publicly available. These experiments suggest that the proposed methods IHSGC, SGC and HSGC for stochastic superpixels generation potentially can achieve better visual and quantitative results (i.e., in terms of boundary recall and under-segmentation error) than the comparative methods that are recent and representative of the state-of-the-art. It shall be observed that the proposed IHSGC and HSGC methods can be used to generate multi-scale image representations and help the recognition of visual patterns, specially those having multi-scale features (e.g. in image processing and computer vision applications).

Based on several experiments conducted on the BSDS300 and BSDS500 image datasets, the proposed stochastic spectral graph contraction methods potentially can improve on state-of-the-art segmentation methods, since they allow to represent the image contents more accurately. The proposed SSGC method and its efficient approximation $SSGC_E$ have distinctive features in comparison to other methods available in the literature. The SSGC and $SSGC_E$ approaches tend to generate more homogeneous image regions with better connectivity. Also, probably because of the stochastic superpixels used to initialize the SSGC and $SSGC_E$ methods, the segmented regions have better boundary adherence to objects in the input images. Besides, the proposed stochastic spectral graph contraction strategy tends to generate meaningful visual data clusters, which helps handling the color and illumination variability that often occurs in natural images.

Based on experimental results, applications of this stochastic methods could be: a) medical imaging analysis and interpretation; b) simplified sparse image representations; c) processing of temporal visual information; d) multi-scale clustering of visual data; and e) creation of multi-scale descriptors.

8 PUBLICATIONS

The main contributions of this research work have been submitted to an international conference and to international journals, as listed below in chronological order of submission:

1. VASQUEZ, D.; SCHARCANSKI, J.; WONG, A. *Stochastic color image segmentation using spatial constraints*. In: IEEE. 2015 International Conference on Instrumentation and Measurement (2015 I2MTC). 2015. **Qualis CAPES A2**.

   Abstract — This paper describes an automated method for segmenting color images based on a modified stochastic region merging strategy with multi-scale spatial constraints. First, a bilateral decomposition is performed, and an over-segmentation process is then performed based multichannel information and multi-scale gradients. Next, each sub-region is represented using a normalized color histogram in the CIELAB color space, and a region adjacency graph is constructed based on the over-segmentation results. Finally, a stochastic region merging strategy with spatial constraints is performed on the region adjacency graph to construct one segmentation map for each scale of representation. Our preliminary visual and quantitative experimental results on the Berkeley image database (BSDS500) are encouraging, and suggest that our proposed approach can provide accurate segmentation results.

2. VASQUEZ, D.; SCHARCANSKI, J. *An iterative approach for obtaining multi-scale superpixels based on stochastic graph contraction operations*. Expert Systems with Applications, v. 102, p. 57–69, 2018. **Qualis CAPES A1, Impact factor 4.292**.

   Abstract — Superpixels have many applications in visual information processing, and can be used to reduce redundant information of an image, as well as the computational complexity of other expensive tasks (e.g., image segmentation). In this work, an iterative hierarchical stochastic graph contraction (IHSGC) method for multi-scale superpixels generation is proposed. A stochastic strategy is used to generate multi-scale superpixels, and each superpixel is represented by a hierarchical tree and describes an image patch at fine and coarse scales simultaneously. The proposed method consists of two main steps. The first step initializes the method based on a multi-channel unsupervised stochastic over-segmentation at the pixel level. The proposed over-segmentation scheme actually performs hierarchical stochastic clustering of visual features (i.e. pixels, image patches, and potentially

can be applied to other visual features as well), while preserving the local spatial relationships across different scales. The second step consists of an iterative hierarchical stochastic graph contraction method. Coarser scales are generated by graph contractions until the desired number of superpixels is obtained. The experimental results based on the popular Berkeley segmentation databases BSDS300 and BSDS500 suggest that the proposed approach potentially can perform better than comparative state-of-the-art methods in terms of boundary recall and under-segmentation error.

3. VASQUEZ, D.; SCHARCANSKI, J.; WONG, A. *Multi-scale Superpixel Generation from Natural Images via Hierarchical Stochastic Graph Contraction*. Submitted to the international journal *Computer Vision and Image Understanding*, **Qualis CAPES A1, Impact factor 3.121**.

   Abstract — A number of different superpixel generation approaches have been proposed in literature, and some the biggest challenges faced by the existing approaches are: i) poor adherence to the object boundaries, and ii) difficulty in generating well-structured superpixels under non-ideal scene conditions (e.g., noise, color and illumination variability), iii) weak boundary separation between objects, and iv) difficulty to obtain hierarchical and nested image representations where a multi-scale superpixel at a coarser scale can be represented as the union of the set of finer scales superpixels covered by the coarser scale superpixel. Motivated to tackle these key challenges, we introduce a novel fast stochastic algorithm for generating multi-scale superpixel representations for natural images. More specifically, the proposed superpixel generation method represents a given image as a weighted adjacency graph, upon which a series of stochastic graph contractions is performed in a hierarchical fashion to obtain a concise, multi-scale graph representation of the scene, with each node in the graph representing a stochastically condensed superpixel at a given scale. The proposed hierarchical stochastic graph contraction (HSGC) algorithm can thus be leveraged to generate multi-scale superpixel representations comprised of superpixels that exhibit highly homogeneous properties, and strong adherence to object boundaries within the scene, even in the presence of uncertainty due to non-ideal scene conditions. Experimental results on the popular Berkeley segmentation databases BSDS300 and BSDS500 demonstrate that the proposed HSGC approach can achieve state-of-the-art performance when compared to other tested superpixel generation methods in terms of boundary recall and

under-segmentation error, suggesting that the proposed HSGC approach could be effective in creating condensed representations of a natural scene image.

4. VASQUEZ, D.; SCHARCANSKI, J.; WONG, A. *Color image segmentation via stochastic spectral graph contraction*. Submitted to the international journal *Pattern Recognition*, **Qualis CAPES A1, Impact factor 7.196**.

