

**NEW ALGORITHMS FOR APPEARANCE MODELING  
IN IMAGE SEGMENTATION**

JEOVÁ FARIAS SALES ROCHA NETO

*M. Sc. in Applied Mathematics, Brown University, 2021*

*M. Sc. in Computer Science, University of Nice-Sophia Antipolis, 2015*

*B. Eng. in Telematics Engineering, Federal University of Ceará, 2016*

A THESIS SUBMITTED FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

BROWN UNIVERSITY

2021

© Copyright 2021 by Jeová Farias Sales Rocha Neto

This dissertation by Jeová Farias Sales Rocha Neto is accepted in its present form by the School of Engineering as satisfying the dissertation requirement for the degree of Doctor of Philosophy.

Date: \_\_\_\_\_, \_\_\_\_\_

Pedro F. Felzenszwalb, advisor.

Recommended to the Graduate Council

Date: \_\_\_\_\_, \_\_\_\_\_

Matthew T. Harrison, reader.

Date: \_\_\_\_\_, \_\_\_\_\_

Benjamin B. Kimia, reader.

Approved by the Graduate Council

Date: \_\_\_\_\_, \_\_\_\_\_

Andrew G. Campbell,  
Dean of the Graduate School.

## Vita

Jeová Farias Sales Rocha Neto received his B.Eng. degree in Telematics Engineering with emphasis in Computer Engineering in the Federal University of Ceara, Fortaleza, Brazil, in January 2016. His undergraduate thesis research consisted in statistical level set methods for Synthetic Aperture Radar (SAR) image segmentation. His undergraduate thesis advisor was Professor Fátima Nelsizeuma Sombra De Medeiros.

Before completing his undergraduate degree, he had received a M. Sc in Computer Science with emphasis in Video, Image and Multimedia from the University of Nice-Sophia Antipolis, Nice, France in 2015. There he studied 3D shape classification methods using 2D projections. His research advisor was Professor Marc Antonini.

In 2016, he joined Brown University to pursue a graduate studies at Brown's School of Engineering and became a member of the Laboratory for Engineering Man/Machine Systems (LEMS). At Brown he studied under Professor Pedro Felzenszwalb in the School of Engineering, received a Sc.M. in Applied Mathematics in 2021 and (with the completion of this thesis) a Ph.D. in Electrical and Computer Engineering also in 2021. His main research interest is in discrete optimization methods for unsupervised image segmentation and clustering.



## Acknowledgments

Firstly, I would like to thank my advisor Prof. Pedro Felzenszwalb for guiding me in my work over these past 5 years. I specially thank him for his patience with me during some personal and professional hardships I faced in my PhD. Without it, I wouldn't be able to finish this thesis on time and complete my degree. I personally very much admire his qualities as a researcher: his attention to details, his quick intuition, his impressive knowledge, his ability to write simply and objectively and, specially, his joy at studying computer vision or machine learning problems. In many ways, that inspires me very deeply as a researcher and I hope to develop some of these traits during my professional career. I also thank Prof. Felzenszwalb for having introduced to me his wife, Prof. Carly Klivans, and their children, Aaron and Audrey. We thankfully shared very good, fun and memorable moments all together.

I would like to thank the examination committee including Prof. Benjamin Kimia and Prof. Matthew Harrison. Their feedback, comments, and criticism are critical to improving this work.

I thank my academic sister, Anna Grim, for having introduced me to Summer@Brown and to APMA's Directed Reading Program, where I could practice my teaching and advising skills and through which I revived my passion for the academic life. I also thank my lab mates, Berk, Hongyi and Peter for the good conversations, for the fun places we visited and for the fun games we played. I thank my academic collaborators, Alice and Marilyn, for the research discussions and their kindness towards me. I also take the opportunity to thank Prof. Basilis Gidas for his warm welcome when I arrived at Brown and for all classes I took with him.

I thank the friends at made at Brown-RISD Catholic Community. I thank Father Albert Duggan, O.P. for the sacraments he administered to me, they became the source of grace I needed to resist the temptation of giving up. I also thank him for answering my hard spiritual questions, for talking to me about Thomistic themes and for being present when I needed. I thank all FOCUS missionaries I met along the way: Kevin, Catherine, Josh, Anna, Mark, Jessica, Angela, Nick and Zoe. In particular, I thank Kevin for being my best American friend, whom I greatly admire, and for being there always. I am also very thankful to the community at Church of St. Mary on Broadway, my home parish. There I made many incredible friends and was able better appreciate the Traditional Catholic Liturgy. I also thank Father John Berg, F.S.S.P., Father William Rock, F.S.S.P. and Father John Kodet, F.S.S.P. for being so spiritually supportive in my struggles during the pandemic. I finally thank Father George Crafts for being my spiritual director over all these years and for teaching me how to work better and sanctify my ordinary life.

I thank my wife and best friend, Leidiane, for her immense support and for being next to me when I felt I couldn't go on anymore. Since we started dating, she has been in some many ways the light that brings joy to me and my greatest reason to move forward and become a better man. I thank my parents, Juscelino and Inês, for their presence, despite the distance, and their prayers for my work. I hope one day I can be the son they deserve. I also thank my sister, Joana Maria, for her support and friendship. A great woman, whom I profoundly admire and whom miss very much since I moved to America.

Finally but most importantly, I thank God for His constant love, His (sometimes difficult) pedagogy, for His abundant Providence that sustained His unworthy servant in the past 5 years.

# Contents

<b>List of Figures</b>	<b>xi</b>
<b>List of Tables</b>	<b>xvi</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation	1
1.1.1 Applications	2
1.1.2 Challenges	3
1.2 Organization of this thesis	4
<b>2 Graph-based Image Segmentation and Spectral Clustering</b>	<b>7</b>
2.1 Graph-based Methods for Image Segmentation	7
2.1.1 Bayesian Formulation	8
2.1.2 Markov Random Fields and the Gibbs Formalism	9
2.1.3 Graph Cuts	11
2.2 Appearance Modeling	15
2.2.1 User interaction	16
2.2.2 Iterative Methods	17
2.2.3 Variational Approaches	19
2.2.4 Implicit Modeling	22
2.2.5 Factorization-Based Methods	23
2.3 Spectral Solvers	25
2.3.1 Normalized Cuts	25
2.3.2 Weight choice	28
2.3.3 Spectral clustering and random walks	29
2.3.4 Practical Considerations	30

2.3.5	Adding constraints to Normalized cuts	30
2.3.6	Multiview Spectral Clustering	31
<b>3</b>	<b>Direct Estimation of Appearance Models</b>	<b>33</b>
3.1	Appearance Models and Image Segmentation	33
3.2	Image Statistics	34
3.3	Appearance Estimation	39
3.3.1	Algebraic Method	39
3.3.2	Spectral Method	42
3.3.3	Estimating $w_0$ , $w_1$ and $\epsilon_r$	44
3.4	Multi-region case	45
3.4.1	Estimation when $r$ is small	46
3.4.2	Estimation when $r$ is large	48
3.5	Examples	49
3.6	Numerical Experiments	49
3.6.1	Evaluation Measures	49
3.6.2	Synthetic Data	53
3.6.3	Evaluating the effect of $\rho$	53
3.6.4	Appearance Model Evaluation on Synthetic Images	54
3.6.5	Segmentation Evaluation on Synthetic Images	56
3.6.6	Real Images	59
3.6.7	Experiments using the tensorial method and on multi-region images	63
3.7	Conclusion	66
<b>4</b>	<b>Spectral Image Segmentation with Global Appearance Modeling</b>	<b>67</b>
4.1	Drawbacks of the Traditional Graph Construction	67
4.2	New Criteria for Image Segmentation	68
4.2.1	Spatial Information: $G_{\text{grid}}$	69
4.2.2	Global Appearance Information: $G_{\text{data}}$	70
4.2.3	Combining Spatial and Appearance information	72
4.3	Segmentation Algorithm	72

4.3.1	Spectral Method	72
4.3.2	Graph Sparsification	73
4.4	Numerical Experiments	75
4.4.1	Segmentation accuracy measure and hardware setting	75
4.4.2	Sparsification algorithm for NCut	75
4.4.3	Evaluation of NCut and MixNCut without sparsification	75
4.4.4	Impact of edge sampling on MixNCut	76
4.4.5	The role of $\lambda$ on MixNCut	76
4.4.6	Experiments in Real Images	79
4.4.7	Experiments in Synthetic Images	81
4.5	Conclusion	85
<b>5</b>	<b>Penalized Normalized Cuts</b>	<b>86</b>
5.1	Prior work and its limitations	86
5.2	Adding the Penalty	88
5.3	Possible Penalties and Related Segmentation Cues	90
5.3.1	Seeds	90
5.3.2	Region Color Histograms	90
5.3.3	Region Mean Colors	91
5.3.4	Global Appearance Models	92
5.3.5	Combining Cues and Summary	93
5.4	Preliminary Results	93
5.4.1	Synthetic Experiments	94
5.4.2	Real experiments	100
5.5	Conclusion	104
<b>6</b>	<b>Conclusion</b>	<b>106</b>
6.1	Contributions	106
6.1.1	Contributions to appearance modelling	106
6.1.2	Contributions to spectral image segmentation	107
6.2	Future Work	109

6.2.1	MRF based segmentation and appearance modeling	109
6.2.2	Spectral image segmentation	111
	<b>References</b>	<b>112</b>

# List of Figures

1.1	Segmentation examples	2
2.1	Graph Cut algorithm. Adapted from [BFL06]	13
2.2	The graph used in the $\alpha$ -expansion algorithm for a 1D image. The nodes representing the image pixels are in black and their current graph partitions are shown in gray. The terminal nodes are depicted as $\alpha$ and $\bar{\alpha}$ . The node $a$ is added in between every pair of nodes whose region assignments are different.	14
2.3	The graph used in the $\alpha\beta$ -swap algorithm for a 1D image. The nodes representing the image pixels are in black and their current graph partitions are shown in gray. The terminal nodes are depicted as $\alpha$ and $\beta$ .	15
2.4	Typical user interactions. Adapted from [TBAMB15]	15
2.5	Model estimation and segmentation using ALT. On the bottom, we see how both foreground and background color distributions estimated by ALT (—) evolve compared to the ground truth appearance models (—). The evolution of the segmentations given the models is shown on top.	17
2.6	Grabcut segmentation results. Adapted from [RKB04]	19
2.7	Smoothed Heaviside function for various values of $\epsilon$ .	22
2.8	Linear combination of spectral histograms. Adapted from [YWC15].	24
2.9	Segmentation result using the original Normalized Cuts formulation. Adapted from [SM00].	29
2.10	Biased Normalized Cuts result. Adapted from [MVM11].	31
3.1	Evaluating independence at a distance. We show the average Bhattacharyya distance between $\hat{\beta}$ and $\hat{\alpha}\hat{\alpha}^\top$ as a function of $r$ for images with a single Brodatz texture (see Figure 3.7).	36

3.2	Evaluating independence at a distance. We proceed as in Figure 3.1, but for individual textures (on the left).	37
3.3	Evaluating independence at a distance. We show the average distance between $\hat{\beta}$ and $\hat{\alpha}\hat{\alpha}^\top$ as a function of $r$ for each region segment of the images in the Grabcut Dataset [RKB04].	37
3.4	Evaluating independence at a distance. We proceed as in Figure 3.3, but now for individual images in the Grabcut Dataset and for their respective foreground and background segments.	38
3.5	The area where pairs of pixels with $\ x - y\  = r$ can be in different regions.	43
3.6	Estimation of appearance models with $\rho = 0.06$ . In (a) we show the input images and their ground truth segmentation. In (b) and (c) we show the appearance models computed using the ground truth segmentation in blue (—), the algebraic method in green (—), the spectral method in red (—) and the tensorial method (—). The images are from the Berkeley Segmentation [MFTM01] dataset.	50
3.7	Selected Brodatz patterns	51
3.8	Ground truth segmentations used to generate synthetic data.	52
3.9	Examples of synthesized images with textures.	52
3.10	Average appearance model estimation error ( $D_B$ ) as a function of $\rho$ on images composed of IID (—) and Brodatz (—) patterns disposed as in GT1 and GT2. For both (a) and (b) the results on the left are from the algebraic method, whereas the results on the right are from the spectral method.	54
3.11	Qualitative segmentation results: (a) original image and its ground truth segmentation, (b) algebraic method, (c) spectral method, (d) ALT, (e) LSWD, (f) ORTSEG, (g) FBS, and (h) PNMF.	57
3.12	Sample images from the Weizmann Segmentation Evaluation Database (SED, shown in the first row) [AGBB11] and from the Singapore Whole sky Nighttime Image SEGmentation Database (SWINSEG, shown in the second row) [DSLW17].	61
3.13	Application our methods in natural scenes. The blue and red contours are the results of segmentation using appearance models estimated using the algebraic and spectral methods, respectively.	62



3.14	Ground truth segmentations used to generate the multi-region synthetic data.	64
3.15	Examples of synthetic textured images generated by the ground truth segmentation in Figure 3.14.	64
4.1	The edges connecting to a pixel $i$ in $G_{\text{grid}}$ and $G_{\text{data}}$ . In the each image, the thickness of each link is proportional to its weight.	69
4.2	Evaluation of NCut-Graylevel and MixNCut performances without sampling. Column (a) show the test images of size $40 \times 40$ , with different levels of noise. In column (c) we show the value of $J$ for a range of values for $\sigma_I$ and $\sigma_X$ . In (c), we show evaluate the segmentation performance of MixNCut over various combinations of $\sigma$ and $\lambda$ . In these experiments NCut averaged $6.31 \pm 5.28$ s of processing time and MixCut, $1.35 \pm 0.25$ s.	77
4.3	Evaluation of sampling performance for various $\alpha$ . Column (a) shows two different test images of size $200 \times 200$ . Column (b) shows the average and standard deviation of $J$ segmentation measure for 100 runs of MixCut on the respective test image. Column (c) shows the average time and standard deviation of the same runs. Here, we set $\lambda = 0.95$ and $\sigma = 1$ .	78
4.4	MixNCut results when varying $\lambda$ . Column (a) shows the input image. Columns (b)-(d) show the computed eigenvectors (on the left) and segmentations (on the right) given by MixNCut for various values of $\lambda$ and $\sigma = 1$ .	78
4.5	Segmentation results comparing NCut and MixNCut on real images. Column (a) shows the input images. Column (b) shows the eigenvector found by the original NCut formulation on the left and the segmentation result on the right. Column (c) shows the eigenvector found by the new MixNCut formulation on the left and the segmentation result on the right.	80
4.6	Segmentation results using the proposed method for images with more than 2 regions.	81

- 4.7 Comparing NCut-Graylevel, NCut-Gabor, and MixNCut on textured images. Column (a) shows the input images. Column (b) shows the eigenvector found by the original NCut formulation on the left and the segmentation result on the right. Column (c) shows the eigenvector found by NCut with Gabor features on the left and the segmentation result on the right. Column (d) shows the eigenvector found by the new MixNCut formulation on the left and the segmentation result on the right. 83
- 4.8 Brodatz Patterns used in the synthetic experiments 84
- 5.1 Datasets used in our synthetic experiments and generalized eigenvectors computed from the traditional NCut formulation. In each experiment, the leftmost set is  $S_1$ , the central one is  $S_2$  and the rightmost one is  $S_3$ . 96
- 5.2 Results for NCut penalized with seeds information. In (a) and (c), we show the initial datasets with the selected seeds circled in red and blue. In (b) and (d), the computed eigenvectors are shown: on left, for the generalized eigenvalue problem with dense matrices and, on the right, the same problem but without the explicit computation of these matrices. The elapsed time of each solver is also presented. 97
- 5.3 Results for NCut penalized with the histogram disparity cue. In (a) and (c), one color out of two possible is assigned to each datapoint. In (b) and (d), the generalized eigenvectors computed using the histogram disparity cue are shown in the same manner as described in Figure 5.2. 98
- 5.4 Results for NCut penalized with the color disparity cue. In (a) and (c), a random RGB color around blue or red is assigned to each datapoint. In (b) and (d), the generalized eigenvectors computed using the mean color disparity cue are shown in the same manner as described in Figure 5.2. 99
- 5.5 Results for for NCut penalized with the appearance model data. In (a) and (c), a appearance model probability value is assigned to each datapoint according to the set it belongs to. On the image, their log-ratio values are depicted. In (b) and (d), the generalized eigenvectors computed using the appearance model data are shown in the same manner as described in Figure 5.2. 100

- 5.6 Segmentation results of a blank image when seeds are added to it. In (a) the proposed seeds are shown in white (background seeds) and dark gray (foreground seeds). The remaining black pixels are unseeded. In (b), we show the resulting penalized normalized cut eigenvector when the seeds penalty is applied ( $\alpha = 1$ ). The resulting segmentation when the eigenvector in (b) is clustered via  $K$ -means is depicted in (c). Image frames were added for visibility. The seed images are from the Grabcut dataset [RKB04]. 101
- 5.7 Segmentation results of a real image when seeds are added to it. In (a) the proposed seeds are shown in blue (background seeds) and green (foreground seeds) on the original image. In (b), on the left we show the normalized cut eigenvector without the use of the seed information and on the right the final segmentation. In (c), on the left we show the penalized normalized cut eigenvector when the seeds penalty is applied ( $\alpha = 1$ ) and on the right we show the final segmentation. Images are from the Grabcut dataset [RKB04]. 102
- 5.8 Penalized Normalized Cut Eigenvectors from different segmentation cues/penalties. In (a) we show the original images corrupted with Gaussian noise. The eigenvectors computed from the histogram disparity, color disparity and appearance models cues are shown in (b), (c) and (d), respectively. 105

# List of Tables

3.1	Average $D_B$ distance between estimated and ground truth appearance models on the synthetic data generated using different segmentation masks. We evaluate our algorithms using different methods for selecting $w_0$ , $w_1$ and $\epsilon$ (see text).	55
3.2	Average $J$ index of different segmentation methods on the synthetic data generated using different segmentation masks.	58
3.3	Comparative Segmentation Performance on the SED1 Database.	61
3.4	Comparative Segmentation Performance on the SWINSEG Dataset.	61
3.5	Average performance measures for estimation and segmentation results on the multi-region synthetic data generated using different ground truth segmentations.	65
4.1	Comparative Segmentation Performance on the SED1 Database.	82
4.2	Comparative Segmentation Performance on the SWINSEG Dataset.	82
4.3	Evaluation of different segmentation methods on textured images. The table summarizes accuracy and running time of each method on images with different ground-truth segmentations.	84
5.1	Summary of proposed penalties for the Penalized Normalized Cut formulation	94
5.2	Comparative Segmentation Performance on the Grabcut Dataset for the penalized and the traditional normalized cuts algorithms.	103

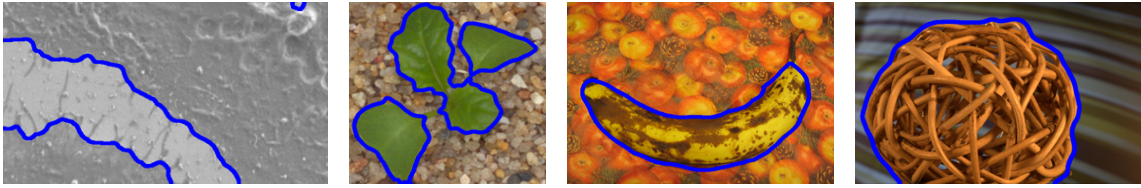
# CHAPTER 1

## Introduction

### 1.1 Motivation

One of the main features of modern society is its technological advancement. Thanks to that, the amount of visual data, mostly in the form of images, one has access to is incomparable to precedent ages and that plays a more and more important role in our lives. Furthermore, images have the ability to record scenes, which can then be stored and studied. Over the years, that opened space to immense advances in diverse fields, once barely explored by scientists and engineers. In medicine, the creation of Magnetic Resonance Imaging (MRI), Computed Tomography (CT), ultrasound, etc. enabled medical diagnostics and prescriptions to be more efficient and assisted the discovery of the cure of many challenging diseases. In remote sensing, satellite and airborne imagery became crucial to forestall surveillance and political decision making to what concerns nature preservation. In telecommunications, the onset of digital images made long distance face-to-face human interactions possible, from affordable video calls to world wide virtual courses.

Along side with the advancements on the acquisition of visual data, there is the process of extracting useful information from it. This task is usually very challenging to be accomplished automatically, despite it being natural to our visual cortex. Therefore, the field of computer vision arose as the study of the means by which one can automate the interpretation of digitized visual scenes, which requires action on different levels. Low level vision is concentrated in pre-processing the image for further understanding or extracting features from it. High level vision focus its attention on identifying the semantics of a scene, i.e, which objects are present, what is their context, how do they related to one another,



**Figure 1.1:** Segmentation examples

etc. Transitioning from low to high level vision, there is the task of partitioning a scene in meaningfully related regions, usually defined as *image segmentation*, one of the most important and widely studied tasks in computer vision.

In image segmentation, we assume that a scene can be partitioned / segmented in regions such that there is an intrinsic and meaningful relationship among the objects within each region. The task then is to recover these regions. In the context of digital images, this task can be seen a pixel labeling procedure or, in some cases, boundary detection. Figure 1.1 depicts some examples of binary segmentation outputs, where the task is to find exactly two segments in an image. In each one of these examples, the one object present in the image is detected. This exemplifies that the algorithm was able to find a meaningful correspondence between the pixels within the segmented region. The interested reader is invited to the survey of segmentation techniques and principles in [GW<sup>+</sup>18].

### 1.1.1 Applications

Segmentation techniques are widely applied in industry and science as they usually are preprocessing steps to high level vision tasks such as object recognition or detection. Some domains of application are the following:

- *Medical imaging and diagnostics:* As mentioned before, with the surge of CT, MRI and ultrasound imagery there was an increase of interest in analyzing this kind of data. Segmentation plays an important role here. For instance, a radiologist may use segmentation techniques to aid their medical analysis. They can substantially reduce the diagnostic time by making use of algorithms that automatically detect different organs, tissue types or disease visual symptoms.

- *Autonomous vehicles:* Self-driving cars need to perceive their environment in order to drive safely. Therefore, they require automation to detect relevant classes of objects in their surroundings, like other vehicles, buildings, and pedestrians. Semantic segmentation enables self-driving cars to recognize which areas in an image are safe to drive.
- *Robotics and manufacturing:* Similarly to self-driving cars, robots also need to extract information from their surroundings in order to operate. In some tasks, this information is purely visual or its cheaper if it's visual. Examples of that can be found in automatic inspection and production line control.
- *Creativity tools:* It is often necessary for image and video editors to separate objects at pixel-level. This creates opportunities to implement targeted effects on specific areas of a given image, such as blurring its background to sharpen the focus on the foreground or create “stickers” out of specific regions. In the industrial context, image segmentation also enables the development of “try-on” experiences, where users can virtually sample different products (clothes, cosmetics, etc.) before buying them.

### 1.1.2 Challenges

Despite its importance, image segmentation is an ill-posed problem. In general, many different and equally reasonable segmentations can arise from a natural image. This usually leads developers and theoreticians to acknowledge the necessity of some level of human supervision in order to achieve desired segmentation results. This supervision can come in the form of seeds, where some individual pixels are manually assigned to different segments; bounding boxes, where the user provides a rectangular region that encompasses the whole of an object to be segmented; and regions' appearance models, i.e., the expected color distributions of each object in the scene. In recent years, there has also been an increasing interest in learning-based solutions to this intrinsic ambiguity. More and more, practitioners are successfully using deep neural networks architectures trained in very large labeled image datasets to attempt segmentation.

Although the above strategies alleviate the ill-posedness of segmentation tasks, they are

usually costly (in the case of labeled data) or simply non-available. For that reason it is also important to consider the problem of *unsupervised* image segmentation. Here, no user interaction is needed, which then, as we shall see, requires the image model to be more restrictive in order to avoid the ambiguity mentioned above.

Beyond this intrinsic ambiguity in image segmentation, the developer has also to face the effect on image quality on their algorithms. In certain applications, such as in medical and remote sensing imaging, the input data is often corrupted during the acquisition process. Furthermore, in these applications, the segmentation algorithm is expected to find low contrast objects to be segmented in an image. In these situations, methods that rely on edge detection procedures may underperform at detecting crucial objects in a scene.

For that reason, region-based methods have been successfully applied to segmentation. These algorithms often depend either (1) on appearance models that characterize the distribution of pixel values in different image regions or (2) on pair-wise pixel similarity functions that explicitly model the relationships among pixels within each region. As one may imagine, this modeling process, when done unsupervisedly, is at the same time challenging and crucial for the algorithms performance. In (1), for example, researchers have attempted to estimate these models by using parametric estimation or iterative methods. Despite their practical success, these solutions present issues in their generalization, due to their parametric nature, and are usually computationally slow. For (2), classical solutions that consider similarity between neighboring pixels fail to model regions whose pixels are locally dissimilar and globally similar, such as textures. Furthermore, the usage of filtering techniques to alleviate this drawback is notorious for oversmoothing segmentation boundaries.

## 1.2 Organization of this thesis

In this thesis, the focus will be given to graph-based segmentation methods, where the segments of an image are found using techniques from graph analysis. Here, we aim to introduce novel segmentation methods to tackle the aforementioned challenges. These contributions can be grouped in two domains: appearance estimation for Markov Random Fields (MRF) based algorithms and similarity graph construction for spectral segmentation



techniques with global appearance modelling. In both cases, we show that the traditional algorithms associated to either section can be improved in performance and/or interpretation. The proposed algorithms also showed practical convenience, being able to process high-resolution images without loss of performance.

The remainder of this thesis is organized as follows:

**Chapter 2** We review the theory and the state of the art techniques in graph-based image segmentation, particularly in the domains of MRF modeling and spectral clustering. We discuss the drawbacks of each method and provide the commonly proposed solutions to them.

**Chapter 3** We introduce two new non-parametric appearance model estimators that work directly on an image, without any explicit user intervention. These methods rely on second order statistics of fast estimation, from which simple algebraic expressions are derived. The estimation is then accomplished by solving a system of linear and quadratic equations in one method or by computing an eigenvector in the other. Overall, when added as a step in classical graph cut segmentation solvers, the new methods are efficient in challenging segmentation scenarios and faster than most of the considered state-of-the-art unsupervised segmentation techniques. We also introduce an appearance estimation technique based on the method of moments for images with multiple regions.

**Chapter 4** We introduce a new spectral image segmentation method that incorporates long range relationships for global appearance modeling. To implement our image segmentation approach we extend the classical normalized cuts spectral algorithm to a setting where there are multiple graphs that encode different grouping cues. Our approach for image segmentation combines two graphs: a dense graph to capture the global appearance of regions and a grid graph to capture the spatial relationships between pixels. To tackle the computational challenge of dealing with a dense graph, we use a graph sparsification approach that enables the efficient segmentation of high resolution images. In contrast to the common practice, the resulting method can segment challenging images without any filtering or pre-processing.

**Chapter 5** We show how to incorporate seed, appearance models and two segmentation cues in traditional spectral segmentation algorithms in a novel manner. The proposed framework, named Penalized Normalized Cuts, is constructed such that a penalty function is added to traditional normalized cut criterion for clustering. Through simple choices of this plug-and-play penalties, different user/expert or prior knowledge about the final segmentation can be added to the spectral clustering pipeline in an interpretable way. On the implementation side, we also show that our formulation leads a scalable sparse eigenvalue problem that can be efficiently solved via typical power iteration methods. Finally, we present preliminary results of our methods in synthetic and real experiments.

**Chapter 6** We conclude this thesis by summarizing the contributions of each proposed method and give an overview of possible future directions of research.

# CHAPTER 2

## Graph-based Image Segmentation and Spectral Clustering

In this chapter we review the theoretical background on graphical methods for image segmentation. We define our image model and give a brief summary of Markov Random Fields (MRFs), the associated optimization techniques and Normalized Cuts. We also present some segmentation algorithms that will be used in the experiments sections of later chapters for performance comparison.

### 2.1 Graph-based Methods for Image Segmentation

Let  $I : \Omega \rightarrow L$  be an image on  $n$  pixels, where  $\Omega$  is a set of pixel locations and  $L$  is a finite set of pixel values. For example, for an  $h$  by  $l$  graylevel image we have  $\Omega = \{1, \dots, h\} \times \{1, \dots, l\}$  while  $L = \{0, \dots, 255\}$ . We use  $I(i)$  to denote the value of a pixel  $i \in \Omega$ . In graph-based image segmentation,  $I$  is represented as a weighted undirected graph  $G = (V, E)$ . The nodes in  $V$  represent the image pixels and the edge set  $E$  consists of edges representing relationships between pixels, whose weight  $w(i, j)$  encapsulates the similarity between pixels  $i$  and  $j$ . The task is to partition  $G$  into  $K$  disjoint sets such that the similarity among the nodes within a set is high, while the similarity of nodes in different sets is low.

Many graph approaches using the above representation have been developed for image segmentation. The reasons behind its popularity are (1) the development of efficient algorithms for computing approximate or exact solution to the the Markov Random Field (MRF) inference problem and (2) the theoretical and practical advances in tackling graph clustering problem, specially with the emergence of spectral clustering techniques.

### 2.1.1 Bayesian Formulation

In order to model the image segmentation problem, we start by defining the vector  $\mathbf{x} \in \{1, \dots, K\}^n$  as a labeling vector that assigns each pixel in  $I$  to a segment. We model it as the realization of a discrete random variable  $\mathbf{X}$  of same dimension and assume it to be distributed according to a posterior probability distribution  $P(\mathbf{X} = \mathbf{x}|I)$  (shortened to  $P(\mathbf{x}|I)$ ). Based on that our goal is to find the appropriate  $\mathbf{x}$  for our problem.

Bayesian statistics is a fundamental theory in estimation and decision-making. According to the Bayes rule, the posterior probability can be computed by using the formula:

$$P(\mathbf{x}|I) = \frac{P(I|\mathbf{x})P(\mathbf{x})}{P(I)}, \quad (2.1)$$

where  $P(\mathbf{x})$  is the prior distribution on labelings  $\mathbf{x}$ , that gathers all the assumptions about how good or realistic labelings should behave;  $P(I|\mathbf{x})$  is the likelihood function, which models the probability that  $I$  was generated by a segmentation represented by  $\mathbf{x}$ , and  $P(I)$  is a distribution on  $I$  which is a constant when  $I$  is given.

In Bayesian decision-making, when both the prior distribution and the likelihood function of a segmentation are known, the best that can be estimated from these sources of knowledge is the Bayes labeling, formulated via a risk function:

$$\text{Risk}(\mathbf{x}) = \mathbb{E}_{\mathbf{x}'|I}[\text{Cost}(\mathbf{x}, \mathbf{x}')] = \sum_{\mathbf{x}' \in \{1, \dots, K\}^n} \text{Cost}(\mathbf{x}, \mathbf{x}')P(\mathbf{X} = \mathbf{x}'|I), \quad (2.2)$$

where, the cost function  $\text{Cost}(\mathbf{x}, \mathbf{x}')$  (which is also called "loss" in other contexts) computes the penalty of estimate  $\mathbf{x}$  when the truth is  $\mathbf{x}'$  and can be chosen based on one's preference. In Bayesian statistics, that risk is minimized in order to obtain the optimal (Bayes) estimate.

In this thesis, we focus our attention to the 0–1 Cost function for its computational benefits<sup>1</sup>:

$$\text{Cost}(\mathbf{x}, \mathbf{x}') = \begin{cases} 0, & \|\mathbf{x} - \mathbf{x}'\| < \delta \\ 1, & \text{otherwise} \end{cases} \quad (2.3)$$

where  $\delta$  is a small constant. As  $\delta \rightarrow 0$ , the risk in Eq. 2.2 can be approximated by

$$\text{Risk}(\mathbf{x}) \approx 1 - cP(\mathbf{x}|I) \quad (2.4)$$

where  $c$  is a constant. The Bayes estimate of  $\mathbf{x}$  is given by the minimizer of  $\text{Risk}(\mathbf{x})$ . Using Eq. 2.1, it can then be computed as:

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x} \in \{1, \dots, K\}^n} \text{Risk}(\mathbf{x}) = \arg \max_{\mathbf{x} \in \{1, \dots, K\}^n} P(\mathbf{x}|I) = \arg \max_{\mathbf{x} \in \{1, \dots, K\}^n} P(I|\mathbf{x})P(\mathbf{x}), \quad (2.5)$$

where we used the fact that  $P(I)$  is constant for a fixed image. The above estimator is called *Maximum a posteriori* (MAP) estimator and it is one of the most popular statistical criteria for optimality in bayesian vision modeling [GG84, Li09].

### 2.1.2 Markov Random Fields and the Gibbs Formalism

Also typical to vision problems is to assign  $P(\mathbf{x}|I)$  to be the posterior distribution of a Markov Random Field. Let  $G = (V, E)$  be an undirected graph whose nodes correspond to the random variables  $\mathbf{X} = \{X_i\}_{i \in V}$ . Let  $\mathcal{N}_i$  be the set of neighboring nodes of  $i$  on  $G$ .  $\mathbf{X}$  is said to be a Markov Random Field (MRF) on  $V$  with respect to  $G$  if the following conditions are satisfied:

**Positivity:**  $P(\mathbf{X} = \mathbf{x}) > 0 \quad \forall \mathbf{x} \in \{1, \dots, K\}^n$ .

**Markovianity:**  $P(X_i|X_{V \setminus i}) = P(X_i|X_{\mathcal{N}_i}), \quad \forall i \in V$ .

While the the positivity condition is only assumed for technical reasons, the Markovianity property above characterizes the local characteristics of  $\mathbf{X}$ . In other words, it states that only

---

<sup>1</sup>Another important cost function is the quadratic cost (or loss), whose risk minimization leads to the computation of the *posterior mean*. Unfortunately, such statistic is computationally intractable for reasonably sized problems, such as the ones in image segmentation.

neighboring labels have direct interactions with each other. This is an important attribute for modeling, since it simplifies computation and represents a reasonable approximation of complex pixel-interactions in real images.

An important step in Bayes labeling of MRF's is to derive its joint distribution. According to the Hammersley-Clifford theorem [HC71], it is possible to show the equivalence between a Markov Random Field and a Gibbs Random Field (GRF).  $\mathbf{X}$  is a GRF on  $V$  with respect to  $G$  if and only if its configurations obey a Gibbs distribution, i.e.,

$$P(\mathbf{X} = \mathbf{x}) = \frac{1}{Z} \exp\left(-\frac{1}{T}\mathcal{E}(\mathbf{x})\right), \quad (2.6)$$

where  $Z = \sum_{\mathbf{x}} \exp\left(-\frac{1}{T}\mathcal{E}(\mathbf{x})\right)$  is a normalizing constant,  $T$  is a constant called temperature (from now on assumed to be 1) and  $\mathcal{E}(\mathbf{x})$  is the energy of a configuration  $\mathbf{x}$ . When  $\mathbf{X}$  is a GRF, the energy  $\mathcal{E}(\mathbf{x})$  can be written as

$$\mathcal{E}(\mathbf{x}) = \sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c), \quad (2.7)$$

where  $\mathcal{C}$  is the set of maximal cliques<sup>2</sup> in  $G$ ,  $\mathbf{x}_c$  are the variables in clique  $c$ ,  $\psi_c(\cdot)$  are positive potential functions, defined according to the problem at hand. This formulation has a great practical value, since it provides a simple way of specifying the joint probability of  $\mathbf{X}$  by choosing appropriate clique potentials  $\psi_c(\mathbf{x}_c)$  according to the desired system behavior.

In most imaging problems,  $G$  only contains pairwise interactions between its nodes, i.e, each clique includes no more than two variables. This leads us to reformulate the configuration energy as:

$$\mathcal{E}(\mathbf{x}) = \sum_{i \in V} \sum_{j \in \mathcal{N}_i} \psi_{i,j}(x_i, x_j). \quad (2.8)$$

In modeling the likelihood distribution  $P(I|\mathbf{x})$ , we also assume it is Gibbs distributed with energy function given by:

$$\mathcal{E}(\mathbf{x}, I) = \sum_{i \in V} \phi_i(x_i, I) \quad (2.9)$$

---

<sup>2</sup>A maximal clique a clique that cannot be extended by including one more adjacent vertex, meaning it is not a subset of a larger clique.

for certain unary potential functions  $\phi_i$ . Now having expressions for both the prior  $P(\mathbf{x})$  and the likelihood  $P(I|\mathbf{x})$ , and defining the energy:

$$\mathcal{E}(\mathbf{x}|I) = \mathcal{E}(\mathbf{x}, I) + \mathcal{E}(\mathbf{x}) = \sum_{i \in V} \phi_i(x_i, I) + \sum_{i \in V} \sum_{j \in \mathcal{N}_i} \psi_{i,j}(x_i, x_j), \quad (2.10)$$

the MAP problem in Eq. 2.5 can be restated as:

$$\hat{\mathbf{x}} = \arg \max_{\mathbf{x} \in \{1, \dots, K\}^n} P(\mathbf{x}|I) = \arg \min_{\mathbf{x} \in \{1, \dots, K\}^n} \mathcal{E}(\mathbf{x}|I) \quad (2.11)$$

In the next section, we will see how the above optimization translates to a segmentation problem and which strategies can be used to (approximately) solve it.

### 2.1.3 Graph Cuts

In binary segmentation problems, where the goal is to partition an image in two regions, the random variables in the MRF framework will be binary and will correspond to regions in the image. A region  $\mathcal{R}$  is a subset of the pixels in  $\Omega$ . Assume the image  $I$  can be divided into two regions  $\mathcal{R}_0$  and  $\mathcal{R}_1$  and let  $\mathbf{x} \in \{0, 1\}^n$  denote now a binary assignment of the pixels in  $\Omega$ .

In this framework, the unary potentials  $\phi_i(x_i, I)$  assume the meaning of how well the pixel  $i$  has its appearance (color information, for example) represented by the model of label  $x_i$ . In many MRF based segmentation solvers [RKB04, TBAMB15], the *appearance model*  $\theta \in \mathbb{R}^L$  described by a label is a discrete distribution that specifies the typical values for the pixels in the image region defined by that label. In other words, it is the normalized histogram of the pixel values (colors) within a region of  $I$ . Let  $\theta_0$  and  $\theta_1$  be the appearance models of regions  $\mathcal{R}_0$  and  $\mathcal{R}_1$ . A typical choice for  $\phi_i$  is  $\phi_i(x_i, I) = -\ln \theta_{x_i}(I(i))$ , i.e., the negative log-probability of pixel value  $I(i)$  being found in region  $x_i$ . In other words, minimizing each  $\phi_i(x_i, I)$  for all  $i$  encodes the intuition that pixels should be assigned to regions where they are most likely to be found in according to their appearance models.

The binary potentials  $\psi_{i,j}(x_i, x_j)$  here give a notion of spacial consistency to the final labeling, i.e., when minimized, it encourages neighboring pixels to have the same labels.

In other words, it gives preference to short region boundaries, a common assumption for realistic segmentations. Therefore, one possible choice for these potentials can be  $\psi_{i,j}(x_i, x_j) = |x_i - x_j|$  for  $i$  and  $j$  neighbours in  $G$ , which is called the Ising model within the statistical mechanics community [Isi25, P+98, Ben10]. Another common choice for them is  $\psi_{i,j}(x_i, x_j) = \exp(-\|I(i) - I(j)\|_2^2/\sigma^2)$ , where  $\|\cdot\|_2$  is the Euclidean norm and  $\sigma$  is a constant [BFL06]. In this thesis, we will use the Ising model for simplicity.

Plugging the definitions of  $\phi_i$  and  $\psi_{i,j}$  back into Eq. 2.10 and adding a multiplicative constant  $\lambda > 0$  to  $\psi_{i,j}$  as a parameter to control the prior model, we have that the MRF energy for a binary segmentation task becomes:

$$\mathcal{E}(\mathbf{x}|\lambda, \theta_0, \theta_1) = - \sum_{i \in \Omega} \ln \theta_{x_i}(I(i)) + \lambda \sum_{i \in \Omega} \sum_{j \in \mathcal{N}_i} |x_i - x_j|. \quad (2.12)$$

If one has the appearance models  $\theta_0$  and  $\theta_1$ , the above energy can be efficiently minimized by what is called the Graph Cuts algorithm. Initially studied by [GPS89], it was then further developed and popularized by the work in Eq. [BVZ99]. To segment a given image, an undirected graph  $G' = (V', E')$  is created from  $G$  by adding to it two node terminals,  $s$  and  $t$ , and the edges connecting each pixel to them:

$$V' = V \cup \{s, t\}, \quad E' = E \bigcup_{i \in V} \{i, s\} \bigcup_{j \in V} \{j, t\} \quad (2.13)$$

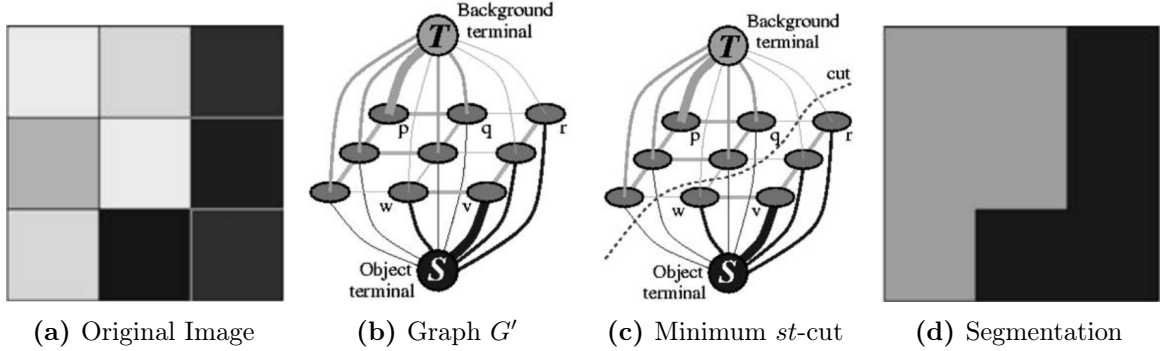
This construction leads us to two different types of edges and therefore weights, in  $G'$

**Terminal edges,  $t$ -links:** These links encode the appearance terms of the cost function, represented in our unary potentials. For our appearance based segmentation purpose, the weight of the edge  $\{i, s\}$  connecting pixel  $i$  and  $s$  is given by  $w(i, s) = -\ln \theta_0(I(i))$ , while  $w(i, t) = -\ln \theta_1(I(i))$  for the edge  $\{i, t\}$  [RKB04, TBAMB15].

**Neighboring edges,  $n$ -links:** These edges capture the region consistency/boundary terms of the cost function. Since we are using Ising model potentials,  $w(i, j)$  will be constant and equal to  $\lambda$ .

An example of this construction is depicted in in Figures 2.1a and 2.1b, where the weight of





**Figure 2.1:** Graph Cut algorithm. Adapted from [BFL06]

each edge is indicated in its thickness. In this example, the user has access to the appearance models of both back and foreground. Then, the segmentation  $\mathbf{x}$  that globally minimizes 2.12 can be computed by finding the minimum  $st$ -cut of  $G'$ <sup>3</sup>. That can be done in polynomial time using a “max-flow” algorithm [BJ01]. Figure 2.1c shows the minimum cost cut of the graph in 2.1b and Figure 2.1d provides the final segmentation result.

So far, we dealt with binary segmentation problems, but this framework can be extended to multiregion segmentation. Consider that we now have  $K$  regions in  $I$ , each with their respective appearance models  $\theta_1, \dots, \theta_K$ , and redefine  $\mathbf{x} \in \{1, \dots, K\}^n$ . Then, the multiregion segmentation problem can be modelled as a minimization of the following energy:

$$\mathcal{E}(\mathbf{x}|\lambda, \theta_0, \dots, \theta_K) = - \sum_{i \in \Omega} \ln \theta_{x_i}(I(i)) + \lambda \sum_{i \in \Omega} \sum_{j \in \mathcal{N}_i} \mathbb{1}(x_i \neq x_j). \quad (2.14)$$

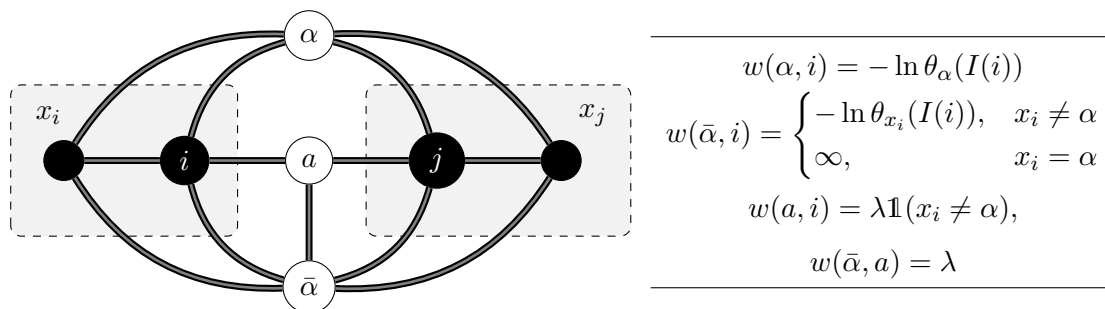
where  $\mathbb{1}(\cdot) \in \{0, 1\}$  is an indicator function. The smoothing term used above is called Potts model. While minimizing Eq. 2.12 can be exactly solved in polynomial time via a min-cut/max-flow algorithm, one can show that computing the global minimum of Eq. 2.14 is, in general, NP-hard [BVZ99]. Indeed, the main contribution of the work in [BVZ99] was to propose *approximate* iterative solvers for the minimization of Eq. 2.14<sup>4</sup>, called  $\alpha$ -expansion

<sup>3</sup>Let  $G$  be a graph containing nodes marked as  $s$  and  $t$ . The minimum  $st$ -cut of  $G$  is a partition of its nodes into sets  $S$  and  $T$  with  $s \in S$  and  $t \in T$  that minimizes  $\sum_{i \in S} \sum_{j \in T} w(i, j)$ , i.e., the weight sum of the edges going across the partition.