   Abstract — Image segmentation is essential for many applications in computer vision (e.g. object recognition and image classification). However, image segmentation can be challenging under non-ideal scenarios such as the presence of noise, exhibit color and illumination variability, or possess poor contrast. To improve robustness under such scenarios, this work proposes an automatic method for color image segmentation based on stochastic graph contraction operations in the spectral domain. In order to handle to spectral decomposition scalability problem, stochastic superpixels are leveraged for improved representational efficiency. The superpixels are represented in the spectral domain to promote their clustering, and stochastic contraction operations are performed on a superpixel adjacency graph (SAG) to obtain the image segmentation. The proposed was evaluated experimentally using the popular Berkeley segmentation databases BSDS300 and BSDS500, and it was found that the proposed approach potentially can perform better than comparative state-of-the-art methods in terms of standard metrics such as covering, probabilistic rand index, and variation of information.

REFERENCES

ACHANTA, R. et al. Slic superpixels compared to state-of-the-art superpixel methods. **IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)**, v. 34, n. 11, p. 2274–2282, 05 2012.

ACHANTA, R.; SUSSTRUNK, S. Superpixels and polygons using simple non-iterative clustering. In: **IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2017. p. 4895–4904.

AGGARWAL, C.; REDDY, C. E. **Data Clustering: Algorithms and Applications**. New York, NY, USA: CRC Press, 2014. ISBN 978-1-46-655821-2. Available from Internet: <http://www.charuaggarwal.net/clusterbook.pdf>.

ARBELAEZ, P. et al. Contour detection and hierarchical image segmentation. **IEEE Transaction on Pattern Analysis Machine Intelligence (PAMI)**, v. 33, n. 5, p. 898–916, 5 2011.

AVELAR, P. et al. Superpixel image classification with graph attention networks. In: **arXiv preprint arXiv:2002.05544**. [S.l.: s.n.], 2020.

BERGH, M. Van den et al. Seeds: Superpixels extracted via energy-driven sampling. **International Journal of Computer Vision**, v. 3, n. 111, p. 298–314, 2015.

CHAN, S.; ZHOU, X.; CHEN, S. Online classification for object tracking based on superpixel. **Neurocomputing**, v. 286, p. 88–108, 2018.

CHEUNG, G. et al. Graph spectral image processing. In: **Proceedings of IEEE**. [S.l.: s.n.], 2018. v. 106.

COMANICIU, D.; MEER, P. Mean shift: a robust approach toward feature space analysis. **IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)**, v. 24, n. 5, p. 603–619, 2002.

DAOUD, M. et al. Automatic superpixel-based segmentation method for breast ultrasound images. **Expert Systems with Applications**, v. 121, p. 78–96, 2019.

DIESTEL, R. **Graph Theory**. New York, NY, USA: Springer, 2000.

FELZENSZWALB, P.; HUTTENLOCHER, D. Efficient graph-based image segmentation. **International Journal of Computer Vision (IJCV)**, v. 59, n. 2, p. 167–181, 2004.

FRACASTORO, G. et al. Superpixel-driven graph transform for image compression. In: **2015 IEEE International Conference on Image Processing (ICIP)**. [S.l.: s.n.], 2015. p. 2631–2635.

GHADIRI, F.; BERGEVIN, R.; BILODEAU, G. From superpixel to human shape modelling for carried object detection. **Pattern Recognition**, v. 89, p. 134–150, 2019.

GRUNDMANN, M. et al. Efficient hierarchical graph-based video segmentation. In: **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2010. p. 2141–2148.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The Elements of Statistical Learning: Data Mining, Inference, and Prediction**. New York, NY, USA: Springer-Verlag, 2009.

HU, Z.; ZOU, Q.; LI, Q. Watershed superpixels. In: IEEE. **2015 IEEE International Conference on Image Processing (ICIP)**. [S.l.], 2015.

JIN, Z.; LI, J.; LI, D. Co-saliency detection for rgbd images based on effective propagation mechanism. **IEEE Access**, v. 7, n. 19088003, p. 141311 – 141318, 2019.

LEI, T. et al. Superpixel-based fast fuzzy c-means clustering for color image segmentation. **IEEE Transactions on Fuzzy Systems**, v. 27, n. 9, 12 2018.

LEVINSHTEIN, A. et al. Turbopixels: Fast superpixels using geometric flows. **IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)**, v. 31, n. 12, p. 2290 – 2297, 12 2009.

LI, H. et al. Inner and inter label propagation: Salient object detection in the wild. **IEEE Transactions on Image Processing**, v. 24, n. 10, p. 3176–3186, 2015.

LI, H. et al. Fast large-scale spectral clustering via explicit feature mapping. **IEEE Transactions on Cybernetics**, p. 1–14, 02 2018.

LI, Z.; CHEN, J. Superpixel segmentation using linear spectral clustering. In: IEEE. **In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.], 2015. p. 1356–1363.

LIU, M. Y. et al. Entropy rate superpixel segmentation. In: **IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2011.

LUXBURG, U. A tutorial on spectral clustering. **Statistics and Computing**, v. 17, p. 395–416, 2007.

MAIRE, M. R. **Contour Detection and Image Segmentation**. Thesis (PhD) — EECS Department, University of California, Berkeley, Sep 2009. Available from Internet: <http://www2.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-129.html>.

NAVA, R.; KYBIC, J. Supertexton-based segmentation in early drosophila oogenesis. In: IEEE. **2015 IEEE International Conference on Image Processing (ICIP)**. [S.l.], 2015. p. 2656–2659.

NEUBERT, P.; PROTZEL, P. Superpixel benchmark and comparison. In: **2012 Forum Bildverarbeitung, Regensburg, Germanny**. [S.l.: s.n.], 2012.

NG, A.; JORDAN, M.; WEISS, Y. On spectral clustering: Analysis and an algorithm. **Advances in Neural Information Processing Systems 14**, MIT Press, p. 849–856, 2002. Available from Internet: <http://papers.nips.cc/paper/2092-on-spectral-clustering-analysis-and-an-algorithm.pdf>.