<sup>4</sup>While we are focused on the Potts model, the authors in [BVZ99] show that their algorithms only require the smoothing term to be either a metric (in the case of  $\alpha$ -expansion) or a semi-metric (in the case of  $\alpha\beta$ -swap).

and  $\alpha\beta$ -swaps algorithms. Both methods are based on successive improvements of a current labeling. In the following, we present a quick overview of these methods, but more details on these constructions and the respective proofs can be found in [BVZ99].

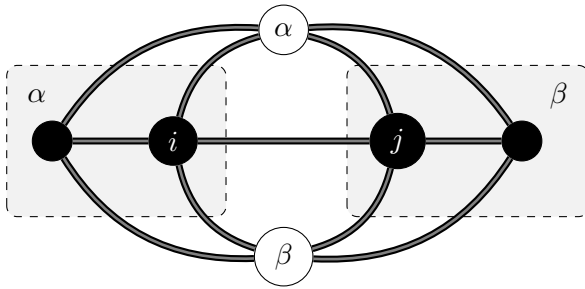
**$\alpha$ -expansion:** The main idea of the  $\alpha$ -expansion algorithm is to successively segment all pixels labeled  $\alpha \in \{1, \dots, K\}$  from those labeled as non- $\alpha$  (here represented by  $\bar{\alpha}$ ) using the standard min-cut/max-flow algorithm. At the end of each segmentation, the pixels on the same segment as the terminal node representing the  $\alpha$  label will be marked as  $\alpha$  and the remaining nodes will keep their prior labels. As shown in [BVZ99], the region in  $I$  labeled as  $\alpha$  will necessarily expand or keep its prior size after each iteration, hence the algorithm's name.



**Figure 2.2:** The graph used in the  $\alpha$ -expansion algorithm for a 1D image. The nodes representing the image pixels are in black and their current graph partitions are shown in gray. The terminal nodes are depicted as  $\alpha$  and  $\bar{\alpha}$ . The node  $a$  is added in between every pair of nodes whose region assignments are different.

At each iteration  $t$  a new graph  $G^{(t)}$  is constructed for the current  $\alpha$  being expanded according to a current pixel assignment  $\mathbf{x}^{(t)}$ . Figure 2.2 gives an example of such graph construction for a 1D image. The algorithm will iterate through each possible label for  $\alpha$  until it converges. The authors in [BVZ99] also provided an approximation factor for this algorithm when minimizing the energy in 2.14.

**$\alpha\beta$ -swaps:** In this algorithm, we are interested in improving the energy in Eq. 2.14 for the set of pixels assigned to a pair of labels  $\alpha$  and  $\beta$  on a current assignment  $\mathbf{x}^{(t)}$ . The  $\alpha\beta$ -swap algorithm successively segments all  $\alpha$ -labeled pixels from  $\beta$ -labeled pixels using the binary graph cut method and relabels as them  $\alpha$  or  $\beta$  according to which segment they end up belonging to.



---

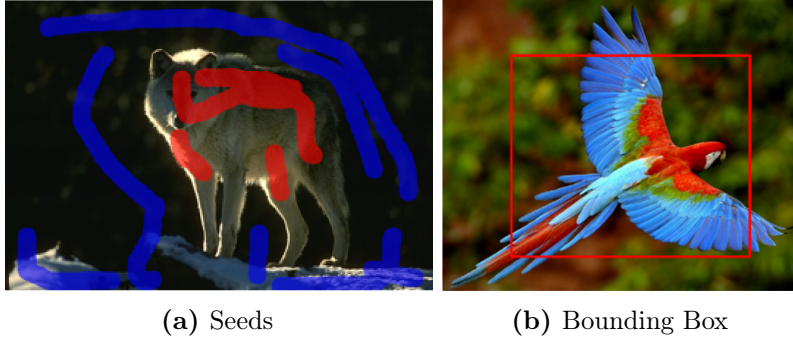

$$w(\alpha, i) = -\ln\theta_\alpha(I(i)) + \lambda \sum_{k \in \mathcal{N}_i} \mathbb{1}(x_k \notin \{\alpha, \beta\})$$

$$w(\beta, i) = -\ln\theta_\alpha(I(i)) + \lambda \sum_{k \in \mathcal{N}_i} \mathbb{1}(x_k \notin \{\alpha, \beta\})$$

$$w(i, j) = \lambda$$


---

**Figure 2.3:** The graph used in the  $\alpha\beta$ -swap algorithm for a 1D image. The nodes representing the image pixels are in black and their current graph partitions are shown in gray. The terminal nodes are depicted as  $\alpha$  and  $\beta$ .



**Figure 2.4:** Typical user interactions. Adapted from [TBAMB15]

At each iteration  $t$  a new graph  $G^{(t)}$  is constructed according to  $\mathbf{x}^{(t)}$ . Figure 2.3 gives an example of such graph construction for a 1D image. The algorithm will change the  $\alpha - \beta$  combination at each iteration and will iterate through all possible combinations until it converges.

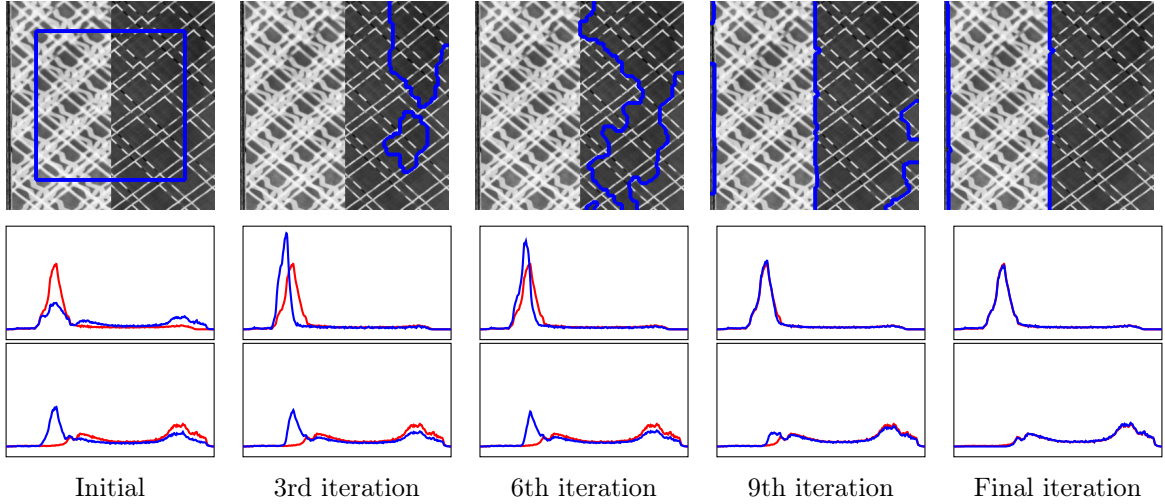
## 2.2 Appearance Modeling

In most real world imaging problems, the appearance models are not available or are expensive to obtain. In fact, as shown above, the segmentation problem would be easily approachable if these models were given. Therefore, the techniques shown in the previous section require the estimation of these models in order to be applicable in practice.

### 2.2.1 User interaction

A solution for the model estimation issue is to allow for some user interaction. Typically, the interactive data comes in the form of seeds, where the user marks some pixels to belong to the background and to the foreground separately (Figure 2.4a), or as a region, usually a rectangular bounding box, coarsely detecting the location of the object in the image (Figure 2.4b). These solution has been vastly employed in many graph-based image segmentation approaches.

The classical approach to use seeds within the graph cut methodology was introduced by [BJ01]. The main idea is to frame the hard constraints imposed by the seeds as weights on the  $t$ -links of the graph construction depicted in Figure 2.1. In [BJ01], the authors set the weights of the edges from "seeded" pixels to their terminals (say, the foreground seeds to terminal corresponding to the foreground region) to be constant and kept all the other  $t$ -link weights as zero. The authors in [Gra06] used the seeds data to define the exit nodes of a random walk on the grid graph defined by the image, where the edge weights corresponds to the likelihood that walker crosses that edge. Starting from each unlabeled pixel, the walker computes the probability that it first reaches one of the seeds. The pixel is then labeled as foreground of background according to that seed. The work in [DMFU10], on the other hand, proposes a framework where the edge weights are learned from the the information provided by the seeds. This new graph is then partitioned using the seed-based graph cut algorithm from [BJ01]. Within the deep learning based approaches, the authors of [XPC<sup>+</sup>16] transform the foreground and background seeds into two Euclidean distance maps that are then concatenated with the RGB channels of the original image to form a image-seed pair. Many of these pairs are then generated from a segmentation dataset, from which a Fully Convolutional Network is trained to estimate segmentation probability maps. The same authors then extended their method to, potentially misaligned, bounding boxes in [XPC<sup>+</sup>17]. In a related approach, [MCPTVG18] proposes adding extreme points in an object (left-most, right-most, top, bottom pixels) to the CNN pipeline in order to compute more precise segmentations with less user input. A final example of the use of bounding boxes in image segmentation is the Grabcut algorithm [RKB04], which will be discussed in the following section.



**Figure 2.5:** Model estimation and segmentation using ALT. On the bottom, we see how both foreground and background color distributions estimated by ALT (—) evolve compared to the ground truth appearance models (—). The evolution of the segmentations given the models is shown on top.

### 2.2.2 Iterative Methods

From Section 2.1, one can arrive at the following observations:

- From a segmentation  $S$  of an image in  $K$  regions, it is possible to compute the appearance models  $\theta_1, \dots, \theta_K$  by computing color histograms<sup>5</sup>.
- From the appearance models  $\theta_1, \dots, \theta_K$ , one can find a segmentation  $S$ . Using our graph image model, this can be done by minimizing Eq. 2.12 when  $K = 2$  using graph cuts or approximately minimizing Eq. 2.14 for  $K > 2$  using the  $\alpha$ -expansion or  $\alpha\beta$ -swap algorithms.

These observations turn the image segmentation problem in a “chicken-and-egg” problem and provide the basis for an simple iterative solution to model-based image segmentation: (1) start with an initial segmentation  $S$ ; (2) compute the normalized color histograms of each region of  $I$  according to  $S$ ; (3) for a constant  $\lambda$ , compute a new segmentation  $S'$  using the current appearance models; (4) iterate until convergence. To compute a new segmentation

<sup>5</sup>In RGB images, this computation may be impractical, given the color space size. Further into the text, we will show how to avoid this issue.

using the current appearance models we minimize the energy in Equation (2.12) using the usual max-flow based algorithms for graph cuts [BVZ99]. It is easy to verify that this simple iterative method converges. Furthermore, the authors in [TAB14] showed that these iterations gives rise to a majorization-minimization method for an entropy-based energy. This local solution then provides a segmentation that is coherent and whose appearance models have low entropy. In practice, when computing appearance models we “smooth” the histograms of each region by adding a constant  $K = 1$  to their bins before normalizing them. In this thesis, this method will called ALT and an example of its iterations is showed in Figure 2.5 for an image composed of two Brodatz textures [Bro66].

The above method becomes impractical when dealing with RGB images, since the histogram computation step becomes prohibitive in that space due the the amount of different colors a typical colored image has. The method proposed in [RKB04], called Grabcut, solves this issue by estimating parametric appearances (Gaussian mixture models, GMM, with 5 components) on the iteration’s estimation step. They also added a final step for further user editing via extra seeds addition. Grabcut’s efficient results in practical settings made it the standard segmentation algorithm in industrial applications, with several open source implementations. Figure 2.6 depicts some of its results on challenging real world images.

Some improvements were accomplished over the original Grabcut algorithms. [CYW<sup>+</sup>08] integrates a local color pattern model and edge model in the graph-cut framework in order to improve robustness and enhance the discriminability of the method. In [HYL08], a constrained Delaunay triangulation is used in order to improve the initial foreground estimation from the bounding box data. The authors successfully applied this technique to clothing segmentation. The work in [TBAMB15] outlines the correspondence between the energy in Eq. 2.12 and a probabilistic version of the  $K$ -means objective function. From there, they show that the optimization promoted by the Grabcut algorithm can be improved by replacing its GMM fitting with a kernel  $K$ -means feature clustering. Furthermore, their new optimization technique is shown to be closely related to normalized cuts, which will be discussed in further details later. Finally, it is worth noting that the Grabcut method and its variants have been extensively applied to compute or improve segmentation results when some data about the foreground and/or background is loosely know. For instance, the work



**Figure 2.6:** Grabcut segmentation results. Adapted from [RKB04]

in [CMH<sup>+</sup>14] uses the information coming from its object saliency estimator in order to give an initial loose segmentation mask to Grabcut.

### 2.2.3 Variational Approaches

Departing from the graph based segmentation modeling, it is worth discussing about variational approaches to segmentation to the extent that they also realize appearance model estimation. Within the vast variational image segmentation literature, geometric deformable models [Set99, CCCD93, MSV95] have been particularly popular within the computer vision community. This is mainly because of their ability to handle topological changes of the unknown object to be segmented and their mathematically well established curve evolution theory. The so-called level set methods sees segmentation boundaries implicitly defined as zero level set functions. These functions are then optimized according to the segmentation problem at hand.

Let  $f(p, t)$  be a moving curve, where  $p$  denotes the parameter for the curve and  $t$  denotes time. We want to evolve the  $f$  so that it forms a segmentation boundary. A reasonable formulation to this propagation is

$$\frac{\partial f(p, t)}{\partial t} = F(x, y)\mathbf{n}, \quad (2.15)$$

where the curve moves along its normal direction  $\mathbf{n}$  at a velocity  $F(x, y)$ , where  $x$  and  $y$  are the coordinates on the 2D plane. The curve propagation stops where  $F(x, y) = 0$ . If  $F(x, y)$  is set to be the reciprocal of the image gradient, for example, the propagation stops where the gradient is large, very likely being the boundary of an object. Alternatively, we can also

define an integral energy function  $E(f)$  for the curve  $f(p, t)$ , such that:

$$E(f) = \int_{\Omega} e \left( x, y, f, \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right) dx dy, \quad (2.16)$$

where  $e$  is a local energy function defined at each coordinate in the space  $\Omega$ . In that case, solving the Euler-Lagrange equation, one can also show that:

$$\frac{\partial f(p, t)}{\partial t} = - \frac{E(f)}{\partial f} \quad (2.17)$$

The propagation in Eq. 2.15, however, does not allow for topological changes in the curve during its propagation. To overcome this problem, the level set method was proposed [Set99]. Let  $\psi(x, y, t)$  be a higher dimensional function whose zero level set now *implicitly* defines the curve  $f$ . Taking  $\psi$  to be a 3D surface, then we have that:

$$f(t) = \{(x, y) | \psi(x, y, t) = 0\}. \quad (2.18)$$

One can then show that the curve evolution on Eq. 2.15 is associated with the following surface evolution:

$$\frac{\partial \psi(x, y, t)}{\partial t} = F(x, y) \|\nabla \psi(x, y, t)\|, \quad (2.19)$$

where  $\nabla(\cdot)$  is the gradient of a function w.r.t. the spacial coordinate point  $(x, y)$ .

In level set methods, the definition of the energy  $E(f)$  (and therefore the associated velocity  $F(x, y)$ ) is crucial. In the famous Chan-Vese Active contour formulation [CV01], the following energy, derived from the Mumford-Shah functional [MS89b], is used:

$$E(f) = \int_{\mathcal{R}_0} (I(x, y) - \mu_0)^2 dx dy + \int_{\mathcal{R}_1} (I(x, y) - \mu_1)^2 dx dy + \lambda |\partial S|, \quad (2.20)$$

where  $\mathcal{R}_0$  and  $\mathcal{R}_1$  are the image regions defined limited by  $f$ ,  $\mu_0$  and  $\mu_1$  are their respective mean intensities,  $|\partial S|$  is the boundary length of the segmentation defined defined by  $f$  and  $\lambda$  is a balancing term. In [CV01], the above energy is extended to an energy on the level set



function  $\psi$ :

$$E(\psi) = \int H_\epsilon(\psi(x, y, t))(I(x, y) - \mu_0)^2 dx dy + \int (1 - H_\epsilon(\psi(x, y, t)))(I(x, y) - \mu_1)^2 dx dy + \lambda \int \delta_\epsilon(\psi(x, y, t)) \|\nabla \psi(x, y, t)\| dx dy, \quad (2.21)$$

where the integral is evaluated on the whole image space,  $H_\epsilon(\cdot)$  is the smoothed Heaviside function,

$$H_\epsilon(z) = \begin{cases} 0, & \text{if } z < -\epsilon \\ 1, & \text{if } z > \epsilon \\ \frac{1}{2} \left(1 + \frac{z}{\epsilon} + \frac{1}{\pi} \sin\left(\frac{\pi z}{\epsilon}\right)\right), & \text{otherwise} \end{cases}, \quad (2.22)$$

and  $\delta_\epsilon$  is its derivative. Figure 2.7 shows the resulting smoothed Heaviside function for some values of  $\epsilon$ . The above energy is minimized alternating the computation of  $\mu_0$  and  $\mu_1$  (with  $\psi$  fixed) as

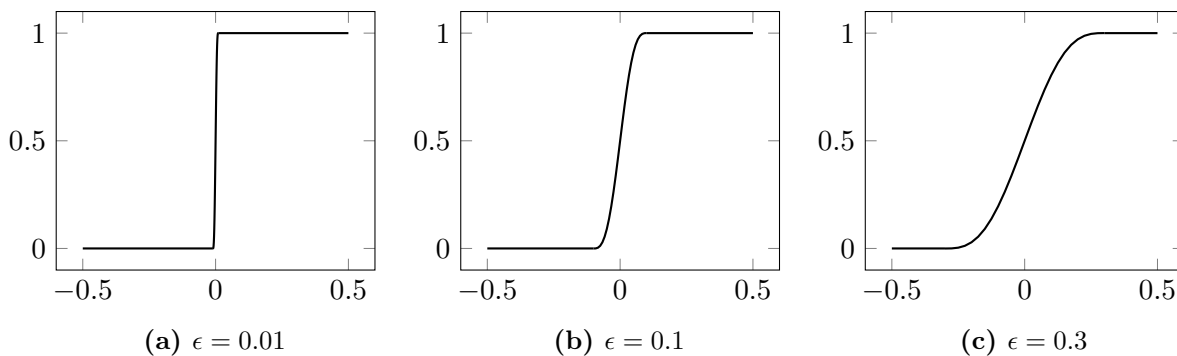
$$\mu_0^{(t)} = \frac{\int I(x, y) H_\epsilon(\psi(x, y, t)) dx dy}{\int H_\epsilon(\psi(x, y, t)) dx dy}, \quad \mu_1^{(t)} = \frac{\int I(x, y) (1 - H_\epsilon(\psi(x, y, t))) dx dy}{\int (1 - H_\epsilon(\psi(x, y, t))) dx dy} \quad (2.23)$$

and  $\psi$  (with  $\mu_0^{(t)}$  and  $\mu_1^{(t)}$  fixed) such that it satisfies the Euler-Lagrange equation:

$$\delta_\epsilon(\psi) \left[ \lambda \operatorname{div} \left( \frac{\nabla \psi}{\|\nabla \psi\|} \right) + \left( I - \mu_0^{(t)} \right)^2 - \left( I - \mu_1^{(t)} \right)^2 \right] = 0. \quad (2.24)$$

When this process converges, the final segmentation is given by the zero level set of  $\psi$ .

In the above, it is possible to notice some similarities between the alternations in Grabcut and in Chan-Vese algorithms: both iterate between computing a statistic that pertains to each region in the current image partition and updating the segmentation given that newly computed statistic. While the statistic in Chan-Vese is simply the mean color intensity, the authors in [NBCE09] propose a level set formulation that computes cumulative appearance models of each region at each step and uses them to update  $\psi$ . In their work, they also show that this new iterative algorithm, under certain assumptions, searches for regions whose local appearance models (i.e., within a certain radius around each pixel) are similar according



**Figure 2.7:** Smoothed Heaviside function for various values of  $\epsilon$ .

to a Wasserstein distance. In this thesis, this method will be called LSWD, for Level Set segmentation based on Wasserstein Distances.

#### 2.2.4 Implicit Modeling

Although the appearance information is necessary in model based segmentation algorithms, it does not need to be explicitly computed. In other words, it is possible to suggest new appearance-based segmentation energies, whose minimization algorithm does not require appearance computation. In the algorithm called "Grabcut in One Cut" [TGVB13], for instance, the authors propose the minimization of the following energy:

$$\mathcal{E}(S, h_0, h_1) = -\|h_0 - h_1\|_1 + \lambda \sum_{i \in \Omega} \sum_{j \in \mathcal{N}_i} |x_i - x_j|, \quad (2.25)$$

where  $\|\cdot\|_1$  is the  $L_1$  norm and  $h_0$  and  $h_1$  are the unnormalized appearance models (histograms) of each region defined by the segmentation  $S$ . Here, note that we are interested in the optimization of both  $S$  and the appearances at the same time, such that the ideal segmentation will put  $h_0$  and  $h_1$  as far from each other as possible according to the  $L_1$  metric. As the algorithm's name suggests, this minimization can be accomplished by computing one graph cut, and the method's ingenuity lies in the construction of such a graph. The idea is to add  $L$  auxiliary nodes to usual image grid graph, one node for each color in  $I$ . Then, each pixel node is connected to the node that corresponds to its color. In [TGVB13], the authors showed that the value of a cut on this simple graph has the same effect as computing  $-\|h_0 - h_1\|_1$

up to an additive constant. In their paper, they also explain how to use seed or bounding box data in their proposed framework.