NOCK, R.; F., N. Statistical region merging. **IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)**, v. 26, n. 11, p. 1452–1458, 2004.

PAL, N.; PAL, S. A review on image segmentation techniques. **Pattern Recognition**, v. 26, n. 9, p. 1277–1294, 1993.

PELE, O.; WERMAN, M. The quadratic-chi histogram distance family. In: **In Proceedings of the 11th European conference on Computer vision Part II (ECCV'10)**. [S.l.: s.n.], 2010. p. 749–762.

SCHARCANSKI, J. Paired graph hierarchies for color image segmentation and representation. In: SBC. **Advances on Graph Theory and Applications, Brazilian Computer Society (SBC)**. [S.l.], 2011. p. 1–12. ISBN 85-88425-07-6.

SHAPIRO, L. G.; STOCKMAN, G. C. **Computer Vision**. New Jersey: Prentice Hall, 2001.

SHEN, J. et al. Real-time superpixel segmentation by dbscan clustering algorithm. **IEEE Transactions on Image Processing**, v. 25, n. 12, p. 5933–5942, 2015.

SHI, J.; MALIK, J. Normalized cuts and image segmentation. **IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)**, v. 22, n. 8, p. 888–905, 2000.

TEPPEI, S. Superpixel segmentation via convolutional neural networks with regularized information maximization. In: . [S.l.: s.n.], 2020. p. 2573–2577.

TUNG, F.; WONG, A.; CLAUSI, D. A. Enabling scalable spectral clustering for image segmentation. **Pattern Recognition**, v. 43, p. 4069–4076, 12 2010.

UNNIKRISHNAN, R.; PANTOFARU, C.; HEBERT, M. A measure for objective evaluation of image segmentation algorithms. In: **2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshopss**. [S.l.: s.n.], 2005. v. 34, p. 34–34. ISBN 0-7695-2372-2.

UZIEL, R.; RONEN, M.; FREIFELD, O. Bayesian adaptive superpixel segmentation. In: **Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)**. [S.l.: s.n.], 2019.

VASQUEZ, D.; SCHARCANSKI, J. An iterative approach for obtaining multi-scale superpixels based on stochastic graph contraction operations. **Expert Systems with Applications**, v. 102, n. 15, p. 57–69, 2018.

VASQUEZ, D.; SCHARCANSKI, J.; WONG, A. A novel 3d approach for the extraction of the wetting front in ct images of soil profiles. In: IEEE. **2013 International Conference on Instrumentation and Measurement (2013 I2MTC)**. [S.l.], 2013. p. 1540–1543.

VASQUEZ, D.; SCHARCANSKI, J.; WONG, A. Automatic framework for extraction and characterization of wetting front propagation using tomographic image sequences of water infiltrated soils. **Plos One**, v. 10, n. 01, 2015.

VASQUEZ, D.; SCHARCANSKI, J.; WONG, A. Stochastic color image segmentation using spatial constraints. In: IEEE. **2015 International Conference on Instrumentation and Measurement (2015 I2MTC)**. [S.l.], 2015.

VÁSQUEZ, D.; SCHARCANSKI, J.; WONG, A. Color image segmentation via stochastic spectral graph contractions. **Pattern Recognition**, Submitted, 2020.

VÁSQUEZ, D.; SCHARCANSKI, J.; WONG, A. Multi-scale superpixel generation from natural images via hierarchical stochastic graph contraction. **Computer Vision and Image understanding**, Submitted, 2020.

VINCENT, L.; SOILLE, P. Watershed in digital spaces: An efficient algorithm based on immersion simulations. **IEEE Transaction on Pattern Analysis and Machine Intelligence**, v. 6, n. 13, p. 583–598, 1991.

WANG, R.; FEIPING, N.; YU, W. Fast spectral clustering with anchor graph for large hyperspectral images. **IEEE Geoscience and Remote Sensing Letters**, v. 14, n. 11, p. 2003–2007, 09 2017.

WANG, X. et al. A global/local affinity graph for image segmentation. **IEEE Transactions on Image Processing**, v. 24, n. 4, p. 1399–1411, 2015.

WANG, Z. et al. Image classification via object-aware holistic superpixel selection. **IEEE Transactions on Image Processing**, v. 22, n. 11, p. 4341–4352, 2013.

WEI, X. et al. Superpixel hierarchy. **IEEE Transaction on Image Processing**, v. 27, n. 10, 2018.

WEST, D. **Introduction to Graph Theory**. Second edition. [S.l.]: Prentice Hall Inc., 2001.

WONG, A.; SCHARCANSKI, J.; FIEGUTH, P. Automatic skin lesion segmentation via iterative stochastic region merging. **IEEE transactions on information technology in biomedicine : a publication of the IEEE Engineering in Medicine and Biology Society**, v. 15, n. 16, p. 929–936, 05 2011.

WU, C. et al. Improved superpixel-based fast fuzzy c-means clustering for image segmentation. **2019 IEEE International Conference on Image Processing (ICIP)**, p. 1455–1459, 09 2019.

WU, C.; ZHANG L .AND ZHANG, H.; YAN, H. Superpixels using fuzzy simple linear iterative clustering and fast precise number control. **arXiv:1812.10932v2**, v. 15 Oct 2019, 2019.

YAN, D.; HUANG, L.; JORDAN, M. I. Fast approximate spectral clustering. In: **15th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD'09)**. [S.l.: s.n.], 2009. p. 907–916.

YANG, A. et al. Unsupervised segmentation of natural images via lossy data compression. **Computer Vision and Image Understanding**, v. 110, n. 2, p. 212–225, 2008.

YANG, Y. et al. Superpixel-based automatic image recognition for landslide deformation areas. **Engineering Geology**, v. 259, n. 105166, 2019.

YIFENG, J. et al. Exploiting surroundedness and superpixel cues for salient region detection. **Multimedia Tools Applications**, v. 79, p. 10935–10951, 2020.