### 2.2.5 Factorization-Based Methods

In [YWC15], Yuan *et al.* developed a method to achieve image segmentation by making use of local pixel information and computing what could be thought as a spectral appearance model. By spectral, the authors mean the filter responses computed through the convolution of an image window  $W$  with a chosen bank of  $K$  filters. Then, for the same window  $W$ , they define a local spectral histogram with respect to a filter bank as:

$$h_W = \frac{1}{|W|} [h_W(1), \dots, h_W(K)], \quad (2.26)$$

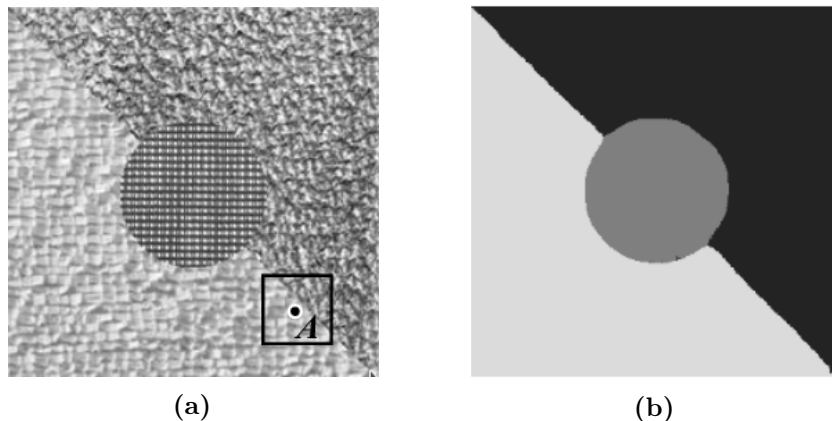
where  $|\cdot|$  denotes cardinality and  $h_W(i)$  is the response of the  $i$ -th filter on  $W$ . In Figure 2.8a, a rectangular region of the image is depicted around a pixel labeled  $A$ . For that region, the authors noticed that:

$$h_W \approx w_1 h_1 + w_2 h_2 + w_3 h_3 = [h_1, h_2, h_3]^\top [w_1, w_2, w_3], \quad (2.27)$$

where  $h_1$ ,  $h_2$  and  $h_3$  are the spectral histograms of the textures on the upper right, center and lower left corners, respectively, and  $w_1 + w_2 + w_3 = 1$  (from the location of  $A$  and its surrounding region,  $w_2$  should be set to zero). In other words, the spectral histogram around  $A$  is a linear combination of the spectral histograms of the constituent textures in  $I$ . That led the authors to consider the histograms of all possible windows in the image (which can be efficiently computed using the integral histograms technique [Por05]). Let  $X$  be the matrix of these histograms, stacked, let  $Z$  be the matrix of representative spectral histograms (one for each region) and let  $\beta$  the matrix of weights. Then, it follows that:

$$X \approx Z\beta. \quad (2.28)$$

One can manually put a seed in each homogeneous region, i.e. fully inside a sole texture, to compute representative features and proceed with least squares to estimate the weights of



**Figure 2.8:** Linear combination of spectral histograms. Adapted from [YWC15].

each region/pixel:

$$\beta \approx (Z^T Z)^{-1} Z^T X. \quad (2.29)$$

Finally, the newly computed weights can determine the segment label of each pixel, by assigning it to the region that has the largest weight. Figure 2.8b show the result of the application of this algorithm to the image in Figure 2.8a. In [YWC15], it is also proposed a method to compute the representative features in an unsupervised manner by extracting the singular vectors of  $X$ . Then, nonnegative matrix factorization via an efficient alternate least squares method is used to ensure the nonnegativity constraints. From this technique, it is also possible to estimate the number of segments in the image. In this thesis, this method will be called FBS, for Factorization-Based Segementation.

Drawing from a similar intuition, [BMB16] proposed an algorithm that first oversegments the image into regions through a watershed transform, extracts features from each region (it could be a color histogram or the average color within), stacks them into a feature matrix and uses nonnegative matrix factorization to estimate the regions' weights. The authors then used the information from the graph generated by the watershed transform to regularize the computed weights. Here, this algorithm will be called PNMF, for Projective Non-Negative Matrix Factorization on a Graph.

Finally, the work in [MMF<sup>+</sup>14] considers an image as a collection of superimposed textures and develops an algorithm, named ORTSEG, based on local color histograms, non-negative

matrix factorization and image deconvolution (deblurring) to estimate the underlying segmentation.

## 2.3 Spectral Solvers

So far we have been dealing with the problem of partitioning a graph representing an image in order to obtain a segmentation. In the previous sections, we explored how this can be accomplished via appearance modeling and graph cuts. Now, we will focus on some algorithms to directly partition the image graph without any reference to the region appearance distributions, even implicitly. To that goal, we shall redefine the original graph weights to enforce that pixels within a partition should have similar color appearance and that nearby pixels should generally be given the same labels. In other words, the edge weights should be defined as similarity functions between pixels, i.e., the more similar pixels  $i$  and  $j$  are, the greater  $w(i, j)$  is. A discussion on particular weight choices is explored in Section 2.3.2.

### 2.3.1 Normalized Cuts

Ideally, in image segmentation we would like the pixels in each partition to be similar to each other and dissimilar to the pixels from other partitions. Let  $(A, B)$  be a disjoint partition (a segmentation) of  $G$ . Suppose  $w(i, j)$  measures the similarity between pixels  $i$  and  $j$ . Then, a good partition choice could be that that minimizes the total edge weight that has been removed to produce  $A$  and  $B$ . This measure is what is called the cut between  $A$  and  $B$  on  $G$ :

$$\text{Cut}(A, B|G) = \sum_{i \in A, j \in B} w(i, j). \quad (2.30)$$

One would consider the optimal partition to be the one that minimizes the cut value. However, using such criteria heavily favors partitions of unbalanced sizes, as cutting small sets of isolated nodes in the graph produces very little cut values. To counter this issue, Shi and Malik [SM00] introduced the normalized cut criteria, which takes into account each partition's total edge connections to all the nodes in the graph. The normalized cut value is

then defined as:

$$\text{NCut}(A, B|G) = \frac{\text{Cut}(A, B|G)}{\text{Vol}(A|G)} + \frac{\text{Cut}(A, B|G)}{\text{Vol}(B|G)}, \quad (2.31)$$

where  $\text{Vol}(A|G) = \sum_{i \in A, j \in G} w(i, j)$  is called the volume of  $A$ . Using the normalized cuts criteria, the small cuts isolated nodes may produce are counterbalanced by the small volume of one of their partitions, which then entails large normalized cut values.

Unfortunately, as mentioned in [SM00], minimizing the normalized cut is NP-complete. Therefore, we have to rely in approximate solvers. The spectral algorithm introduced in [SM00] solves a continuous relaxation of the minimum NCut problem. Let  $W$  be the weighted adjacency matrix of  $G$  and let  $D$  be the diagonal degree matrix with  $D(i, i) = \sum_{j \in V} w(i, j)$ . The matrix  $L = D - W$  is the Laplacian of  $G$ . Let  $\mathbf{y} \in \{-1, 1\}^n$  be an indicator vector of the partition  $(A, B)$  and  $\mathbf{1}$  a vector of ones in  $n$  dimensions. Then, following the demonstration in [BC06], we have that:

$$\begin{aligned} \text{Cut}(A, B|G) &= \frac{1}{4}(\mathbf{1} + \mathbf{y})^\top W(\mathbf{1} - \mathbf{y}) = \frac{1}{4}(\mathbf{1}^\top W \mathbf{1} - \mathbf{y}^\top W \mathbf{y}) \\ &= \frac{1}{4}\mathbf{y}^\top (D - W)\mathbf{y} = \frac{1}{4}\mathbf{y}^\top L\mathbf{y}. \end{aligned} \quad (2.32)$$

Using the same reasoning, we have that  $\text{Vol}(A|G) = \mathbf{1}^\top D(\mathbf{1} - \mathbf{y})$  and  $\text{Vol}(B|G) = \mathbf{1}^\top D(\mathbf{1} + \mathbf{y})$ . Finally, introducing the new variables  $v_A$  and  $v_B$ , we have that the minimum normalized cut problem can be stated as:

$$\begin{aligned} \min_{\mathbf{y}, v_A, v_B} \quad & \frac{1}{4} \left( \frac{\mathbf{y}^\top L \mathbf{y}}{v_A} + \frac{\mathbf{y}^\top L \mathbf{y}}{v_B} \right) \\ \text{s.t.} \quad & \mathbf{1}^\top D(\mathbf{1} - \mathbf{y}) = 2v_A, \quad \mathbf{1}^\top D(\mathbf{1} + \mathbf{y}) = 2v_B \\ & \mathbf{y} \in \{-1, +1\}^n \end{aligned} \quad (2.33)$$

Defining  $\mathbf{d} = D\mathbf{1}$  and noticing that  $\mathbf{y}^\top D\mathbf{y} = v_A + v_B$ , the above problem can be restated as:

$$\begin{aligned} \min_{\mathbf{y}, v_A, v_B} \quad & \left( \frac{v_A + v_B}{4v_A v_B} \right) \mathbf{y}^\top L \mathbf{y} \\ \text{s.t.} \quad & \mathbf{d}^\top \mathbf{y} = v_A - v_B, \quad \mathbf{d}^\top \mathbf{1} = v_A + v_B \\ & \mathbf{y} \in \{-1, +1\}^n \end{aligned} \quad (2.34)$$

Now, define  $\mathbf{x} \in \mathbb{R}^n$  as

$$\mathbf{x} = \sqrt{\frac{v_A + v_B}{4v_A v_B}} \left( I - \frac{\mathbf{1}\mathbf{d}^\top}{\mathbf{d}^\top \mathbf{1}} \right) \mathbf{y}. \quad (2.35)$$

Note that  $\mathbf{d}^\top \mathbf{x} = 0$ . Replacing  $\mathbf{y}$  by  $\mathbf{x}$  in Eq. 2.34 and adding the (redundant) constraint  $\mathbf{x}^\top D\mathbf{x} = 1$  to the minimization problem, we have:

$$\begin{aligned} \min_{\mathbf{x}, v_A, v_B} \quad & \mathbf{x}^\top L\mathbf{x} \\ \text{s.t.} \quad & \mathbf{d}^\top \mathbf{x} = 0, \quad \mathbf{d}^\top \mathbf{1} = v_A + v_B, \quad \mathbf{x}^\top D\mathbf{x} = 1 \\ & \mathbf{x} \in \left\{ -\sqrt{\frac{v_A}{(v_A + v_B)v_B}}, \sqrt{\frac{v_B}{(v_A + v_B)v_A}} \right\}^n \end{aligned} \quad (2.36)$$

Minimizing only on  $\mathbf{x}$  and dropping the combinatorial constraint on it, we have the following relaxed version of the above problem:

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} \quad & \mathbf{x}^\top L\mathbf{x} \\ \text{s.t.} \quad & \mathbf{1}^\top D\mathbf{x} = 0, \quad \mathbf{x}^\top D\mathbf{x} = 1. \end{aligned} \quad (2.37)$$

which can be solved via the generalized eigenvalue problem:

$$L\mathbf{x} = \lambda D\mathbf{x}. \quad (2.38)$$

The algorithm then selects the generalized eigenvector  $\mathbf{x}$  corresponding to the second smallest eigenvalue (the smallest eigenvalue is trivially equal to 0 for connected graphs), and partitions  $V$  by thresholding  $\mathbf{x}$ . Figure 2.9 shows an example of eigenvector and segmentation computed according to the Normalized Cut algorithm described above.

The information contained in the other eigenvectors can be used in  $K$ -class clustering problems. As shown in [MS01] and [NJW01], the idea is to treat each row of the matrix containing the first  $K + 1$  generalized eigenvectors of  $L$  as an embedding of the original pixel data and cluster it into  $K$  clusters using  $K$ -means (or any other algorithm). Another approach to generalize the NCut objective and algorithm to multiple classes can be found in [YS03].

The NCut criteria and algorithm have been applied in many data clustering and computer

vision contexts. The authors in [AMFM10], for instance, used the generalized eigenvectors of the normalized cut algorithms to provide global cues for edge detection. In [TDP<sup>+</sup>18], the normalized cut criteria is used as a loss function to train deep convolutional neural networks in supervised image segmentation tasks. In [TMAB16], it replaces the appearance terms in MRF-based segmentation algorithms. On the theory side, the work in [DGK04] showed the correspondence between the normalized cut and kernel  $K$ -means algorithms.

Finally, despite the impact and theoretical implication of normalized cuts, it is worth mentioning that other cut criteria were proposed and studied for image segmentation, for instance, mean cut [WS01] and ratio cut [WS03], which will not be discussed in this thesis.

### 2.3.2 Weight choice

Choosing the weights of  $G$  is crucial when designing spectral algorithms. In graph-based image processing, the common choice of weighing dates back to the conception of bilateral filters [TM98] and combines two grouping cues in a single weight:

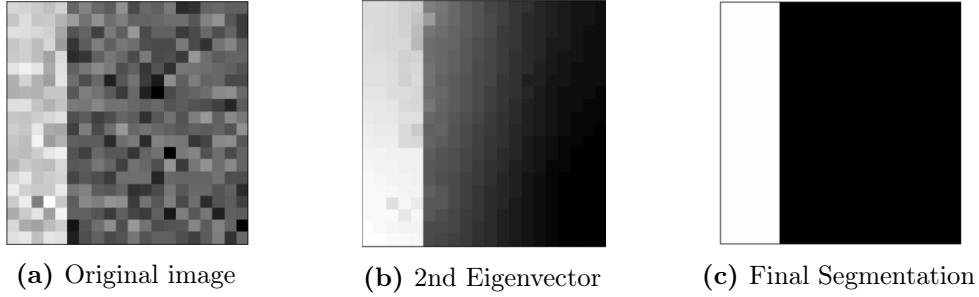
$$w(i, j) = \exp\left(-\frac{\|I(i) - I(j)\|^2}{2\sigma_I^2}\right) \exp\left(-\frac{\|X(i) - X(j)\|^2}{2\sigma_X^2}\right), \quad (2.39)$$

where  $\sigma_I$  and  $\sigma_X$  are fixed parameters set by the user and  $X(\cdot)$  is the location in two dimensions of a pixel in the image. In the original normalized cuts paper, it was also added an indicator function of whether  $\|X(i) - X(j)\| < r$  to  $w(i, j)$ , where  $r$  is also chosen by the user. The goal is to sparsify  $G$  in order to reduce the eigenvector computational burden. Figure 2.9 shows a segmentation result using the normalized cuts algorithm as proposed in [SM00].

Besides using the intensity values in Eq. 2.39, the authors in [SM00] also proposed the usage of filter responses at various scales and orientations in order to model texture information [MBLS01].

Further developments on formulating the weight function for graph-based segmentation methods also considered the use of intervening contour cue [AMFM10, MVM11, CC15]. The intervening contour concept suggests that pixels on the two different sides of a boundary are more likely to belong to different segments. The affinity between pixels is measured





**Figure 2.9:** Segmentation result using the original Normalized Cuts formulation. Adapted from [SM00].

as the maximum gradient magnitude or the maximum probability of boundary (Pb) value [MFM04] on a straight-line path between them.

### 2.3.3 Spectral clustering and random walks

In [MS01] the NCut algorithm is described in terms of a Markov chain. Let  $P = D^{-1}W$ . The matrix  $P$  is the transition matrix of a Markov chain over the vertices  $V$ . The long term behavior of this Markov chain can be characterized by the solutions to the eigenvector problem

$$P\mathbf{x} = \lambda\mathbf{x}. \quad (2.40)$$

A solution  $(\lambda, \mathbf{x})$  to the eigenvector problem in Eq. 2.40 leads to a solution  $(1 - \lambda, \mathbf{x})$  to the generalized eigenvector problem in Eq. 2.38 and vice-versa. Therefore the generalized eigenvector  $\mathbf{x}$  used in the NCut algorithm corresponds to the eigenvector of  $P$  with second largest eigenvalue.

This interpretation of the solutions  $\mathbf{x}$  of Eq. 2.40 also leads to an interpretation of the NCut objective as a random walk. Let  $(A, B)$  be a partition of  $V$  and  $P(A \rightarrow B|A)$  be the probability that a walker transitions from a vertex in  $A$  to a vertex in  $B$  in one step if the current state is in  $A$  and its walk is started in the chain's stationary distribution  $\pi^\infty = \frac{\mathbf{d}}{\text{Vol}(V)}$ .

One can show that:

$$P(A \rightarrow B|A) = \frac{\sum_{i \in A, j \in B} w(i, j)}{\text{Vol}(A)}, \quad (2.41)$$

which entails the following:

$$\text{NCut}(A, B|G) = P(A \rightarrow B|A) + P(B \rightarrow A|B). \quad (2.42)$$

In other words, it means that minimizing the NCut objective has the effect of computing a partition of  $V$  such that, once the walker is in one of the parts, the probability of evading it is low. This idea can also be tied to the concept of low-conductivity sets in Markov chains [MS01].

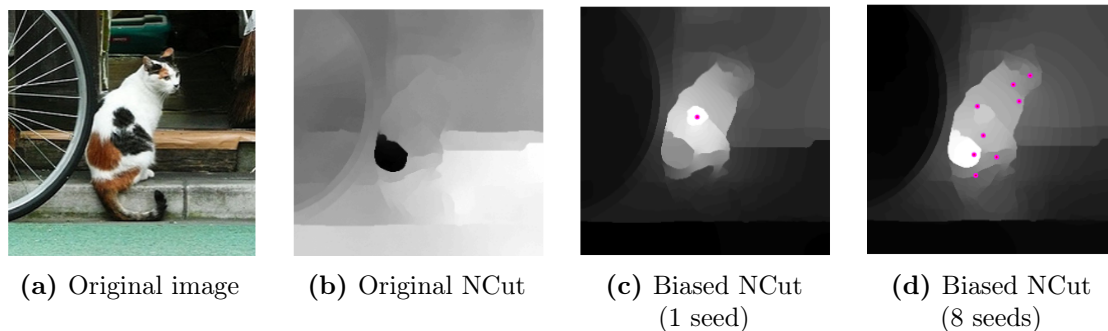
### 2.3.4 Practical Considerations

On the implementation side, the normalized cuts algorithm poses two main problems when applied to image segmentation: how to efficiently construct the graph over the pixel set and how to perform the spectral decomposition. In order to solve them, the usual approach is to restrict the pixel affinities to take place only between pixels within a small neighbourhood, as mentioned in Section 2.3.2. That leads to a sparse graph, in which the spectral decomposition can be solved efficiently with relatively low memory usage. Another possible solution is to oversegment the original image into superpixels [ASS<sup>+</sup>12], construct a similarity graph of them. The eigenvector resulting from the new affinity matrix can then be interpolated back to the image resolution and thresholded for the final segmentation [CC15].

Furthermore, works such as [PF98, FBCM04, LC10] attenuate the computation burden by relying on approximate eigenvector solvers. In [PF98], the easy-to-compute first eigenvector of the similarity matrix is used to separate figure from background. In [FBCM04], Nystrom low-rank approximation to the similarity matrix before the eigenvalue decomposition takes place. In [LC10], a truncated power iteration is proposed to approximately compute the second generalized eigenvector in 2.38.

### 2.3.5 Adding constraints to Normalized cuts

Similarly to what was discussed in model-based segmentation, it is often necessary to add constraints to segmentation results that typically stem from seed information. Following the onset of spectral methods for image segmentation, some effort was put in trying to



**Figure 2.10:** Biased Normalized Cuts result. Adapted from [MVM11].

adapt these new algorithms to make use of constraints imposed by the user. One of its first attempts is found in [YS01], where the authors generalized the original NCut problem by making it subject to constraints of the form  $U\mathbf{x} = 0$ , to which they also develop a spectral algorithm. This result was further generalized to accommodate for general linear constraints in the work of [EOK11]. In [MVM11], on the other hand, Maji *et al.* showed that enforcing constraints in this manner is not robust when the constraints are noisy and proposed an algorithm that alters, with very little additional time, the unconstrained problem's solution to take the imposed constraints into account. Their method, called Biased Normalized Cuts, can then be run and tested under different sets of seeds with little additional computational burden. Figure 2.10 shows the resulting eigenvectors when different sets of seeds are imposed to the original normalized cut eigenvector. Noticing that the Biased NCut method does not accommodate for two different sets of seeds, one of the background labeling and the other for the foreground, Chew and Cahill [CC15] proposed a spectral method that deals with both must-link constraints (among seeds of the same type) and cannot-Link constraints (between seeds of different kind).

### 2.3.6 Multiview Spectral Clustering

Finally, spectral clustering techniques have also been adapted to multiview learning. In such problem, one is concerned on extracting information from data generated from multiple different feature collectors (or views). For instance, in multimedia-content understanding, multimedia segments can be simultaneously described by their video and audio signals; and,

in content-based web-image retrieval, an object is simultaneously described by visual features from the image and the text surrounding it. One of the earlier results on clustering within this paradigm can be traced to [BS04], where the authors propose multiview counterparts of traditional clustering algorithms, such as EM and  $K$ -means.

Within the spectral framework, the work presented in [DS05] proposes a two-view clustering algorithm that computes the normalized eigenvectors arising from a bipartite graph that encodes the views. The authors in [KRD11] describe an multiview iterative procedure that enforces consistency among views by solving the generalized eigenvalue problem for each view separately using the eigenvectors from other views computed in previous iterations. Finally, other relevant approaches to multiview clustering consist in mixing the graph of different views into a sole graph either via their a convex combination [ZB07, ZZLY18] or according to their Laplacian's power mean [MGTH18]. To the best of our knowledge, no multiview clustering algorithm has been developed to treat image segmentation. One possible explanation for this lack of literature is due to the scaling issues that many of the current multi-view clustering methods face.

# CHAPTER 3

## Direct Estimation of Appearance Models

In this chapter, we describe a novel approach for estimating appearance models directly from an image, without explicit consideration of the pixels that make up each region. Our approach is based on algebraic expressions that relate local image statistics to the appearance models of spatially coherent regions. We describe two algorithms that can use second order pixel intensity statistics for estimating appearance models for images with two regions. The first algorithm is based on solving a system of linear and quadratic equations. The second algorithm is a spectral method based on an eigenvector computation. We also propose an estimator using third-order statistics. that is able to that estimate appearance models of multi-region images and the areas of their respective regions We present experimental results that demonstrate the proposed methods work well in practice and lead to effective image segmentation algorithms.

### 3.1 Appearance Models and Image Segmentation

Let  $I$  be an image of  $L$  colors. Recalling Section 2.1.3, we define an appearance model  $\theta \in \mathbb{R}^L$  as the distribution of the typical values (colors) of pixels in a given image region. In graph-based image segmentation, its is very common to use the appearance models of each of the  $I$ 's constituent regions to better distinguish them. In particular, the usage of appearance models can be broadly categorized into three classes:

**The appearances are known *a priori*** The appearances are an input to the segmentation algorithm. They are either entirely provided by the user or estimated via other external data, such as seeds [BJ01, TBAMB15].

**The appearances are *iteratively optimized with the segmentation*** Departing from crude segmentations estimates, the algorithm alternates between fitting the models and segmented the images given the fitted models [RKB04]. A further explanation of these methods is found in Section 2.2.2.

**The appearances are *jointly optimized with the segmentation*** The segmentation and appearance models are fitted simultaneously in a common optimization problem [VKR09, TGV13]. Section 2.2.4 discusses these methods more deeply.

The methods in this chapter are concerned with the first category of segmentation solvers, but in a novel manner. We propose new algorithms to estimate the non-parametric appearance models without the user interaction. That would address some problem faced by the other categories, such as runtime and lack of generalization, while also introducing the usage of high-order moments and estimators to graph-based segmentation.

## 3.2 Image Statistics

In order to approach the appearance model estimation problem, we treat the image as a realization of a random field and consider two distributions that can be directly estimated from an observed image.

Let  $x, y \in \Omega$  be a pair of pixels at a fixed distance  $r$  from each other, selected uniformly at random. Since the pixels are in a discrete grid we use the L1 norm to measure the distance between them.

$$Q = \{(x, y) \in \Omega^2 \mid \|x - y\| = r\}.$$

$$(x, y) \sim \text{Uniform}(Q).$$

Let  $\alpha \in \mathbb{R}^L$  be a distribution over  $L$  where  $\alpha(i)$  is the probability that pixel  $x$  has value  $i$ . Let  $\beta \in \mathbb{R}^{L \times L}$  be a distribution over  $L \times L$  where  $\beta(i, j)$  is the probability that pixel  $x$  has value  $i$  and pixel  $y$  has value  $j$ .

$$\alpha(i) = P(I(x) = i).$$

$$\beta(i, j) = P(I(x) = i, I(y) = j).$$

Note that we can easily estimate  $\alpha$  and  $\beta$  from an observed image. We simply enumerate all pairs of pixels at distance  $r$  from each other and count the number of times we observe pixels with particular values. If the image is large we can also estimate the two distributions using a random sample of pairs of pixels at distance  $r$ . We use  $\hat{\alpha}$  and  $\hat{\beta}$  to denote the estimates of  $\alpha$  and  $\beta$  computed from an observed image.

$$\hat{\alpha}(i) = \frac{1}{|Q|} \sum_{(x,y) \in Q} \mathbb{1}(I(x) = i).$$

$$\hat{\beta}(i, j) = \frac{1}{|Q|} \sum_{(x,y) \in Q} \mathbb{1}(I(x) = i) \mathbb{1}(I(y) = j).$$

Let  $\mathcal{R}_0$  and  $\mathcal{R}_1$  be a partition of  $\Omega$  into two regions. Let  $w_0 = P(x \in \mathcal{R}_0)$  and  $w_1 = P(x \in \mathcal{R}_1)$ . Now consider the probability that  $x$  and  $y$  are in different regions. Since  $x$  and  $y$  are exchangeable, let  $\epsilon = P(x \in \mathcal{R}_0, y \in \mathcal{R}_1) = P(x \in \mathcal{R}_1, y \in \mathcal{R}_0)$ . The probability that  $x$  and  $y$  are in different regions is  $2\epsilon$ .

Now let  $\theta_0$  and  $\theta_1$  be appearance models associated with  $\mathcal{R}_0$  and  $\mathcal{R}_1$  respectively. We will relate  $\alpha$  and  $\beta$  to  $w_0$ ,  $w_1$ ,  $\theta_0$ ,  $\theta_1$  and  $\epsilon$  using two key assumptions. The first assumption is that regions have homogeneous appearance in the following sense.

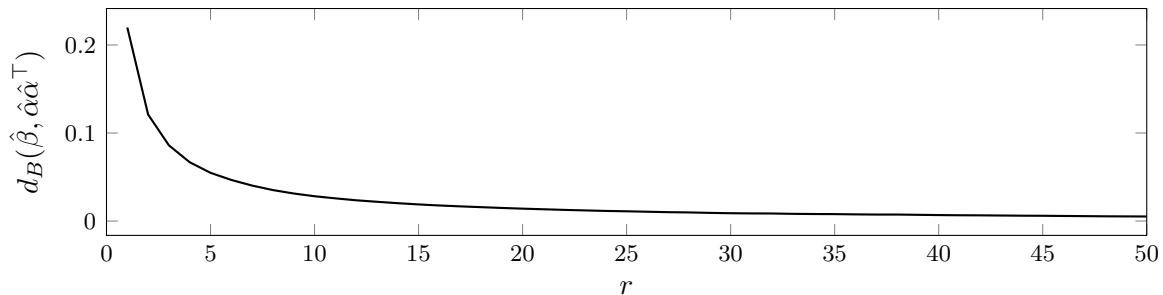
**Assumption 1** (Homogeneity). *The probability that a pixel takes a particular value depends only on the region the pixel belongs to,*

$$P(I(x) = i | x \in \mathcal{R}_s) = \theta_s(i).$$

Note that this assumption does not specify a full generative model for the image. We assume the pixels in each region have the same marginal distribution, but their joint distribution could involve dependencies, as we see for example in images with textures. Similar assumptions have been used in other approaches for image segmentation, including several methods for unsupervised texture segmentation [NBCE09, YWC15, MMF<sup>+</sup>14].

The second assumption captures the idea that sufficiently far away pixels are independent.

**Assumption 2** (Independence at a distance). *If  $x$  and  $y$  are two pixels with  $\|x - y\| = r$*



**Figure 3.1:** Evaluating independence at a distance. We show the average Bhattacharyya distance between  $\hat{\beta}$  and  $\hat{\alpha}\hat{\alpha}^\top$  as a function of  $r$  for images with a single Brodatz texture (see Figure 3.7).

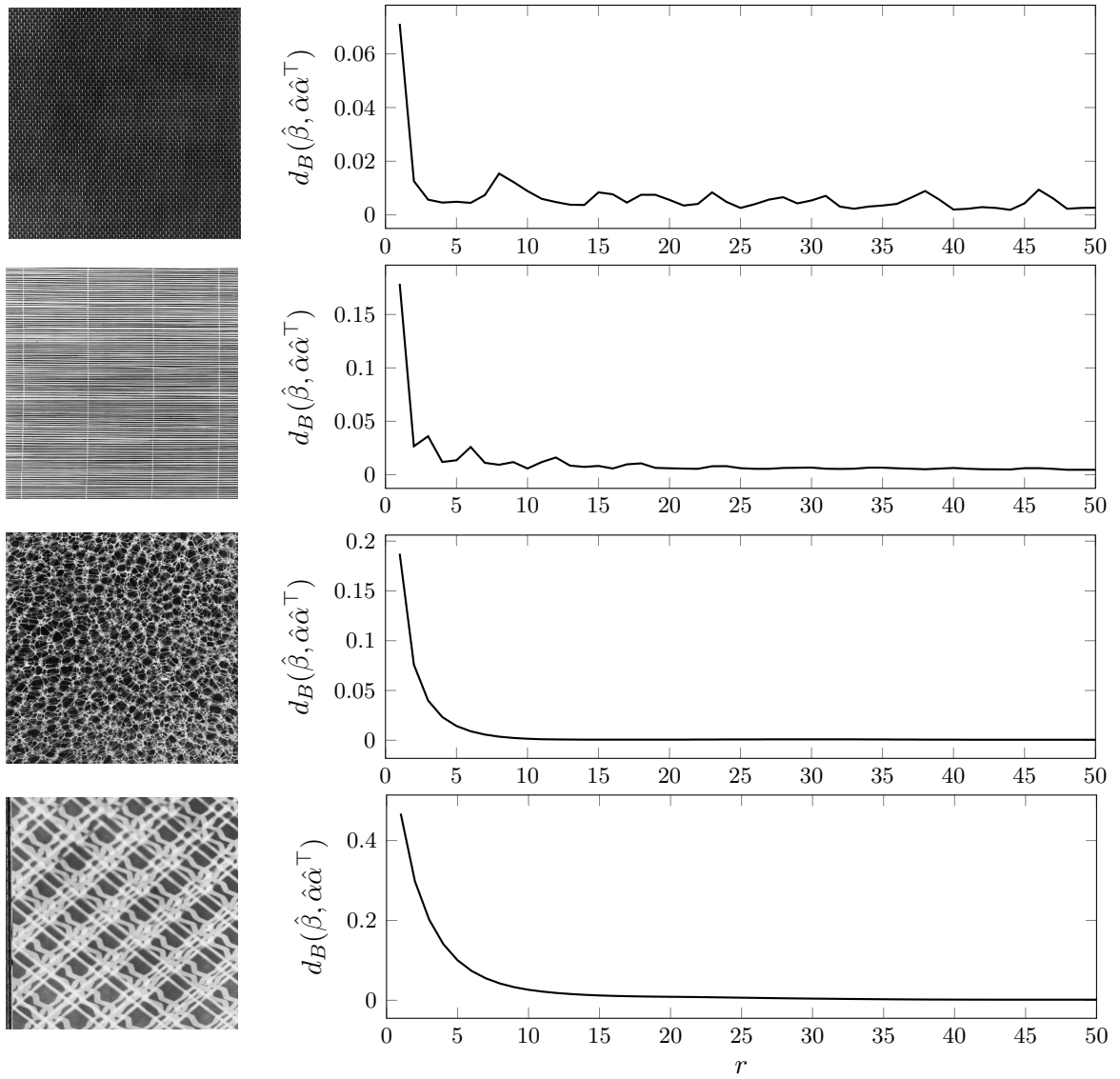
then,

$$P(I(x) = i, I(y) = j | x \in \mathcal{R}_s, y \in \mathcal{R}_t) = P(I(x) = i | x \in \mathcal{R}_s)P(I(y) = j | y \in \mathcal{R}_t).$$

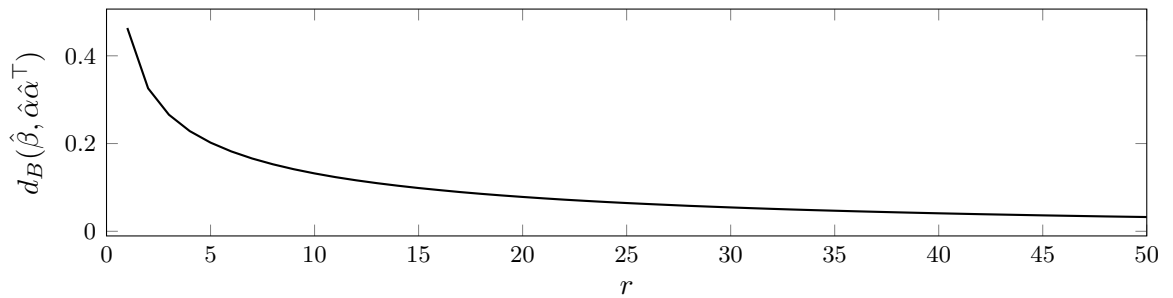
For images with a single region Assumption 2 is equivalent to  $\beta = \alpha\alpha^\top$ . Figure 3.1 evaluates this assumption in textured images from the Brodatz dataset [Bro66]. See Figure 3.7 for examples of the images in the Brodatz dataset. In this case each image has a single textured region and we compare  $\hat{\beta}$  to  $\hat{\alpha}\hat{\alpha}^\top$  for different values of  $r$  using the Bhattacharyya distance (see Section 3.6.1). When  $r$  is small we see that the two distributions,  $\hat{\beta}$  and  $\hat{\alpha}\hat{\alpha}^\top$ , are quite different, because nearby pixels are not independent. As we increase  $r$  we see that  $\hat{\beta}$  is close to  $\hat{\alpha}\hat{\alpha}^\top$ , suggesting that pixels that are relatively far from each other are independent. In Figure 3.2, we show this effect in individual textures. From it, we observe that in highly regular textures there is trepidation as the distance between  $\hat{\beta}$  and  $\hat{\alpha}\hat{\alpha}^\top$  decays with the increase in  $r$ . Despite that, in most textures, we observed a smooth decay, mimicking what is shown in the average experiment of Figure 3.1.

We also evaluated Assumption 2 in real scenes. Figure 3.3 we show the same experiment as in Figure 3.1, but now considering the foreground and background segments of the images in the Grabcut Dataset [RKB04]. Figure 3.4 shows the same experiments for individual images in that same dataset. For computational simplicity, we converted all the evaluated images in grayscale before running the experiments. Note that the same pattern found in 3.1 is observed, bringing more evidence of the realistic usage of Assumption 2.

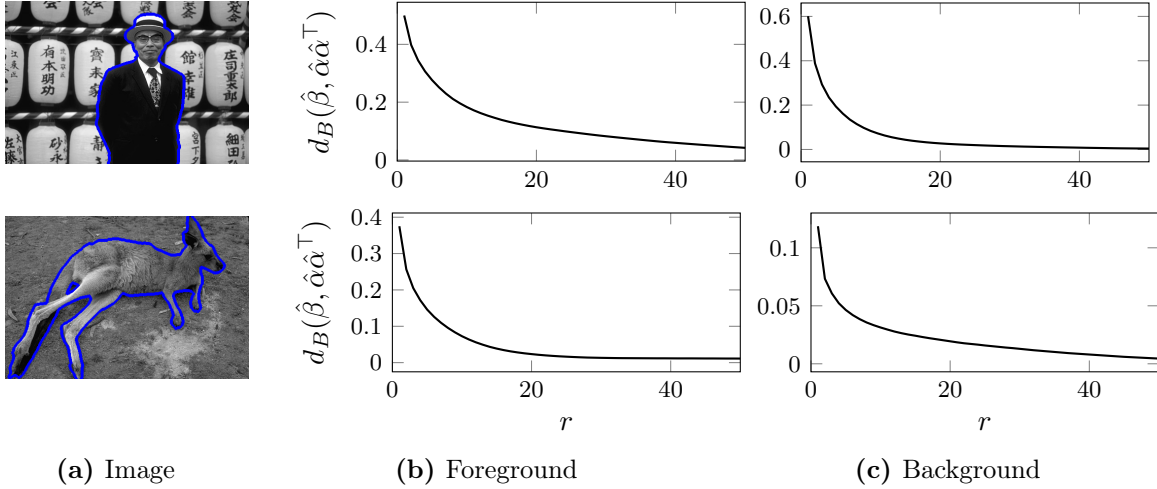




**Figure 3.2:** Evaluating independence at a distance. We proceed as in Figure 3.1, but for individual textures (on the left).



**Figure 3.3:** Evaluating independence at a distance. We show the average distance between  $\hat{\beta}$  and  $\hat{\alpha}^\top$  as a function of  $r$  for each region segment of the images in the Grabcut Dataset [RKB04].



**Figure 3.4:** Evaluating independence at a distance. We proceed as in Figure 3.3, but now for individual images in the Grabcut Dataset and for their respective foreground and background segments.

The following proposition shows how the two assumptions above lead to algebraic expressions relating  $\alpha$  and  $\beta$  to  $w_0$ ,  $w_1$ ,  $\theta_0$ ,  $\theta_1$  and  $\epsilon_r$ . These algebraic expressions will enable us to estimate  $\theta_0$  and  $\theta_1$  without explicit consideration of  $\mathcal{R}_0$  and  $\mathcal{R}_1$ .

**Proposition 1.** *Under Assumptions 1 and 2 we have:*

$$\alpha = w_0\theta_0 + w_1\theta_1. \quad (3.1)$$

$$\beta = (w_0 - \epsilon_r)\theta_0\theta_0^\top + (w_1 - \epsilon_r)\theta_1\theta_1^\top + \epsilon_r\theta_0\theta_1^\top + \epsilon_r\theta_1\theta_0^\top. \quad (3.2)$$

*Proof.* For (3.1) let  $x$  be a pixel selected uniformly at random. Then,

$$\begin{aligned} \alpha(i) &= P(I(x) = i) \\ &= P(x \in \mathcal{R}_0)P(I(x) = i | x \in \mathcal{R}_0) + P(x \in \mathcal{R}_1)P(I(x) = i | x \in \mathcal{R}_1) \\ &= w_0\theta_0(i) + w_1\theta_1(i) \end{aligned}$$

For (3.2) let  $x$  and  $y$  be two pixels with  $\|x - y\| = r$  selected uniformly at random. Note that

$$P(x \in \mathcal{R}_0) = P(x \in \mathcal{R}_0, y \in \mathcal{R}_0) + P(x \in \mathcal{R}_0, y \in \mathcal{R}_1).$$

Therefore  $P(x \in \mathcal{R}_0, y \in \mathcal{R}_0) = w_0 - \epsilon_r$ . Similarly  $P(x \in \mathcal{R}_1, y \in \mathcal{R}_1) = w_1 - \epsilon_r$ . Now,

$$\begin{aligned}
\beta(i, j) &= P(I(x) = i, I(y) = j) \\
&= P(x, y \in \mathcal{R}_0)P(I(x) = i, I(y) = j \mid x, y \in \mathcal{R}_0) + \\
&\quad P(x, y \in \mathcal{R}_1)P(I(x) = i, I(y) = j \mid x, y \in \mathcal{R}_1) + \\
&\quad P(x \in \mathcal{R}_0, y \in \mathcal{R}_1)P(I(x) = i, I(y) = j \mid x \in \mathcal{R}_0, y \in \mathcal{R}_1) + \\
&\quad P(x \in \mathcal{R}_1, y \in \mathcal{R}_0)P(I(x) = i, I(y) = j \mid x \in \mathcal{R}_1, y \in \mathcal{R}_0) \\
&= (w_0 - \epsilon_r)\theta_0(i)\theta_0(j) + (w_1 - \epsilon_r)\theta_1(i)\theta_1(j) + \epsilon_r\theta_0(i)\theta_1(j) + \epsilon_r\theta_1(i)\theta_0(j)
\end{aligned}$$

□

### 3.3 Appearance Estimation

As discussed in the previous section we can estimate  $\alpha$  and  $\beta$  directly from an image. In this section we show how we can recover  $\theta_0$  and  $\theta_1$  from  $\alpha$  and  $\beta$  or their estimates  $\hat{\alpha}$  and  $\hat{\beta}$ . Initially we assume that  $w_0$ ,  $w_1$  and  $\epsilon$  are known, and present two different methods (Sections 3.3.1 and 3.3.2) for estimating  $\theta_0$  and  $\theta_1$ . We consider the case where  $w_0$ ,  $w_1$  and  $\epsilon$  are unknown in Section 3.3.3.

#### 3.3.1 Algebraic Method

Suppose  $\alpha$ ,  $\beta$ ,  $w_0$ ,  $w_1$  and  $\epsilon$  are known, and consider the problem of recovering  $\theta_0$  and  $\theta_1$ . Let  $L = \{1, \dots, k\}$ . We have  $2k$  unknowns  $\theta_0(1), \dots, \theta_0(k)$  and  $\theta_1(1), \dots, \theta_1(k)$ . Proposition 1 defines  $k$  linear and  $k^2$  quadratic constraints,

$$\alpha(i) = w_0\theta_0(i) + w_1\theta_1(i). \quad (3.3)$$

$$\beta(i, j) = (w_0 - \epsilon)\theta_0(i)\theta_0(j) + (w_1 - \epsilon)\theta_1(i)\theta_1(j) + \epsilon\theta_0(i)\theta_1(j) + \epsilon\theta_1(i)\theta_0(j). \quad (3.4)$$

Since  $\theta_0$  and  $\theta_1$  define probability distributions we also have the additional constraints that both vectors should sum to one and  $\theta_0(i) \geq 0$ ,  $\theta_1(i) \geq 0$  for  $1 \leq i \leq k$ .

**Minimal constraints** We first consider a simple method that uses a subset of the constraints to solve for all of the unknowns in the appearance models. The approach uses the  $k$  linear constraints defined by  $\alpha$  and  $k$  quadratic constraints defined by a single row of  $\beta$ . For the derivation below we treat  $\alpha$  and  $\beta$  as known although in practice we only have empirical estimates of the two distributions.

1. Let  $i \in L$ . Using the quadratic constraint defined by  $\beta(i, i)$  and the linear constraint defined by  $\alpha(i)$  we can solve for  $\theta_0(i)$  and  $\theta_1(i)$ ,

$$\theta_0(i) = \alpha(i) \pm \frac{\sqrt{\frac{w_0 w_1 - \epsilon}{w_1^2} (\beta(i, i) - \alpha^2(i))}}{\frac{w_0 w_1 - \epsilon}{w_1^2}}, \quad \theta_1(i) = \frac{\alpha(i) - w_0 \theta_0(i)}{w_1}. \quad (3.5)$$

2. Now consider each  $j \in L$  with  $j \neq i$ . Since we solved for  $\theta_0(i)$  and  $\theta_1(i)$  in Step 1, now  $\beta(i, j)$  defines a linear constraint on  $\theta_0(j)$  and  $\theta_1(j)$ . Together with the linear constraint defined by  $\alpha(j)$  we can solve for  $\theta_0(j)$  and  $\theta_1(j)$ ,

$$\theta_0(j) = \frac{w_1 \beta(i, j) - \alpha(j)(w_1 \theta_1(i) + \epsilon(\theta_0(i) - \theta_1(i)))}{(w_0 w_1 - \epsilon)(\theta_0(i) - \theta_1(i))}, \quad \theta_1(j) = \frac{\alpha(j) - w_0 \theta_0(j)}{w_1}. \quad (3.6)$$

When  $\beta(i, i) = \alpha^2(i)$  we have  $\theta_0(i) = \theta_1(i)$ . To avoid dividing by zero when solving for  $\theta_0(j)$  in Step 2 and to increase the robustness of the method, we can select  $i$  maximizing  $\beta(i, i) - \alpha^2(i)$  in Step 1. Note that if  $\beta(i, i) - \alpha^2(i) = 0$  for all  $i$  then  $\theta_0 = \theta_1 = \alpha$ .

To solve for  $\theta_0(i)$  in Step 1 we require that  $(w_0 w_1 - \epsilon)(\beta(i, i) - \alpha^2(i)) \geq 0$ . Proposition 2 below shows that under the assumptions we have made

$$\beta(i, i) - \alpha^2(i) = (w_0 w_1 - \epsilon)(\theta_0(i) - \theta_1(i))^2. \quad (3.7)$$

Therefore  $(w_0 w_1 - \epsilon)(\beta(i, i) - \alpha^2(i)) \geq 0$ .

**Least squares solution** The approach described above uses a small number of the constraints defined by  $\alpha$  and  $\beta$  to exactly recover  $\theta_0$  and  $\theta_1$ . However, in practice we only have empirical estimates of  $\alpha$  and  $\beta$ . We also don't expect real data to perfectly fit our assumptions. We now describe an alternative method that uses all of the constraints defined

by  $\alpha$  and  $\beta$  in a least squares formulation.