ZHANG, C. et al. Multiscale fast spectral clustering based on k-d tree. In: **2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)**. [S.l.: s.n.], 2019. p. 1607–1611.

ZHANG, C. et al. Image segmentation based on multiscale fast spectral clustering. **arXiv:1812.04816**, v. 12 Dec 2018, 2018.

ZHANG, J. et al. Spatiotemporal saliency detection based on maximum consistency superpixels merging for video analysis. **IEEE Transactions on Industrial Informatics**, v. 16, n. 1, p. 606 – 614, 2019.

APPENDIX A — EXTENDED ABSTRACT (PORTUGUESE)

Segmentação de imagens é a tarefa de particionar uma imagem digital em múltiplas regiões, as quais representam conjuntos de pixels, também conhecidos como objetos da imagem. O objetivo da segmentação é decompor a imagem em segmentos para seu posterior análise, e a mudança de representação da imagem para um algo mais significativo (SHAPIRO; STOCKMAN, 2001). A segmentação de imagens é uma tarefa importante para muitas aplicações em visão computacional, tais como: segmentação de imagens médicas (DAOUD et al., 2019), detecção de saliências (WANG et al., 2015; NAVA; KYBIC, 2015), (JIN; LI; LI, 2019), (ZHANG et al., 2019), reconhecimento de objetos (LI et al., 2015), (GHADIRI; BERGEVIN; BILODEAU, 2019) e classificação de padrões visuais (WANG et al., 2013), (YANG et al., 2019), mas pode enfrentar problemas desafiadores, tais como o ruído, variabilidade de cor e iluminação; .

Nesse contexto, superpixels são usados para sobre-segmentar imagens, através do agrupamento de pixels que têm propriedades similares, tais como, luminância, cor, ou textura, enquanto é reduzida a informação redundante e o custo computacional. Também, superpixels potencialmente podem ser favoráveis para tratar o problema da escalabilidade de muitas tarefas complexas em visão computacional.

Por tanto, o processo de gerar superpixels em imagens naturais tem quatro grandes desafios a serem enfrentados: i) pobre aderência às bordas dos objetos na imagem, ii) dificuldade para gerar superpixels em condições não ideais, por exemplo, ruído e variabilidade de cor e iluminação, e iii) pobre definição das bordas dos objetos na imagem, e iv) obter representações hierárquicas das regiões da imagem.

A aderência às bordas é uma propriedade importante dos algoritmos de superpixels, na qual os superpixels obtidos preservam as bordas da imagem (ou seja as bordas entre os objetos na imagem). Então, devido a esses desafios nas imagens naturais, a aderência às bordas pode ter um impacto negativo no resultado de tarefas posteriores ao passo de obtenção de superpixels (por exemplo: segmentação, classificação) se o passo de inicialização com superpixels não for adequado (VASQUEZ; SCHARCANSKI; WONG, 2015b).

Métodos de geração de superpixels oferecem um resultado de sobre-segmentação com propriedades desejáveis, tais como: 1) cada pixels esta contido num único superpixel; 2) todos os pixels contidos dentro de um superpixel têm características similares; 3) o número de superpixels gerados é controlável. Ainda é importante adicionar uma

característica adicional, tal como a habilidade de representar o conteúdo da imagem em múltiplas escalas, com preservação do relacionamento espacial entre os superpixels, usando uma estrutura hierárquica (VASQUEZ; SCHARCANSKI, 2018). Assim, os superpixels de escalas finas são contidos nos superpixels nas escalas grosseiras, ou a união de superpixels de escala fina formam um superpixel em escala grosseira. Sendo que os métodos populares para gerar superpixels geram representações em escala simples, tais como (ACHANTA et al., 2012) e (LI; CHEN, 2015), é possível usar eles para conseguir representações multi-escalas mas sem correspondências de bordas e os seus resultados apresentam interseções dos superpixels em escalas grosseiras e finas; por tanto gerar superpixels multi-escala com correspondência das bordas nas múltiplas escalas e com boa aderência as bordas ainda é um problema em aberto e é o foco desta tese.

Como mencionado antes a característica de aderência as bordas é importante no campo dos superpixels, porque quanto mais próximos da borda do objeto estiver a borda do superpixel, melhores resultados se obterão em tarefas mais complexas que dependem dessa característica como no caso de segmentação de imagens, ou aplicações como a segmentação de tumores.

Assim, superpixels podem ser aplicados na segmentação espectral de imagens (SHI; MALIK, 2000), onde a decomposição espectral de uma quantidade grande de elementos ou pixels é muito custosa. Assim, nós podemos usar os superpixels com uma boa aderência as bordas para construir o grafo de adjacência de superpixels e obter os K primeiros autovetores. Finalmente, podemos aplicar algoritmos de agrupamento nos vértices do grafo usando uma abordagem estocástica e a representação dos autovetores como descritores dos vértices do grafo de superpixels.

Nesta tese são propostos *Métodos Estocásticos para Segmentação Multi-escala de Imagens Naturais*, para resolver problemas do processamento de imagens e visão computacional tal como a geração de superpixels e a segmentação espectral de imagens. São desenvolvidos três métodos para geração de superpixels em imagens naturais, usando uma abordagem estocástica para direcionar os principais desafios nesse tipo de imagens, tais como ruido, variabilidade de cor e iluminação; assim como obter resultados com boa aderência as bordas dos objetos e representações multi-escala. Também é desenvolvido um algoritmo para aplicar os superpixels estocásticos na segmentação espectral de imagens naturais, visando diminuir a o custo computacional da decomposição espectral e favorecendo a sua escalabilidade.