Let  $i_1, \dots, i_k$  be an ordering of  $L = \{1, \dots, k\}$ . Our empirical results show that ordering the indices in decreasing value of  $\hat{\beta}(i, i) - \hat{\alpha}^2(i)$  works well and is better than a random order.

1. We start by solving for  $\theta_0(i_1)$  and  $\theta_1(i_1)$  using the quadratic constraint defined by  $\hat{\beta}(i_1, i_1)$  and the linear constraint defined by  $\hat{\alpha}(i_1)$ .

$$\theta_0(i_1) = \hat{\alpha}(i_1) \pm \frac{\sqrt{\frac{w_0 w_1 - \epsilon}{w_1^2} (\hat{\beta}(i_1, i_1) - \hat{\alpha}^2(i_1))}}{\frac{w_0 w_1 - \epsilon}{w_1^2}}, \quad \theta_1(i_1) = \frac{\hat{\alpha}(i_1) - w_0 \theta_0(i_1)}{w_1}. \quad (3.8)$$

2. We iterate  $\ell$  from 2 to  $k$  and solve for  $\theta_0(i_\ell)$  and  $\theta_1(i_\ell)$  in each step. When solving for  $\theta_0(i_\ell)$  and  $\theta_1(i_\ell)$  we already have values for  $\theta_0(i_1), \dots, \theta_0(i_{\ell-1})$  and  $\theta_1(i_1), \dots, \theta_1(i_{\ell-1})$ . Therefore  $\hat{\beta}(i_1, i_\ell), \dots, \hat{\beta}(i_{\ell-1}, i_\ell)$  define  $\ell - 1$  linear constraints on  $\theta_0(i_\ell)$  and  $\theta_1(i_\ell)$ . Together with the constraint defined by  $\hat{\alpha}(i_\ell)$  we form a system with  $\ell$  linear equations and 2 unknowns that can be solved using linear least squares:

$$\forall j \in \{i_1, \dots, i_{\ell-1}\} \quad ((w_0 - \epsilon)\theta_0(j) + \epsilon\theta_1(j))\theta_0(i_\ell) + ((w_1 - \epsilon)\theta_1(j) + \epsilon\theta_0(j))\theta_1(i_\ell) = \hat{\beta}(j, i_\ell),$$

$$w_0\theta_0(i_\ell) + w_1\theta_1(i_\ell) = \hat{\alpha}(i_\ell).$$

3. We improve our estimates by iterating  $\ell$  from 1 to  $k$  and re-estimate  $\theta_0(i_\ell)$  and  $\theta_1(i_\ell)$  in each step. To re-estimate  $\theta_0(i_\ell)$  and  $\theta_1(i_\ell)$  we use  $\hat{\beta}(j, i_\ell)$  and the current values for  $\theta_0(j)$  and  $\theta_1(j)$  for  $j \neq i_\ell$  to define  $k - 1$  linear constraints. Together with the constraint defined by  $\hat{\alpha}(i_\ell)$  we form a system with  $k$  linear equations and 2 unknowns that can be solved using linear least squares:

$$\forall j \neq i_\ell \quad ((w_0 - \epsilon)\theta_0(j) + \epsilon\theta_1(j))\theta_0(i_\ell) + ((w_1 - \epsilon)\theta_1(j) + \epsilon\theta_0(j))\theta_1(i_\ell) = \hat{\beta}(j, i_\ell)$$

$$w_0\theta_0(i_\ell) + w_1\theta_1(i_\ell) = \hat{\alpha}(i_\ell)$$

Empirically we found that iterating over the entries one time using this method is enough to obtain improved results.

4. We set  $\theta_s(i) = \max(\theta_s(i), 0)$  and normalize  $\theta_0$  and  $\theta_1$  to add up to one. This ensures  $\theta_0$  and  $\theta_1$  define valid probability distributions.

Note that there are two choices for the value of  $\theta_0(i_1)$  when solving the quadratic equation in Step 1. We consider both choices to estimate full appearance models. We then compare  $\beta$  defined by the estimated models and Equation (3.2) to  $\hat{\beta}$  using the Bhattacharyya distance. We select the appearance models leading to the smaller Bhattacharyya distance.

### 3.3.2 Spectral Method

Now we describe a spectral method for estimating the appearance models. As in the previous section we assume  $w_0$ ,  $w_1$  and  $\epsilon$  are known. The following proposition provides the basis for the approach.

**Proposition 2.**

$$\beta - \alpha\alpha^\top = (w_0w_1 - \epsilon)(\theta_0 - \theta_1)(\theta_0 - \theta_1)^\top \quad (3.9)$$

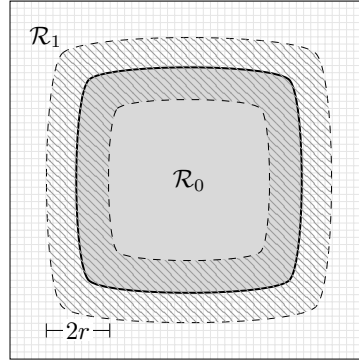
*Proof.* First note that  $w_0 + w_1 = 1$  implies  $w_0w_1 = w_0 - w_0^2$  and  $w_0w_1 = w_1 - w_1^2$ .

$$\begin{aligned} \beta - \alpha\alpha^\top &= (w_0 - \epsilon)\theta_0\theta_0^\top + (w_1 - \epsilon)\theta_1\theta_1^\top + \epsilon\theta_0\theta_1^\top + \epsilon\theta_1\theta_0^\top - (w_0\theta_0 + w_1\theta_1)(w_0\theta_0 + w_1\theta_1)^\top \\ &= (w_0 - \epsilon - w_0^2)\theta_0\theta_0^\top + (w_1 - \epsilon - w_1^2)\theta_1\theta_1^\top + (\epsilon - w_0w_1)\theta_0\theta_1^\top + (\epsilon - w_1w_0)\theta_1\theta_0^\top \\ &= (w_0w_1 - \epsilon)\theta_0\theta_0^\top + (w_0w_1 - \epsilon)\theta_1\theta_1^\top - (w_0w_1 - \epsilon)\theta_0\theta_1^\top - (w_1w_0 - \epsilon)\theta_1\theta_0^\top \\ &= (w_0w_1 - \epsilon)(\theta_0 - \theta_1)(\theta_0 - \theta_1)^\top \end{aligned}$$

□

Let  $u = \theta_0 - \theta_1$ . The above result shows that that matrix  $\beta - \alpha\alpha^\top$  is of rank one and its only eigenvector with non-zero eigenvalue is proportional to  $u$ . Moreover, the corresponding eigenvalue is  $(w_0w_1 - \epsilon)\|u\|^2$ .

The matrix  $\hat{\beta} - \hat{\alpha}\hat{\alpha}^\top$  defines an approximation to  $\beta - \alpha\alpha^\top$ . Let  $v$  be the dominant eigenvector of  $\hat{\beta} - \hat{\alpha}\hat{\alpha}^\top$  normalized so that  $\|v\| = 1$ . The vector  $v$  gives us an estimate of  $u/\|u\|$  up to a sign ambiguity. The corresponding eigenvalue  $\lambda$  can be used to approximate  $(w_0w_1 - \epsilon)\|u\|^2$ .



**Figure 3.5:** The area where pairs of pixels with  $\|x - y\| = r$  can be in different regions.

We can estimate  $u$  as,

$$\hat{u} = \pm \sqrt{\frac{\lambda}{w_0 w_1 - \epsilon}} v \quad (3.10)$$

We can then use  $\hat{\alpha}$  and Equation (3.1) to estimate  $\theta_0$  and  $\theta_1$ ,

$$\theta_0 = \hat{\alpha} + w_1 \hat{u},$$

$$\theta_1 = \hat{\alpha} - w_0 \hat{u}.$$

Finally, we set  $\theta_s(i) = \max(\theta_s(i), 0)$  and normalize  $\theta_0$  and  $\theta_1$  to add up to one. This ensures  $\theta_0$  and  $\theta_1$  define valid probability distributions.

To handle the sign ambiguity in Equation (3.10) we consider both choices to estimate  $\theta_0$  and  $\theta_1$ . We then compare  $\beta$  defined by the estimated appearance models and Equation (3.2) to  $\hat{\beta}$  using the Bhattacharyya distance. We select the choice of sign in Equation (3.10) leading to the smaller Bhattacharyya distance.

Notice that we can use simple power iteration methods to compute  $v$  and  $\lambda$ . The algorithm complexity depends only on  $|L|$ , which is usually much smaller than the number of pixels in the image. The rate of convergence of power iteration depends on the spectral gap, which is proportional to  $w_0 w_1 - \epsilon$ .

### 3.3.3 Estimating $w_0$ , $w_1$ and $\epsilon_r$

The methods described above assume  $w_0$ ,  $w_1$  and  $\epsilon$  are known. We have experimented with three different approaches for estimating the appearance models when  $w_0$ ,  $w_1$  and  $\epsilon$  are unknown. The first approach simply selects a typical, or average, value for each of the unknown parameters. The second approach involves an explicit search over a discretized set of choices for the unknown parameters. The third approach considers an alternation of appearance model and parameter estimation.

**Typical values** A simple approximation for  $w_0$  and  $w_1$  involves setting  $w_0 = w_1 = 0.5$ . Although this is a crude approximation we have found that it leads to good appearance models in a wide variety of images.

In order to approximate  $\epsilon$ , we start from the assumption that the ground truth segmentation  $S$  is spatially coherent and that the boundary between regions  $\partial S$  is short, i.e.,  $|\partial S| \approx \sqrt{|\Omega|}$ . In this case  $\epsilon$  is proportional to the area within distance  $r$  from  $\partial S$  divided by  $|\Omega|$ , see Figure 3.5. In our experiments we set  $r = \rho\sqrt{|\Omega|}$ , where  $\rho$  is a parameter set by the user. This makes the selection of the distance  $r$  be adaptive to the image resolution. Our estimate of  $\epsilon$  then becomes,

$$\epsilon = \kappa \frac{r\sqrt{|\Omega|}}{|\Omega|} = \kappa\rho. \quad (3.11)$$

We have found that setting  $\kappa = 0.5$  often leads to good results in practice.

**Searching over  $w_0$ ,  $w_1$  and  $\epsilon$**  To search over  $w_0$  and  $w_1$  we use the fact that  $w_0 + w_1 = 1$  and simply search over possible values for  $w_0$ . We assume without loss of generality that  $w_0 \leq w_1$  and  $w_0 \in (0, 0.5)$ . In practice we discretize the interval  $(0, 0.5)$  using a step of size of 0.05, leading to 11 choices for  $w_0$ . To estimate  $\epsilon$  we search over the interval  $(0, 0.1)$  using a step size of 0.01, leading to 11 choices for  $\epsilon$ . Together this leads to 121 combined choices for  $w_0$  and  $\epsilon$ .

For each choice of parameters  $w_0$ ,  $w_1$  and  $\epsilon$  we estimate  $\theta_0$  and  $\theta_1$  using either the algebraic or spectral method above. We then compare  $\beta$  defined by the estimated appearance models and Equation (3.2) to the empirical  $\hat{\beta}$  computed from the image. We select the model



parameters minimizing the Bhattacharyya distance between  $\beta$  and  $\hat{\beta}$ .

Searching over  $w_0$ ,  $w_1$  and  $\epsilon$  with the spectral method is fairly efficient because the bottleneck in the spectral method is computing the dominant eigenvector of  $\hat{\beta} - \hat{\alpha}\hat{\alpha}^\top$ . Since this matrix does not depend on the unknown parameters the eigenvector only has to be computed once. In this case searching for the parameters leads to limited overhead. Searching for the parameters with the algebraic method is much less efficient. The experiments in the Section 3.6 evaluate the running time of the different approaches.

**Alternating model and parameter estimation** We make use of the expressions in Proposition 1, to propose an alternating scheme for  $w_0$ ,  $w_1$  and  $\epsilon_r$  estimation:

1. Start with an initial guess of  $w_0$ ,  $w_1$  and  $\epsilon_r$ . In practice we initially set  $w_0 = w_1 = 0.5$  and  $\epsilon_r = 0.1$ .
2. Compute estimates for  $\theta_0$  and  $\theta_1$  using the current values of  $w_0$ ,  $w_1$  and  $\epsilon_r$ .
3. Using Eq. 3.1, compute a new estimate for  $w_0$  and  $w_1$  using the current values of  $\theta_0$  and  $\theta_1$  via least squares regression.
4. Using Eq. 3.2, compute a new estimate for  $\epsilon_r$  using the current values of  $\theta_0$ ,  $\theta_1$ ,  $w_0$  and  $w_1$  via least squares regression.
5. Normalize  $w_0$  and  $w_1$  to sum to be positive to one as we did in Section 3.3.1.
6. Iterate until the absolute difference of the current and the previous estimates  $w_0$ ,  $w_1$  and  $\epsilon_r$  is below a certain threshold  $t$ .

In our experimental section, we also test the case where we remove step 4 above and estimate  $\epsilon_r$  by searching over a finite range of value, as in the previous section. In practice we note that both algorithms always converge within at most 15 iterations, when  $t = 0.01$ .

### 3.4 Multi-region case

Our methods for direct appearance estimation shown above have two main drawbacks: (1) they only apply to binary segmentation problems and (2) they do not provide an "elegant"

algorithm to estimate the weights  $w_0$  and  $w_1$ , other than direct evaluation of some finite set of possible weights. In this section we will describe an algorithm that addresses both issues. Here, assume the image has  $K$  regions  $\mathcal{R}_1, \dots, \mathcal{R}_K$ , each with their own appearance model  $\theta_1, \dots, \theta_K \in \mathbb{R}^L$  and their respective areas  $w_1, \dots, w_K$ , normalized to sum to one. We also assume  $K$  is known.

### 3.4.1 Estimation when $r$ is small

Let  $\gamma \in \mathbb{R}^{L \times L \times L}$  be such that  $\gamma(i, j, k)$  is the probability that three pixels  $x, y, z \in \Omega$ , each  $r$  pixels apart from each other, selected uniformly at random have values  $i, j$  and  $k$ , respectively,

$$\gamma(i, j, k) = P(I(x) = i, I(y) = j, I(z) = k).$$

Note that  $\gamma$  can be estimated in the same fashion as  $\beta$ , by counting. Let  $p_r$  be proportional to the probability that two pixels  $x, y$  and  $z$  do not belong to the same region for a given  $r$ . Initially assume that  $r$  is small enough so  $p_r$  is negligible, while still satisfying Assumption 2. In other words, we assume that the pixels  $i, j$  and  $k$ , drawn uniformly at random in  $I$ , are close enough, so it is reasonable to expect them to belong to the same region. Furthermore, we expect their intensities  $I(i), I(j)$  and  $I(k)$  to still be independent in that setting. An example of such images are those whose pixel intensities are drawn independently from a distribution that only depends on the region they fall in.

**Proposition 3.** *For an image with  $K$  regions, under Assumptions 1 and 2, and  $p_r$  negligible, we have:*

$$\alpha = \sum_{s=1}^K w_s \theta_s. \quad (3.12)$$

$$\beta \approx \sum_{s=1}^K w_s \theta_s \theta_s^\top. \quad (3.13)$$

$$\gamma \approx \sum_{s=1}^K w_s \theta_s \otimes \theta_s \otimes \theta_s, \quad (3.14)$$

where  $\otimes$  represents the Kronecker product between vectors<sup>1</sup>.

---

<sup>1</sup>Here, the Kronecker product  $v_1 \otimes v_2 \otimes \dots \otimes v_m$  of  $m$  vectors  $v_1, v_2, \dots, v_m \in \mathbb{R}^N$  is a tensor  $T \in \mathbb{R}^{N \times N \times \dots \times N}$

*Proof.* For (3.12), let  $x \in \Omega$  be chosen uniformly at random. Then, we have:

$$\alpha(i) = P(I(x) = i) = \sum_{s=i}^K P(x \in \mathcal{R}_s)P(I(x) = i | x \in \mathcal{R}_s) = \sum_{s=i}^K w_s \theta_0(i)$$

For (3.13), let  $x, y \in \Omega$  with  $\|x - y\| = r$  be chosen uniformly at random and assume  $p_r$  (and therefore  $P(x \in \mathcal{R}_s, y \in \mathcal{R}_t)$  for  $s \neq t$ ) be negligible, while holding Assumption 2. Then, we have:

$$\begin{aligned} \beta(i, j) &= P(I(x) = i, I(y) = j) \\ &\approx \sum_{s=1}^K P(x, y \in \mathcal{R}_s)P(I(x) = i, I(y) = j | x, y \in \mathcal{R}_s) \\ &\approx \sum_{s=1}^K w_s P(I(x) = i | x \in \mathcal{R}_s)P(I(y) = j | y \in \mathcal{R}_s) \\ &= \sum_{s=1}^K w_s \theta(i)\theta(j), \end{aligned}$$

Since  $p_r$  is assumed to be negligible, it follows that  $P(x \in \mathcal{R}_s, y \in \mathcal{R}_t, z \in \mathcal{R}_u)$  is also small except when  $s = t = u$ . This leads to  $P(x \in \mathcal{R}_s) = P(y \in \mathcal{R}_s) = P(z \in \mathcal{R}_s) \approx P(x, y, z \in \mathcal{R}_s), \forall s$ . Now, for (3.14),

$$\begin{aligned} \gamma(i, j, k) &= P(I(x) = i, I(y) = j, I(z) = k) \\ &\approx \sum_{s=1}^K P(x, y, z \in \mathcal{R}_s)P(I(x) = i, I(y) = j, I(z) = k | x, y, z \in \mathcal{R}_s) \\ &\approx \sum_{s=1}^K w_s P(I(x) = i | x \in \mathcal{R}_s)P(I(y) = j | y \in \mathcal{R}_s)P(I(z) = k | z \in \mathcal{R}_s) \\ &= \sum_{s=1}^K w_s \theta(i)\theta(j)\theta(k), \end{aligned}$$

where the third equality is due to the independence assumption from Assumption 2.  $\square$

In Eqs. 3.12-3.14 there is a clear application of the more general estimation setting proposed

---

defined as:

$$T(a_1, a_2, \dots, a_m) = v_1(a_1)v_2(a_2) \dots v_m(a_m), \quad (3.15)$$

where  $a_1, a_2, \dots, a_m \in \{1, 2, \dots, N\}$ . Note that  $v_1 v_2^\top$  can be written as a Kronecker product  $v_1 \otimes v_2$  in this definition.

by [AHK12], which reduces the estimation problem to a series matricial and tensorial algebra operations. Let  $\Theta = [\theta_1|\theta_2|\dots|\theta_K] \in \mathbb{R}^{d \times K}$  and  $W = \text{diag}(w) \in \mathbb{R}^{K \times K}$  as the diagonal matrix whose elements are the values in  $w$ . Now, from Eqs. 3.13 and 3.14, we can write  $\beta$  and the  $s$ -th slice of  $\gamma$ , denoted as  $\gamma(\cdot, \cdot, s)$ , as:

$$\beta = \Theta W \Theta^\top, \quad \gamma(\cdot, \cdot, s) = \Theta W^{\frac{1}{2}} \text{diag}(\Theta(s, \cdot)) W^{\frac{1}{2}} \Theta^\top, \quad (3.16)$$

where  $\Theta(s, \cdot) = [\theta_1(s), \dots, \theta_K(s)]$  is the  $s$ -th row of  $\Theta$ . From the above expression, we note that  $\beta$  is positive semidefinite and therefore we can compute its SVD as  $\beta = USU^\top$ . Now, defining  $M = US^{\frac{1}{2}}$  leads to  $\Theta W^{\frac{1}{2}} = MO$  for a unique orthonormal matrix  $O$ . The authors in [AFH<sup>+</sup>12] use the pseudo inverse of  $M$ , here defined as  $M^\dagger$ , as a whitening step on the slices of  $\gamma$ . In their algorithm, this step was used to transform  $\gamma$  into a symmetric orthogonally decomposable tensor whose high order singular vectors could efficiently be estimated via a tensorial analogous to the power method in matrices. The slices of this whitened tensor  $\gamma^{\text{white}}$  now present the following property:

$$\gamma^{\text{white}}(\cdot, \cdot, s) = M^\dagger \gamma(\cdot, \cdot, s) (M^\dagger)^\top = O \text{diag}(\Theta(s, \cdot)) O^\top, \forall s \in \{1, \dots, K\}. \quad (3.17)$$

The above equation tells us that there must be a unique orthonormal matrix  $O$  that diagonalizes all the slices of  $\gamma^{\text{white}}$  simultaneously. For  $K = 2$ , an approximation of  $O$  can be found in constant time by the method proposed by [RRB17] and, for  $K \geq 2$ , the algorithm proposed in [CS96] approximately solves it in polynomial time. Having computed the approximate  $O$ , an approximate solution to  $\Theta$  can be found as  $\Theta = MO$ , followed by the projection of its columns to the simplex of  $K - 1$  dimensions, i.e., a row-wise normalization. Finally, noticing that  $\alpha = \Theta w$  from Eq. 3.12, an approximation of  $w$  can be computed as  $\Theta^\dagger \alpha$  followed by its projection to the simplex of  $K - 1$  dimensions. Throughout this thesis, this approach will be called the tensorial method, for its usage of tensorial operations.

### 3.4.2 Estimation when $r$ is large

Proposition 3 does not strictly hold in practice, except for some specific scenarios, such as when the image's pixel values are randomly drawn IID from a distribution that only depends

on the region they are located. In general, pixels are not expected to satisfy Assumption 2 for small  $r$ . Therefore, further investigation is necessary in order to adapt the tensorial method to this more general case. In the experimental section, we nonetheless show some results that, ideally, would require a large  $r$  in order to have Assumption 2 satisfied.

### 3.5 Examples

Figure 3.6 illustrates some estimation results on real images for all the estimation methods proposed in this work. In these examples we used  $\rho = 0.06$  to select  $r$ . The values of  $w_0$ ,  $w_1$  and  $\epsilon$  were estimated separately for each image by searching over discrete choices as described in the last section. For comparison we also show the appearance models computed using ground truth segmentations. For the case of a ground truth segmentation the appearance models are normalized histograms of the pixel values within each region. We see that both the algebraic and spectral methods give good results in these examples, leading to appearance models that are close to the ground truth. We also note that the tensorial method gives reasonably good estimation results for most image regions considered in Figure 3.6, having its performance comparable to the algebraic and spectral methods, despite the images not necessarily satisfy the conditions in Proposition 3. However, in the one of them ( $\mathcal{R}_0$  of the first image), the algebraic and spectral methods clearly outperform it, showing the impact of difficulty estimating appearances from regions that do not observe the requirements from Proposition 3.

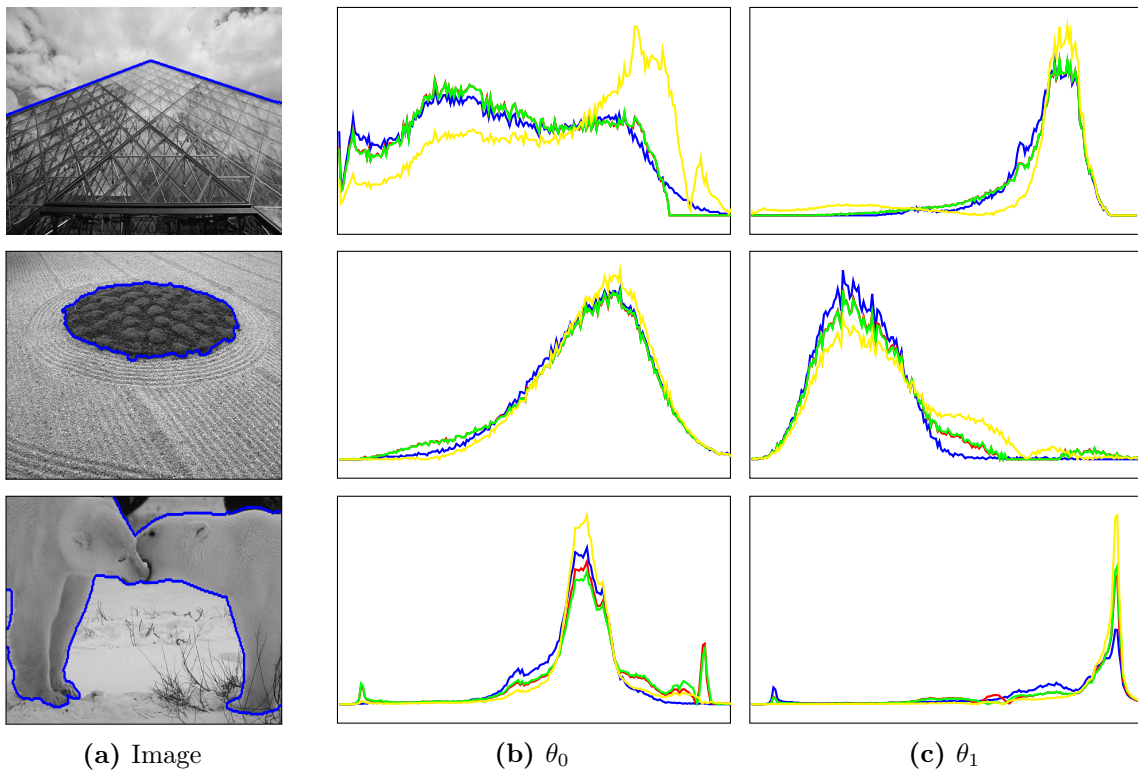
## 3.6 Numerical Experiments

### 3.6.1 Evaluation Measures

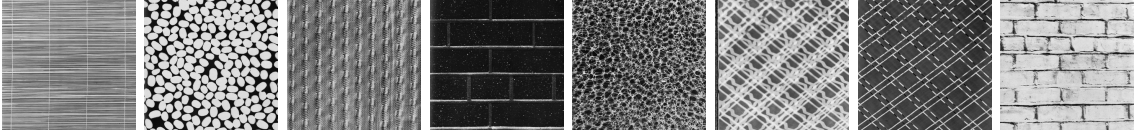
To evaluate the quality of the appearance models we estimate we compare them to the appearance models defined by a ground truth segmentation using the Bhattacharyya distance,

$$d_B(p, q) = -\ln \left( \sum_{i \in L} \sqrt{p(i)q(i)} \right), \quad (3.18)$$

where  $p, q$  are discrete probability distributions over  $L$ .



**Figure 3.6:** Estimation of appearance models with  $\rho = 0.06$ . In (a) we show the input images and their ground truth segmentation. In (b) and (c) we show the appearance models computed using the ground truth segmentation in blue (—), the algebraic method in green (—), the spectral method in red (—) and the tensorial method (—). The images are from the Berkeley Segmentation [MFTM01] dataset.



**Figure 3.7:** Selected Brodatz patterns

Let  $I$  be an image with a ground truth segmentation defined by two regions  $\mathcal{R}_0$  and  $\mathcal{R}_1$ . Let  $\theta_0$  and  $\theta_1$  be the normalized histograms of the pixel values within each region. Let  $\hat{\theta}_0$  and  $\hat{\theta}_1$  be the appearance models estimated from  $I$  using one of our algorithms. We assess the quality of the estimates using a sum of two Bhattacharyya distances, allowing for a permutation of the region labels,

$$D_B = \min \left( \frac{d_B(\theta_0, \hat{\theta}_0) + d_B(\theta_1, \hat{\theta}_1)}{2}, \frac{d_B(\theta_0, \hat{\theta}_1) + d_B(\theta_1, \hat{\theta}_0)}{2} \right). \quad (3.19)$$

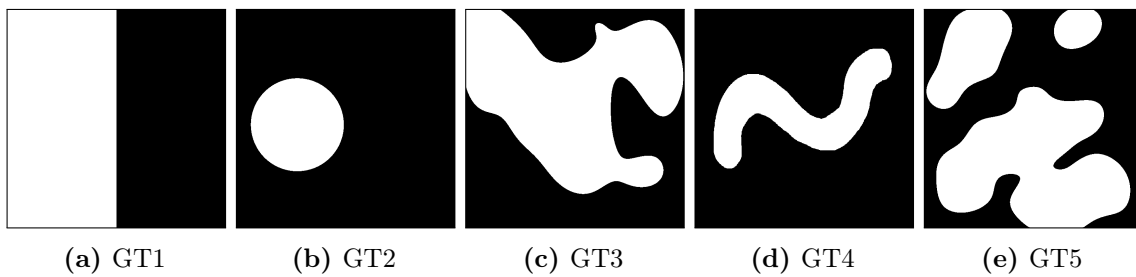
We will also evaluate the accuracy of segmentations obtained using the estimated appearance models by comparing them to the ground truth segmentations. We assess the overlap between two regions  $J, Q \subseteq \Omega$  in different segmentations using the Jaccard index,

$$J(S, Q) = |S \cap Q| / |S \cup Q|. \quad (3.20)$$

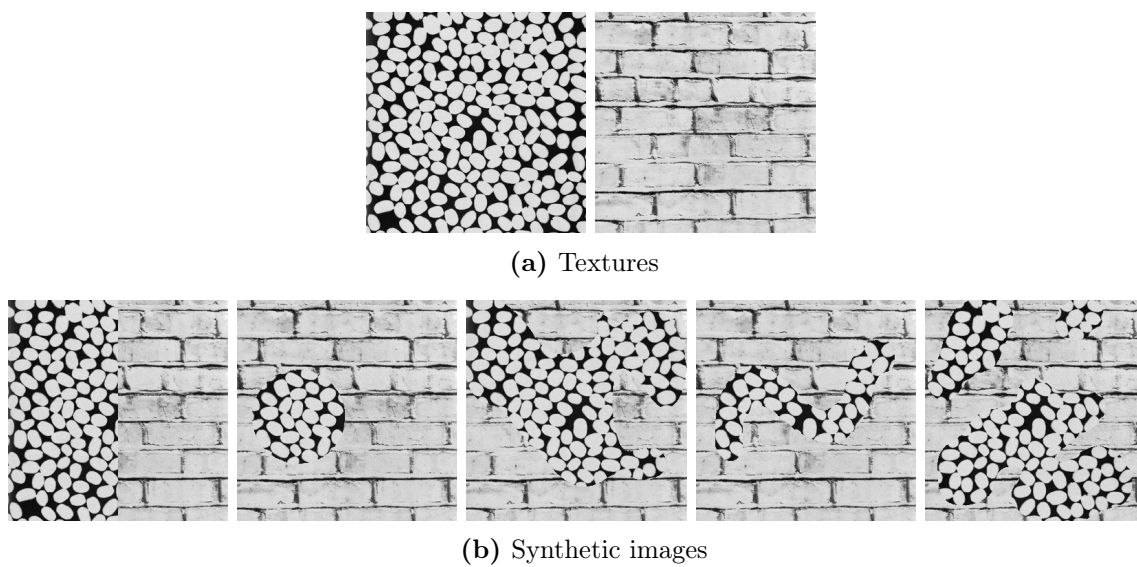
Again let  $I$  be an image with a ground truth segmentation defined by  $\mathcal{R}_0$  and  $\mathcal{R}_1$ . Let  $\mathcal{Q}_0$  and  $\mathcal{Q}_1$  be the two regions obtained by segmenting  $I$  using the estimated appearance models. We compare the two segmentations using a sum of two Jaccard indices, again allowing for a permutation of the region labels,

$$J = \max \left( \frac{J(\mathcal{R}_0, \mathcal{Q}_0) + J(\mathcal{R}_1, \mathcal{Q}_1)}{2}, \frac{J(\mathcal{R}_0, \mathcal{Q}_1) + J(\mathcal{R}_1, \mathcal{Q}_0)}{2} \right). \quad (3.21)$$

All of our algorithms were implemented in Matlab and the experiments presented here were run on a Intel(R) Core(TM) i5-6200U CPU 2.30GHz with 8 Gb of RAM.



**Figure 3.8:** Ground truth segmentations used to generate synthetic data.



**Figure 3.9:** Examples of synthesized images with textures.



### 3.6.2 Synthetic Data

We first illustrate the results of a series of experiments with synthetic data. To generate the synthetic data we used the segmentations masks in Figure 3.8 together with pairs of images defined as follows:

- IID: we used 50 pairs of random appearance models to generate pairs of images. For each appearance model we generate a  $320 \times 320$  image where the pixel values are independent samples from the corresponding distribution.
- Brodatz: we selected all possible pairings of images from the Brodatz textures [Bro66] shown in Figure 3.7. We resized the images to be  $320 \times 320$  pixels and added uniform IID noise to the pixels to to remove quantization artifacts.

For each pair of images defined above we use the segmentation masks in Figure 3.8 to generate graylevel images with two regions. Figure 3.9 shows the images generated using two Brodatz patterns.

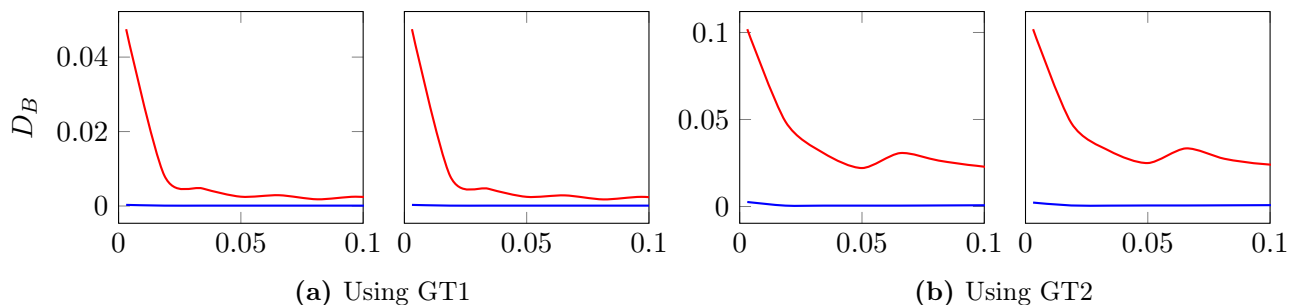
### 3.6.3 Evaluating the effect of $\rho$

In our experiments we set  $r = \rho\sqrt{\Omega}$  where  $\rho$  is a parameter set by the user. This makes the selection of the distance  $r$  be adaptive to the image resolution.

In Figure 3.10 we evaluate the quality of the appearance models estimated by our methods using different values of  $\rho$  on the synthetic data defined by the ground truth segmentations GT1 and GT2 (Figure 3.8). For these experiments we set  $w_0$ ,  $w_1$  and  $\epsilon$  using the ground truth values defined by the corresponding segmentation masks.

Both of our algorithms almost perfectly recover the underlying appearance models for images where the pixels in each region are IID. In this case the methods work well over the whole range of values of  $\rho$  tested. This is expected since these images strictly follow both Assumption 1 and Assumption 2 and, therefore, provide the optimal setting for our algorithms.

For images with Brodatz textures Assumption 2 is violated for small values of  $\rho$ . As  $\rho$  increases the assumption is satisfied and the quality of our estimation improves.



**Figure 3.10:** Average appearance model estimation error ( $D_B$ ) as a function of  $\rho$  on images composed of IID (—) and Brodatz (—) patterns disposed as in GT1 and GT2. For both (a) and (b) the results on the left are from the algebraic method, whereas the results on the right are from the spectral method.

### 3.6.4 Appearance Model Evaluation on Synthetic Images

We compare the performance of our methods to estimate appearance models to a variation of the iterative scheme described in [TAB14], here called ALT.

In ALT, we start with an initial segmentation of the image and alternate between computing new appearance models using the current segmentation and computing a new segmentation using the current appearance models. This procedure is iterated until convergence. To update the appearance models using the current segmentation we histogram the pixel values in each region. We “smooth” the histograms by adding a constant  $K = 1$  to their bins before normalizing them. To update the segmentation using the current appearance models we minimize the energy in Equation (2.12) using a max-flow/min-cut algorithm ([GPS89, BVZ99]).

For the experiments described here the initial segmentation used for ALT is defined by a square region in the middle of the image. Figure 2.5 shows an example of how the segmentation and appearance models evolve over time. Empirically, we found that ALT works well in many examples but a typical failure mode leads to assigning the whole image to single segment.

Table 3.1 compares the results of our methods to the result of ALT using several values of  $\lambda$  for the segmentation step. We used  $\rho = 0.06$  (which corresponds to  $r \approx 20$  pixels for the  $320 \times 320$  synthetic images) for both the algebraic and spectral methods. We evaluated our

**Table 3.1:** Average  $D_B$  distance between estimated and ground truth appearance models on the synthetic data generated using different segmentation masks. We evaluate our algorithms using different methods for selecting  $w_0$ ,  $w_1$  and  $\epsilon$  (see text).

Method		Image Setting										Time (s)
		GT1		GT2		GT3		GT4		GT5		
		IID	Brodatz	IID	Brodatz	IID	Brodatz	IID	Brodatz	IID	Brodatz	
<i>Algebraic</i>	GT $w_0$ and $\epsilon$	0.000	0.003	0.001	0.028	0.000	0.006	0.001	0.030	0.000	0.007	0.23
	$w_0 = 0.5$ and set $\epsilon = 0.5\rho$	0.000	0.003	0.019	0.064	0.001	0.007	0.016	0.058	0.002	0.011	0.14
	Search $w_0$ and $\epsilon$	0.000	0.010	0.002	0.040	0.002	0.017	0.003	0.032	0.003	0.021	7.81
	Alternate $w_0, w_1$ and $\theta_0, \theta_1$ , search $\epsilon_r$	0.000	0.004	0.017	0.047	0.001	0.009	0.017	0.051	0.002	0.013	2.62
	Alternate $w_0, w_1, \epsilon_r$ and $\theta_0, \theta_1$	0.006	0.019	0.002	0.017	0.002	0.022	0.028	0.108	0.001	0.018	0.48
<i>Spectral</i>	GT $w_0$ and $\epsilon$	0.000	0.002	0.001	0.034	0.000	0.006	0.001	0.032	0.000	0.007	0.06
	$w_0 = 0.5$ and set $\epsilon = 0.5\rho$	0.000	0.003	0.019	0.067	0.001	0.007	0.016	0.059	0.002	0.011	0.06
	Search $w_0$ and $\epsilon$	0.000	0.008	0.001	0.034	0.002	0.016	0.003	0.040	0.003	0.020	0.16
	Alternate $w_0, w_1$ and $\theta_0, \theta_1$ , search $\epsilon_r$	0.000	0.003	0.017	0.050	0.001	0.009	0.017	0.052	0.002	0.014	0.09
	Alternate $w_0, w_1, \epsilon_r$ and $\theta_0, \theta_1$	0.006	0.018	0.002	0.015	0.002	0.017	0.028	0.108	0.001	0.016	0.10
ALT	$\lambda = 1$	0.044	0.091	0.000	0.112	0.000	0.079	0.000	0.098	0.000	0.083	3.70
	$\lambda = 3$	0.043	0.019	0.000	0.035	0.000	0.021	0.000	0.033	0.000	0.026	3.26
	$\lambda = 5$	0.043	0.020	0.005	0.014	0.000	0.004	0.032	0.019	0.042	0.016	3.29
	$\lambda = 10$	0.043	0.073	0.031	0.033	0.027	0.008	0.032	0.046	0.042	0.055	3.04

algorithms using five different approaches for selecting  $w_0$ ,  $w_1$  and  $\epsilon$ . In the first approach we set the parameters to the their ground truth values defined by the corresponding segmentation mask. In the second approach we fix the parameters to typical values that work well for many images. In the third approach we search over the parameters explicitly (see Section 3.3.3). In the fourth and fifth approaches, we follow the alternation schemes proposed in Section 3.3.3. All of the approaches lead to good results but searching for the optimal parameters leads to a significant increase in runtime for the algebraic method.

We see that our algorithms perform extremely well on images where the pixel values in each region are IID. The results on images with textures are also good and compare favorably to ALT. This result is compelling in particular because the proposed methods do not rely on an iterative model re-estimation scheme such as in ALT, which makes them faster and independent of initialization. The average runtime of the different methods are shown in the last column of Table 3.1.

### 3.6.5 Segmentation Evaluation on Synthetic Images

After estimating appearance models using either the algebraic or spectral methods we compute segmentations by minimizing Equation (2.12) using a max-flow/min-cut algorithm ([GPS89, BVZ99]). We compared this approach to several texture segmentation methods.

The methods we compare to include Level Set Segmentation using Wasserstein Distances (LSWD) [NBCE09], Images as Occlusions of Textures (ORTSEG) [MMF<sup>+</sup>14] and Factorization Based Segmentation (FBS) [YWC15]. For each of these methods, we used the Matlab implementations provided by the authors. We tuned the parameters of each method to improve their performance in our dataset. We also evaluate the segmentation results obtained with the iterative scheme ALT described above.

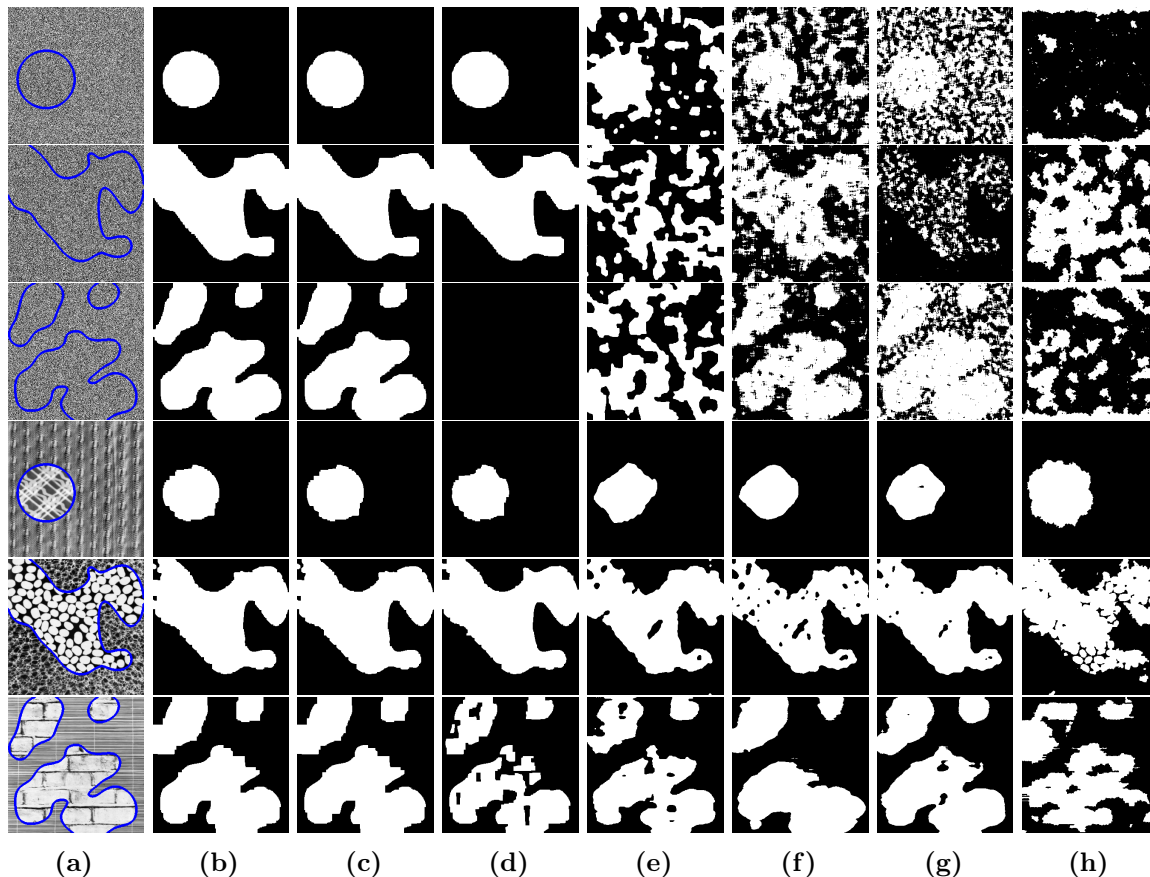
All of the methods we have used for comparison assume either explicitly or implicitly that regions have homogeneous appearance. FBS uses a filter bank to define local features, while LSWD and ORTSEG work with raw pixel values. LSWD, ORTSEG and FBS require the selection of a window size parameter that has a function similar to  $r$  in our methods.

We used  $\rho = 0.06$  to estimate appearance models with our methods. We set  $w_0$ ,  $w_1$  and  $\epsilon$  by searching over the parameters explicitly (see Section 3.3.3). We compute segmentations using several choices for  $\lambda$  in Equation (2.12) and evaluate each choice separately.

Figure 3.11 illustrates some of the segmentations obtained using the different methods for both types of images (IID and Brodatz) used for evaluation. In these examples we used  $\lambda = 5$  to compute segmentations with our methods and in ALT.

Table 3.2 provides a quantitative evaluation on the full set of synthetic images generated using the procedure described in Section 3.6.2. This is the same data used to generate the results in Table 3.1. Notice that the runtime of our methods is increased for the segmentation experiments (Table 3.2) when compared to the model estimation experiments (Table 3.1) due to the addition of the graph cut computation to obtain a segmentation after estimating appearance models.