A.1 Métodos estocásticos para geração de superpixels

Neste trabalho são propostos métodos estocásticos para geração de superpixels em imagens naturais, uma primeira versão multi-escala chamada *Iterative Hierarchical Stochastic Graph Contraction* (IHSGC) (VASQUEZ; SCHARCANSKI, 2018) considera informação multi-escala e representação hierárquica do conteúdo da sobre-segmentação da imagem com boa aderência as bordas dos objetos. Dois métodos adicionais foram propostos como melhora ao IHSGC, uma versão para gerar superpixels em escalas únicas chamada *Stochastic Graph Contraction* (SGC), e uma versao multi-escala que utiliza como inicialização o resultado do SGC, chamada *Hierarchical Stochastic Graph Contraction* (HSGC). SGC e HSGC superam as métricas do método previamente publicado IHSGC, inclusive em tempo de processamento (VÁSQUEZ; SCHARCANSKI; WONG, 2020b).

A ideia do método IHSGC é gerar superpixels multi-escala com boa aderência as bordas dos objetos na imagem, alem disso oferecer uma representação hierárquica do relacionamento espacial dos superpixels nas escalas finas e grosseiras (superpixels de escalas finas devem estar contidos em superpixels de escalas grosseiras). Alem disso é considerado criar um algoritmo que lide com os desafios próprios das imagens naturais, como o ruído, variação de cor e iluminação, e a pobre definição das bordas dos objetos, devido aos problemas antes mencionados.

O método IHSGC tem um primeiro passo de inicialização que cria uma sobre-segmentação como escala mais fina, esse passo de inicialização é um método estocástico multi-canal proposto em (VASQUEZ; SCHARCANSKI; WONG, 2015b). Em seguida como segundo passo, cada segmento obtido é representado por um histograma 3D normalizado no espaço de cor CIELAB e se constrói o grafo de adjacência dos superpixels G, posteriormente aplicam-se contrações dos vértices desse grafo de forma estocástica para obter múltiplas escalas de representação.

O segundo passo do IHSGC processa as arestas do grafo de superpixels G em ordem ascendente ao peso da aresta e usa uma função de contração dos vértices $\beta(v_i, v_j)$ dado por:

$$\beta(T_i^s, T_j^s) = \exp\left(\frac{-(\chi^2(H_i^s, H_j^s)^2}{\Lambda(T_i^s, T_j^s)}\right), \tag{A.1}$$

onde χ^2 é a distancia qui-quadrado entre os histogramas H_i^s e H_j^s dos respectivos superpixels T_i^s e T_j^s na escala s, e Λ é um termo de penalidade.

Para decidir se dois vértices devem ser contraídos, usa-se um valor aleatório τ que é comparado com o valor de beta, então dois vértices são contraídos se $\tau \leq \beta(T_i^s, T_j^s)$, onde $\tau \in [0, 1]$ é um numero aleatório de distribuição uniforme. T_i^s e T_j^s são os vértices (superpixels) em análise, caso contrario T_i^s e T_j^s não são contraídos. Essa forma de avaliar a contração de dois vértices é expressado como $C_{cs}(T_i^s, T_j^s)$:

$$C_{cs}(T_i^s, T_j^s) = \begin{cases} 1 & , \text{ if } \tau \leq \beta(T_i^s, T_j^s); \\ 0 & , \text{ otherwise .} \end{cases} \tag{A.2}$$

O método IHSGC obteve bons resultados na geração de superpixels com aderência as bordas dos objetos na imagem quando comparado com métodos populares do estado-da-arte, mas com a característica de obter representações multi-escala da sobre-segmentação da imagem. A desvantagem é o maior custo computacional devido ao calculo dos histogramas para representar os superpixels, e o cálculo da distancia qui-quadrado para o ordenamento das arestas do grafo de superpixels.

Adicionalmente, dois métodos estocásticos para gerar superpixels foram propostos, um específico para gerar escalas únicas de representação, apenas recebendo o número de superpixels desejado como parâmetro ed entrada, chamado *Stochastic Graph Contraction* (SGC), e outro como versão multi-escala do SGC, chamado *Hierarchical Stochastic Graph Contraction* que utiliza o SGC como passo de inicialização.

Os métodos SGC e HSGC são uma melhora do IHSGC em termos de métricas de boundary recall e under-segmentation error (NEUBERT; PROTZEL, 2012), também do tempo de processamento sendo melhores (ou comparáveis) aos métodos populares e atuais do estado-da-arte. Especificamente as melhoras estão relacionadas a utilizar feições simples para representar como a cor média dos pixels contidos no superpixel, também a restrições de tamanho na formação do superpixels, por exemplo para gerar N^d superpixels (número desejado), o tamanho desejado será definido como N_0/N^d, onde N_0 é o número de pixels da imagem. Alem disso se tem um grafo relativamente mais completo pois se utiliza a conectividade dos 8 vizinhos na representação inicial do grafo nos pixels da imagem. O método SGC visa ser um método rápido para geração de superpixels de escala única com o número de superpixels desejado como único parâmetro de entrada; e o HSGC pretende ser um método de geração de superpixels multi-escala que pode oferecer uma representação hierárquica do relacionamento espacial dos superpixels nas escalas finas e grosseiras. Ambos métodos seguem uma abordagem estocástica que consegue lidar com os desafios próprios das imagens naturais, como as variações de cor, iluminação e ruido.

A.2 Método estocástico para segmentação espectral de imagens

É proposto um método estocástico para segmentação espectral de imagens naturais; dissemos segmentação espectral devido ao uso da decomposição em autovetores a partir da matriz laplaciana normalizada do grafo de adjacência de superpixels. Esse método é chamado de *Spectral Stochastic Graph Contraction* (SSGC) e é proposto como aplicação do uso dos superpixels estocásticos (VÁSQUEZ; SCHARCANSKI; WONG, 2020a). Contudo, SSGC é proposto para mostrar que é possível desenvolver uma abordagem estocástica que utilize informação dos autovetores da matriz laplaciana normalizada, e demonstrar que essa abordagem obtêm resultados superiores ao método tradicional que realiza a contração dos vértices diretamente nos valores das cores dos superpixels.

O SSGC utiliza como passo de pre-processamento o método de geração de superpixels SGC (ver Seção A.1), procurando reduzir o custo computacional da obtenção dos autovetores, pois é mais custoso fazer a decomposição de uma matriz de tamanho $N_0 \times N_0$ do que $N \times N$, onde N_0 é o número de pixels na imagem, e N é o número de superpixels, respectivamente.

Depois de obter os superpixels da imagem, nós representamos o seu relacionamento espacial usando um grafo de adjacência G, e cada superpixel é descrito usando um histograma normalizado 3D no espaço de cores CIELAB. A matriz de similaridade W é obtida usando a seguinte função de similaridade:

$$w_{ij} = \exp\left(-\frac{\chi^2(H_i, H_j)}{0.05}\right), \tag{A.3}$$

onde H_i e H_j são dois histogramas normalizados 3D de cor de tamanho $24 \times 24 \times 24$ e representam o conteúdo de cada superpixel v_i e v_j, respectivamente; e $\chi^2(H_i, H_j)$ é a distancia qui-quadrado entre os histogramas.

Com a matriz de similaridade, calculamos a matriz laplaciana normalizada L_{rw} para obter seus K menores autovetores e formar a matriz U. A matriz Laplaciana normalizada pode ser definida como $L_{rw} = D^{-1}L = I - D^{-1}W$ (See Section 2.3) (LUXBURG, 2007). Então é calculada a decomposição espectral de $L_{rw}U = \lambda U$, onde os autovetores são as colunas $\{u_{ij}|i = 1, \ldots, N_{sp}\}$ da matriz U e λ_i são seus correspondentes autovalores, e os K menores autovetores $\{u_{ij}|i = 1, \ldots, K\}$ definem o sub-espaço $N_{sp} \times K$ no qual os superpixels são representados e agrupados.

SSGC é um método que aplica operações de contração de grafos para gerar as regiões do resultado final da segmentação. Onde $\beta(v_i, v_j)$ é definido como uma função likelihood de contração de vértices v_i e v_j que compartilham a aresta $e_m(v_i, v_j)$, que avalia se esses vértices devem ser contraídos ou não:

$$\beta(v_i, v_j) = \exp\left(-\frac{\sum_{k=1}^{k=K}(u_{ik} - u_{jk})^2}{\Lambda(v_i, v_j)}\right), \tag{A.4}$$

Onde Λ é uma função de penalidade da contração e é definida como segue:

$$\Lambda(v_i, v_j) = \frac{\ln(N_0)}{Q}\left(\frac{1}{|v_i|} + \frac{1}{|v_j|}\right), \tag{A.5}$$

Onde N_0 é o número de pixels na imagem original, $|v_i|$ é o número de elementos (pixels) no vértice v_i, e Q é um parâmetro ed regularização que controla o processo de agrupamento (definido como $Q = 200$ para os experimentos e é atualizado por $Q = Q/2$ quando não é possível contrair mais vértices do grafo, por exemplo quando tem poucas regiões muito diferentes).

As arestas $e_m(v_i, v_j)$, do grafo de superpixels $G(V, E)$, são processadas em ordem ascendente do seu peso w_{ij} (ver Eq. A.6), e quando dois vértices adjacentes são contraídos, os autovetores são calculados novamente usando o grafo $G'(V', E')$ com o novo vértice. Os pesos w_{ij} são calculados usando os valores dos autovetores para representar cada superpixel (vértice) $\{u_{i1}, ..., u_{iK}\}$, $i = 1, .., N_{sp}$:

$$w_{ij} = \sqrt{\sum_{k=1}^{k=K}(u_{ik} - u_{jk})^2}. \tag{A.6}$$

Se $\tau \leq \beta(v_i, v_j)$, onde $\tau \in [0, 1]$ é um numero aleatório, os vértices v_i e v_j são contraídos, caso contrario não são contraídos. Esse criterio de contracao é definido como no predicado de contracao $C_{spectral}(v_i, v_j)$ a seguir:

$$C_{spectral}(v_i, v_j) = \left\{ \begin{array}{ll} 1 & , \text{ if } \beta(v_i, v_j) \geq \tau; \\ 0 & , \text{ otherwise .} \end{array} \right\}. \tag{A.7}$$

As características importante do método SSGC são: i) usa superpixels estocásticos com boa aderência as bordas dos objetos na imagem, como passo de pre-processamento; ii) usa feições simples para computar a matriz de similaridade; e iii) define operações de contração de vértices no espaço dos autovetores, construídos a partir do grafo de adjacên-

cia dos superpixels.

O método SSGC tem um alto custo computacional devido ao cálculo dos autovetores depois de cada contração, gerando maior tempo de processamento para uma quantidade grande de superpixels. Por tanto, para melhorar a performance em tempo foi definido uma versão eficiente ($SSGC_E$) ou com melhor performance em tempo, na qual os valores dos autovetores são atualizados pela media ponderada nos vértices a serem contraídos.

A.3 Resultados e conclusões

Nesse trabalho, foram propostos métodos estocásticos multi-escala para segmentação de imagens naturais, principalmente na geração de superpixels multi-escala, e adicionalmente na geração de superpixels estocásticos de escala única com aplicação na segmentação espectral. Finalmente foi proposto um método de segmentação espectral baseado em contrações estocásticas de grafo.

As principais contribuições são:

- Uma publicação em conferencia internacional qualis A2 (2015 IEEE I2MTC), relacionada ao método de sobre-segmentação estocástica multi-camada usado como primeiro passo do método IHSGC (VASQUEZ; SCHARCANSKI; WONG, 2015b).
- Uma publicação em revista internacional qualis A1 (Expert Systems with Applications), relacionada ao método de geração de superpixels multi-escala com representação hierárquica e correspondência das bordas entre superpixels nas escalas de representação (IHSGC), alem disso obtendo boa aderência as bordas dos objetos da imagem quando comparado aos métodos do estado-da-arte (VASQUEZ; SCHARCANSKI, 2018).
- Um artigo submetido à revista internacional Computer Vision and Image Understanding (qualis A1), relacionado aos métodos de geração de superpixels SGC e HSGC, como melhora do IHSGC (VÁSQUEZ; SCHARCANSKI; WONG, 2020b).
- Um artigo submetido a revista internacional Pattern Recognition (qualis A1), relacionado ao método de segmentação espectral SSGC (VÁSQUEZ; SCHARCANSKI; WONG, 2020a).

Em resumo, temos apresentado métodos estocásticos para gerar superpixels em representação de escala única (SGC) e em multiplas-escalas (IHSGC e HSGC). Esses métodos tem características importantes como: i) uso de contrações de grafos para agru-

pamento de feições visuais de forma hierárquica; e ii) oferecer uma representação hierárquica do grafo dos superpixels com varias escalas de representação e ao mesmo tempo preservar a correspondência das bordas e o relacionamento espacial dos superpixels nas escalas finas e grosseiras.

Resultados quantitativos são apresentados a seguir, onde os métodos propostos são avaliados usando as métricas de aderência à bordas (BR) e erro de sub-segmentação (UE) definidos em (NEUBERT; PROTZEL, 2012). Figs. A.1 e A.2, mostram os resultados quantitativos dos métodos avaliados usando os banco de imagens BSDS300 e BSDS500 (ARBELAEZ et al., 2011), respectivamente. Em ambas figuras temos o resultado da aderência às bordas no lado esquerda, e a métrica do erro de sub-segmentação no lado direito; e podemos concluir que os métodos propostos IHSGC, SGC e HSGC obtêm resultados superiores ou comparáveis com os métodos do estado-da-arte.

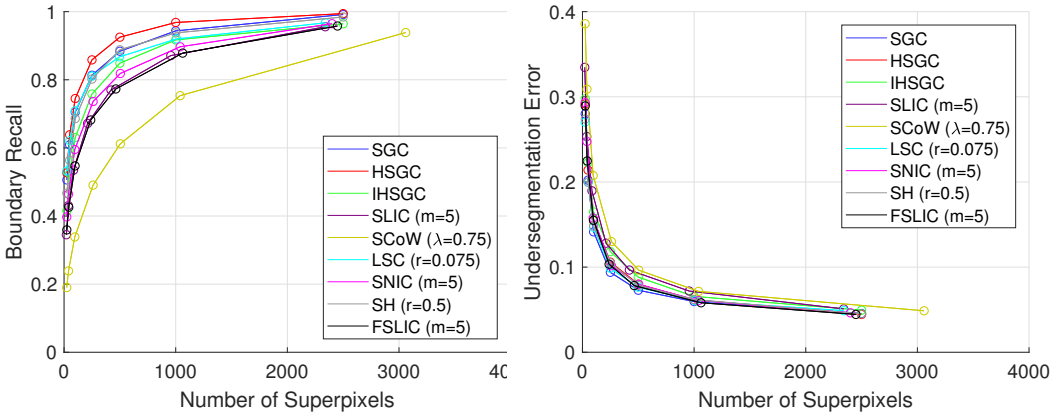
HSGC (curva vermelha) se mostra superior a todos os métodos em termos de aderência às bordas dos objetos na imagem, o qual indica que representa uma boa opção para aplicações que precisem representar o conteúdo da imagem em múltiplas escalas desde o passo de pre-processamento, podendo ser útil também na criação de novos descritores multi-escala de regiões, como passo prévio a um processo de segmentação ou classificação. O método SGC mostra melhor performance para ambas métricas e ainda superior aos métodos do estado-da-arte.

A Fig. A.3 mostra o tempo de execução dos métodos avaliados para os banco de imagens BSDS300 (lado esquerdo) e BSDS500 (lado direito), respectivamente. Nessa figura não se mostra o IHSGC porque é o método mais demorado, devido ao custo computacional para obter os histogramas de cor 3D, as distancias qui-quadrado, e o ordenamento das arestas do grafo de superpixels. Considerando que os métodos propostos SGC e HSGC foram desenvolvidos em MATLAB, eles apresentam desempenho em termos de tempo de execução comparável aos métodos do estado-da-arte os quais foram desenvolvidos em C++ (código fonte dos autores).

Finalmente, foi apresentado um método de segmentação estocástica espectral de imagens naturais, como uma aplicação dos superpixels gerados por SGC, e para mostrar que uma abordagem estocástica pode obter bons resultados quando usada no contexto de segmentação espectral.

Nas tabelas A.1 , A.2 e A.3 se mostram os resultados quantitativos para o metodo proposto e métodos do estado-darte, que realizam um passo de inicialização usando numero de superpixels igual a 30, 60 e 120 respectivamente. Nessas tabelas, se mostram

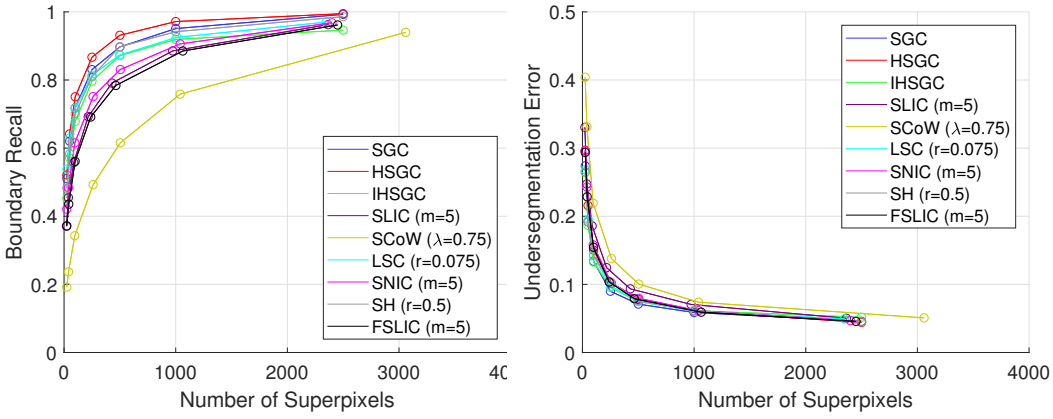
Figure A.1: Métricas de (a) aderência às bordas e (b) erro de sub-segmentação para métodos propostos e métodos do estado-da-arte, usando o conjunto de teste do banco de imagens BSDS300.



(a) BR (b) UE

Fonte: O Autor.

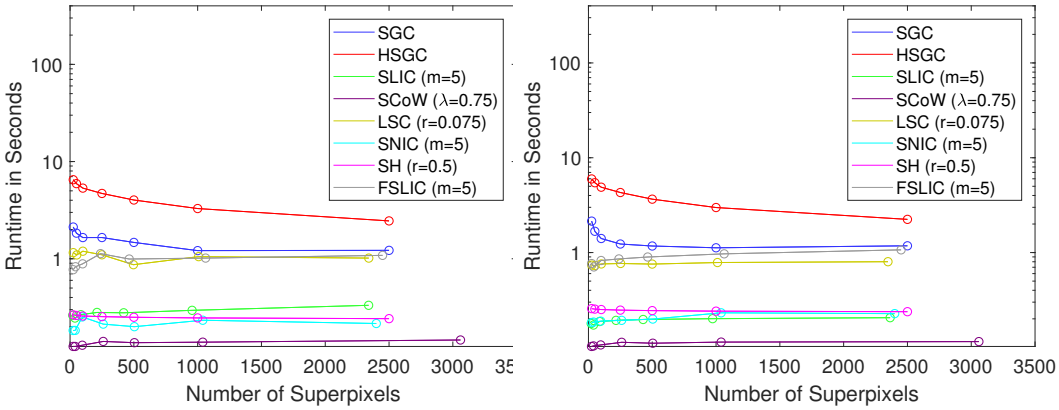
Figure A.2: Métricas de (a) aderência às bordas e (b) erro de sub-segmentação para métodos propostos e métodos do estado-da-arte, usando o conjunto de teste do banco de imagens BSDS500.



(a) BR (b) UE

Fonte: O Autor.

Figure A.3: Tempo de execução dos métodos propostos, e métodos do estado-da-arte, usando o conjunto de teste do banco de imagens (a) BSDS300 e (b) BSDS500.



(a) Tempo de execução BSDS300 (b) Tempo de execução BSDS500

Fonte: O Autor.

os resultados quantitativos para dois banco de imagens BSDS300 e BSDS500, e quatro métricas segmentation covering (COV), probabilistic rand index (PRI), volume of information (VoI) (ARBELAEZ et al., 2011) e tempo de execução. Os métodos avaliados são: o método proposto SSGC, a sua versão rápida $SSGC_E$, Stochastic Region Merging initialized by SLIC superpixels (SRM) (WONG; SCHARCANSKI; FIEGUTH, 2011), Normalized Spectral Clustering (NCUT) initialized by SLIC superpixels (SHI; MALIK, 2000), Superpixel-based Fast Fuzzy C-Means Clustering (SFFCM) (LEI et al., 2018), Improved Superpixel-based Fast Fuzzy C-Means Clustering (ISFFCM) (WU et al., 2019). Dos resultados podemos concluir que o uso de superpixels como método de inicialização reduz o tempo de processamento, levando aproximadamente $5s$ para o método proposto gerar uma segmentação de 20 regiões; também o uso de uma abordagem estocástica junto a informação dos autovetores traz melhora nos resultados das métricas do que os métodos tradicionais.

Table A.1: Comparação dos algoritmos de segmentação de imagens coloridas para os datasets BSDS300 e BSDS500 usando 30 superpixels como passo de inicialização.

Método	BSDS300			BSDS500			Time (s)
	COV (↑)	PRI (↑)	VoI (↓)	COV (↑)	PRI (↑)	VoI (↓)	
SFFCM	0.55	0.78	2.07	0.57	**0.80**	1.98	5.10
ISFFCM	0.48	0.76	2.41	0.49	0.78	2.37	3.00
NCUT	0.51	0.76	2.17	0.54	0.78	2.12	2.10
SRM	0.50	0.76	2.29	0.50	0.77	2.29	**1.80**
SSGC	**0.61**	**0.79**	**1.76**	**0.62**	**0.80**	**1.76**	7.20
$SSGC_E$	0.59	0.78	1.88	0.61	**0.80**	1.84	5.40

Table A.2: Comparação dos algoritmos de segmentação de imagens coloridas para os datasets BSDS300 e BSDS500 usando 60 superpixels como passo de inicialização.

Method	BSDS300			BSDS500			Time (s)
	COV (↑)	PRI (↑)	VoI (↓)	COV (↑)	PRI (↑)	VoI (↓)	
SFFCM	0.55	0.78	2.07	0.57	**0.80**	1.98	5.10
ISFFCM	0.46	0.75	2.62	0.49	0.78	2.49	4.50
NCUT	0.54	0.77	2.08	0.55	0.78	2.06	3.30
SRM	0.52	0.77	2.24	0.52	0.77	2.29	**2.10**
SSGC	0.59	0.77	**1.84**	0.59	0.77	**1.82**	15.00
$SSGC_E$	**0.60**	**0.79**	1.90	**0.60**	**0.80**	1.88	7.80

Table A.3: Comparação dos algoritmos de segmentação de imagens coloridas para os datasets BSDS300 e BSDS500 usando 120 superpixels como passo de inicialização.

Method	BSDS300			BSDS500			Time (s)
	COV (↑)	PRI (↑)	VoI (↓)	COV (↑)	PRI (↑)	VoI (↓)	
SFFCM	0.55	0.78	2.07	0.57	0.80	1.98	5.10
ISFFCM	0.45	0.75	2.78	0.49	0.78	2.62	6.90
NCUT	0.58	0.78	1.93	0.59	0.80	1.94	3.00
SRM	0.54	0.78	2.12	0.54	0.78	2.21	**2.40**
SSGC	0.58	0.78	1.90	0.59	0.78	1.87	31.80
$SSGC_E$	**0.61**	**0.79**	**1.85**	**0.63**	**0.81**	**1.78**	10.80