Table 3.2 demonstrates a clear advantage of our methods under the IID case. For the Brodatz setting, the results demonstrate that our methods provide high quality segmentations without



**Figure 3.11:** Qualitative segmentation results: (a) original image and its ground truth segmentation, (b) algebraic method, (c) spectral method, (d) ALT, (e) LSWD, (f) ORTSEG, (g) FBS, and (h) PNMf.

relying on iterative approaches and filter banks. This makes our methods faster than most of the other approaches, while still leading to accurate results.

These results confirm the soundness and efficacy of the assumptions presented in Section 3.2. They also confirm that it is possible to design high performance texture segmentation algorithms working directly with raw images. As can be seen in Figure 3.11 this leads to segmentations that are accurate near region boundaries, where methods that rely on filter responses often suffer. Finally, although not presented here for the sake of simplicity, our methods could have their segmentation performance further improved when using the estimated appearance as an initial guess for an iterative scheme such as ALT.

**Table 3.2:** Average  $J$  index of different segmentation methods on the synthetic data generated using different segmentation masks.

		Image Setting										
		GT1		GT2		GT3		GT4		GT5		
Method	$\lambda$	IID	Brodatz	IID	Brodatz	IID	Brodatz	IID	Brodatz	IID	Brodatz	Time (s)
<i>Algebraic</i>	3	1.000	0.896	0.991	0.789	0.977	0.869	0.980	0.780	0.956	0.850	8.13
	5	1.000	0.919	0.989	0.814	0.955	0.873	0.966	0.787	0.884	0.841	8.15
	7	1.000	0.937	0.986	0.838	0.919	0.856	0.886	0.782	0.708	0.811	8.22
	10	1.000	0.936	0.976	0.860	0.793	0.837	0.826	0.777	0.570	0.767	8.31
<i>Spectral</i>	3	1.000	0.894	0.991	0.777	0.977	0.863	0.980	0.780	0.958	0.846	0.53
	5	1.000	0.918	0.989	0.805	0.957	0.862	0.965	0.795	0.898	0.841	0.57
	7	1.000	0.934	0.986	0.819	0.918	0.845	0.884	0.783	0.759	0.808	0.64
	10	1.000	0.930	0.974	0.850	0.804	0.824	0.823	0.783	0.590	0.771	0.72
ALT	1	0.467	0.683	0.990	0.591	0.986	0.700	0.984	0.620	0.982	0.694	3.70
	3	0.500	0.874	0.991	0.761	0.981	0.854	0.977	0.757	0.974	0.820	3.26
	5	0.500	0.874	0.858	0.843	0.966	0.904	0.187	0.785	0.500	0.835	3.29
	10	0.500	0.673	0.141	0.784	0.551	0.842	0.177	0.631	0.500	0.668	3.04
LSWD	–	0.936	0.959	0.602	0.737	0.805	0.844	0.576	0.669	0.718	0.776	89.35
ORTSEG	–	0.804	0.935	0.785	0.773	0.761	0.883	0.766	0.762	0.719	0.851	1.53
FBS	–	0.585	0.908	0.582	0.734	0.549	0.842	0.581	0.700	0.547	0.810	0.08

### 3.6.6 Real Images

We also tested the proposed algorithms on real images from a variety of datasets, including the Berkeley Segmentation Dataset [MFTM01], the Plant Seedlings Dataset [GJJ<sup>+</sup>17] and a Scanning Electron Microscope (SEM) dataset [AMCC18]. The images were chosen such that Assumption 1 approximately holds.

Figure 3.13 shows some of the results obtained using our methods for estimating appearance models followed by segmentation using graph cuts. For each image, we used  $\rho = 0.03$  and  $\lambda = 5$ . These results illustrate how the proposed algorithms work well on a variety of different types of images.

For these experiments we added a pre-processing step to our algorithms to reduce the total number of colors in RGB images to a smaller number of quantized values. This is necessary in order to obtain good estimates for  $\alpha$  and  $\beta$ .

To quantize the colors in an RGB image we repeatedly partition the color space until each partition has at most 1000 pixels. Starting from the whole set of pixels, we partition the set into two using a random hyperplane in RGB space going through the center of mass of the set. We recurse this procedure until the stopping criteria is met. The same approach could be used for vector valued images such as hyperspectral images that arise in remote sensing applications.

For a quantitative evaluation of our methods on real imagery, we also tested our methods in two image datasets typically used in Segmentation and Salient Object Detection (SOD):

**Segmentation Evaluation Database (SED1)** : 100 gray-scale images of various sizes from diverse natural scenes each with only one salient object/foreground [AGBB11]. Figure 3.12 displays some images from that dataset.

**Singapore Whole sky Nighttime Image SEGmentation Database (SWINSEG)** : 115 diverse nighttime sky/cloud  $500 \times 500$  color images. The images were captured using a ground-based whole sky imager called Wide Angle High Resolution Sky Imaging System (WAHRISIS), designed and deployed at Nanyang Technological University in Singapore [DSLW17]. Figure 3.12 depicts some sample images found in that database.

Both datasets provide ground truth segmentation data. For these experiments, we adopted the  $F$ -score to evaluate the performance of our method at detecting foreground pixels. The  $F$ -score  $F_\beta$  is defined as:

$$F_\beta = \frac{(1 + \beta^2) \times \text{Precision} \times \text{Recall}}{\beta^2 \times \text{Precision} + \text{Recall}}, \quad (3.22)$$

where we set  $\beta = 0.3$ , when not explicitly specified, and Precision and Recall are defined as:

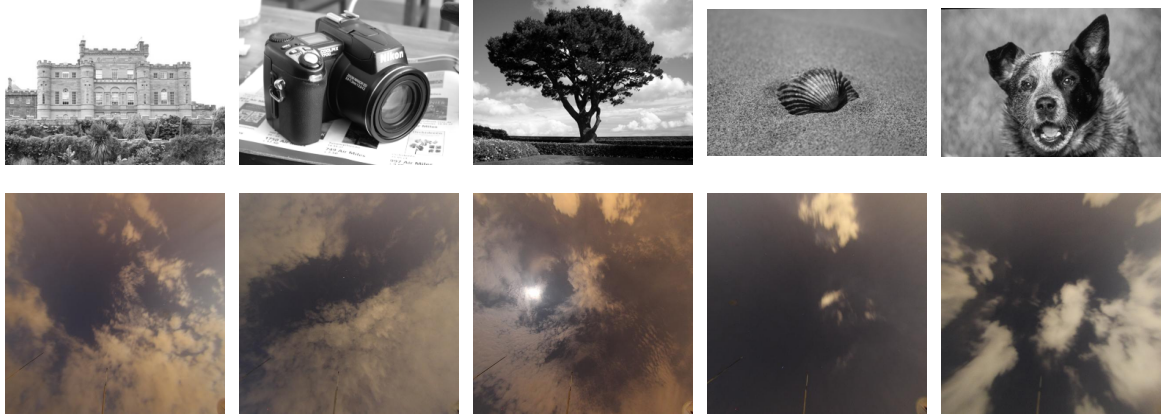
$$\text{Precision} = \frac{Q \cap \mathcal{R}}{Q}, \quad \text{Recall} = \frac{Q \cap \mathcal{R}}{\mathcal{R}}, \quad (3.23)$$

where  $Q$  and  $\mathcal{R}$  are the estimated and ground-truth foreground regions, respectively. As stated in [vR79],  $F_\beta$  “measures the effectiveness of retrieval with respect to a user who attaches  $\beta$  times as much importance to recall as precision”.

For each image in each dataset, we tried all combinations of  $\rho \in \{0.01, 0.02, 0.03, \dots, 0.1\}$  and  $\lambda \in \{0, 2, 4, \dots, 10\}$ , choosing the best result in terms of  $F$ -score as our final segmentation estimation. Note that this parameter selection procedure individualized for each image is also done in some works such as [AGBB11]. For each parameter combination, we also considered the  $F$ -score of the segmentation when the labels are swapped.

In Table 3.3, we show how our methods’ segmentation performances in the images from the SED1 Dataset [AGBB11] and compare them to some of the  $F$ -scores reported in [WS01] for various other methods. In it we show that our performance is comparable to other algorithms that are not based on deep learning architectures, proposed in [ZLWS14] and [YXSJ13]. Table 3.4 shows our performance in the images from the SWINSEG. Here we compared our methods’ performance to the results of other methods reported in [DSLW17]. In fact, we use make use of two different performance metrics,  $F_\beta$  and  $F_1$  in order to make a fair comparison to existing evaluations in the literature. Since many images on the SWINSEG dataset do not satisfy the statistical homogeneity requirement from Assumption 1, our methods are unable to be as performing on this dataset as it is on SED1.





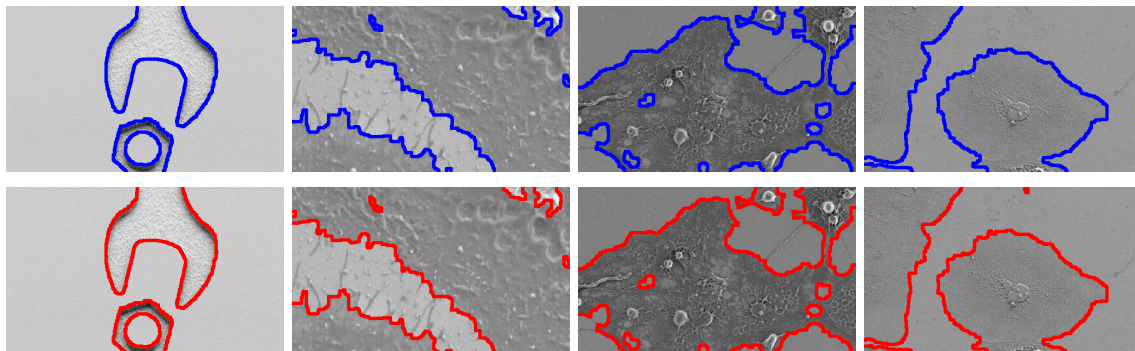
**Figure 3.12:** Sample images from the Weizmann Segmentation Evaluation Database (SED, shown in the first row) [AGBB11] and from the Singapore Whole sky Nighttime Image Segmentation Database (SWINSEG, shown in the second row) [DSLW17].

**Table 3.3:** Comparative Segmentation Performance on the SED1 Database.

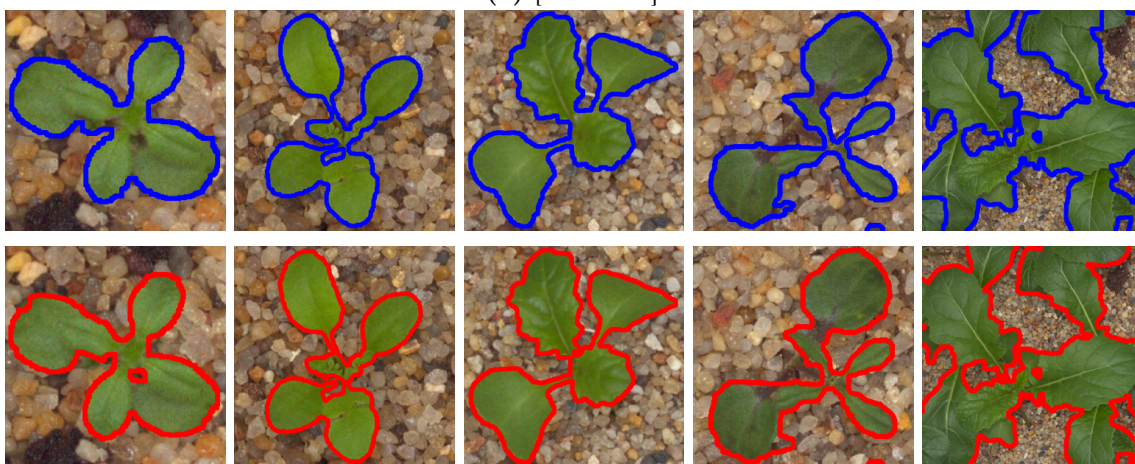
Method	Measure
	$F_\beta$
Algebraic	0.8121
Spectral	0.8056
[WWL+18]	0.8811
[LY16]	0.8546
[LY15]	0.8194
[ZLWS14]	0.7889
[YXSJ13]	0.7426

**Table 3.4:** Comparative Segmentation Performance on the SWINSEG Dataset.

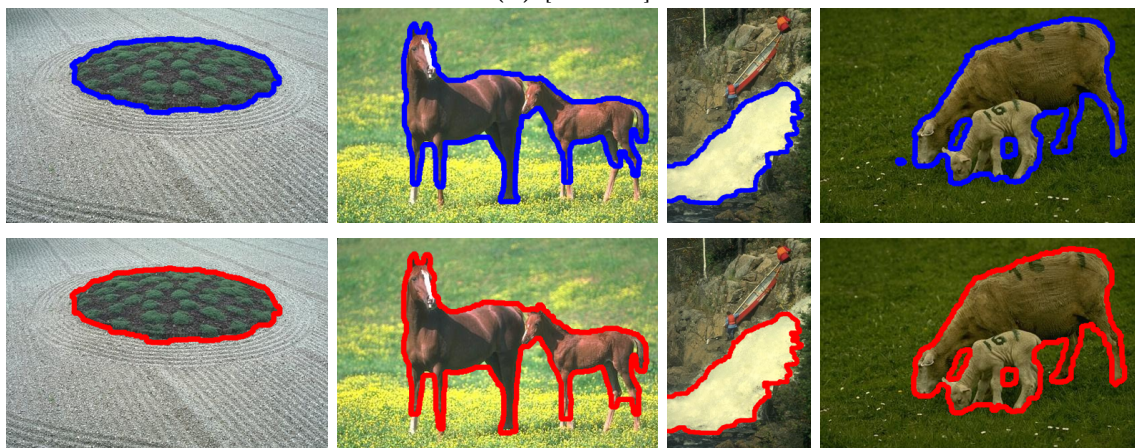
Method	Measure		
	$F_1$	Precision	Recall
Algebraic	0.75	0.95	0.66
Spectral	0.75	0.94	0.66
[YLM+09]	0.79	0.98	0.69
[YLM+10]	0.26	0.90	0.19
[SWCX17]	0.77	0.68	0.93
[GAL16]	0.62	0.47	0.99
[DSLW17]	0.83	0.95	0.76



(a) [AMCC18]



(b) [GJJ+17]



(c) [MFTM01]

**Figure 3.13:** Application our methods in natural scenes. The blue and red contours are the results of segmentation using appearance models estimated using the algebraic and spectral methods, respectively.

### 3.6.7 Experiments using the tensorial method and on multi-region images

For our multi-region experiments, we generate synthetic data in a similar manner as in our experiments on images with two regions. Figure 3.14 shows the ground truth segmentations used here. Note that they comprise ground truths of 3, 4 and 5 regions. The IID data consisted of 50  $320 \times 320$  images whose pixels values are independent samples of 50 random  $K$ -tuples of appearance models,  $K \in \{2, 3, 4, 5\}$ . The Brodatz data was generated by selecting uniformly at random 50 sets of  $K$  different Brodatz patterns shown in Figure 3.7. Figure 3.15 show some examples of Brodatz images generated from the ground truth segmentations in Figure 3.14.

From the estimated appearance models, we computed our estimated segmentations using  $\alpha\beta$  swaps, as explained in 2.1.3. We chose  $\alpha\beta$ -swaps iterations instead  $\alpha$ -expansion for simplicity, since the appearance estimation algorithm is more central to our work.

Here, we also need to generalize our performance metrics to the multi-region setting. For the estimation part, we take the mean of the  $K$  Bhattacharyya distances between our estimated appearances,  $\hat{\theta}_1, \dots, \hat{\theta}_K$ , and the ground truth ones,  $\theta_1, \dots, \theta_K$ , generalizing Eq. 3.19. Since the ordering of these pairs need to match, we go over all permutations of region labels and consider the minimum mean Bhattacharyya distance. Let  $Perm(K)$  be the set of all permutations of the sequence  $\{1, \dots, K\}$ . Our multi-region estimation assessment measure is then defined as:

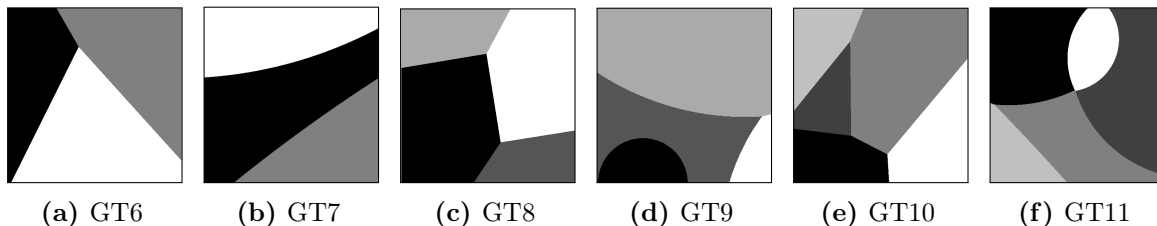
$$\text{Multi}D_B = \min_{\pi \in Perm(K)} \left( \frac{1}{K} \sum_{s=1}^K d_B \left( \theta_s, \hat{\theta}_{\pi(s)} \right) \right). \quad (3.24)$$

In a similar vein, we can generalize the Jaccard measure of segmentation quality in Eq. 3.21 as follows:

$$\text{Multi}J = \max_{\pi \in Perm(K)} \left( \frac{1}{K} \sum_{s=1}^K (J(\mathcal{R}_s, \mathcal{Q}_{\pi(s)})) \right), \quad (3.25)$$

where  $\mathcal{R}_1, \dots, \mathcal{R}_K$  and  $\mathcal{Q}_1, \dots, \mathcal{Q}_K$  are the ground truth and estimated segmentations, respectively.

Table 3.5 presents some quantitative segmentation and estimation results using the above metrics on the generated multi-region synthetic data. Since the tensorial method is also able

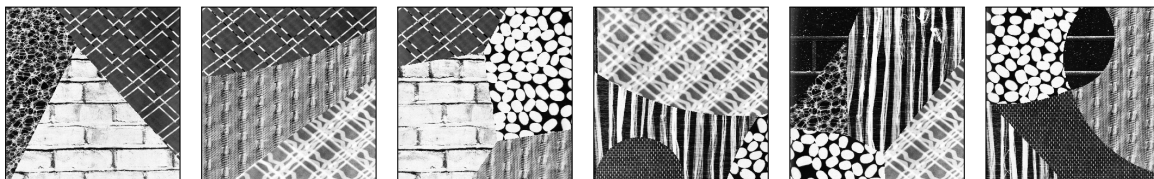


**Figure 3.14:** Ground truth segmentations used to generate the multi-region synthetic data.

to estimate the normalized region sizes in each image, we also computed the Bhattacharyya distance between the ground truth and estimated region weights,  $w$  and  $\hat{w}$ , respectively, according to the region permutation found in Equation 3.24. Additionally, the table also presents the runtime of the estimation and the segmentation parts separately. Following the previous sections, we set  $\lambda = 5$  and  $\rho = 0.06$  in these experiments.

These results demonstrate the effectiveness of using the tensorial method within the IID generation setting. It is able to attain low appearance estimation errors and high Multi  $J$  values, besides also being able to estimate well the normalized region sizes. This performance is expected since these images strictly follow the assumptions in Proposition 3. On the other hand, our proposed method does not perform as well within the Brodatz image framework, also as expected. Interestingly enough, however, it does generate good region size estimates and relatively performing segmentation results, specially for low  $K$ . This suggests that our proposed method could perform better in higher resolution textured images.

Runtime-wise, the table also demonstrates that the estimation algorithm is fast for all tested  $K$ . The overall method's runtime is mainly due to the  $\alpha\beta$ -swap iterations for the segmentation part and increases with  $K$ , which is also expected.



**Figure 3.15:** Examples of synthetic textured images generated by the ground truth segmentation in Figure 3.14.



**Table 3.5:** Average performance measures for estimation and segmentation results on the multi-region synthetic data generated using different ground truth segmentations.

Measure	Image Setting							
	$K = 2$				$K = 3$			
	GT1		GT2		GT6		GT7	
	IID	Brodatz	IID	Brodatz	IID	Brodatz	IID	Brodatz
Multi $D_B$	0.001	1.248	0.816	1.236	0.016	1.268	0.029	1.236
$d_B(w, \hat{w})$	0.000	0.004	0.055	0.006	0.023	0.034	0.006	0.044
Multi $J$	1.000	0.906	0.988	0.904	0.977	0.749	0.985	0.760
Est. Time (s)	0.219	0.134	0.256	0.134	0.191	0.113	0.192	0.113
Seg. Time (s)	0.396	0.366	0.480	0.384	1.375	1.133	1.236	1.145
Measure	$K = 4$				$K = 5$			
	GT8		GT9		GT10		GT11	
	IID	Brodatz	IID	Brodatz	IID	Brodatz	IID	Brodatz
	IID	Brodatz	IID	Brodatz	IID	Brodatz	IID	Brodatz
Multi $D_B$	0.060	1.221	0.014	1.233	0.041	1.270	0.079	1.289
$d_B(w, \hat{w})$	0.058	0.062	0.215	0.136	0.101	0.079	0.004	0.057
Multi $J$	0.984	0.630	0.923	0.623	0.979	0.620	0.974	0.629
Est. Time (s)	0.196	0.153	0.197	0.123	0.202	0.126	0.205	0.125
Seg. Time (s)	2.565	0.563	2.512	2.523	4.296	3.663	4.001	3.515

### 3.7 Conclusion

In this chapter we addressed the problem of estimating appearance models for images with multiple regions. We showed that this can be accomplished without explicit reasoning about the set of pixels in each region. Instead, by assuming homogeneity of appearance within each region and independence at a distance we derived three methods for estimating the appearance of the unknown regions in an image. Our first two algorithms combine the distribution of pixel values within the whole image with the distribution of pairs of nearby values to obtain algebraic expressions that can be used to solve for appearance models in images with two regions. For our third approach, we also considered the distribution of triplets of nearby pixels and showed how to apply it to appearance model estimation in multi-region images. Our experiments demonstrate the proposed methods work well in a variety of settings and the resulting appearance models can be effectively used for segmentation of textured images. These results also suggest that segmentation algorithms can be improved by making use of second and third order pixel statistics. This study, however, lacks a throughout evaluation of tensorial method for different parameters and image generation settings, besides its comparisons to other multi-region segmentation algorithms. These investigations are left to future work.

# CHAPTER 4

## Spectral Image Segmentation with Global Appearance Modeling

In this chapter, we introduce a new spectral method for image segmentation that incorporates long range relationships for global appearance modeling. The approach combines two different graphs, one is a sparse graph that captures spatial relationships between nearby pixels and another is a dense graph that captures pairwise similarity between *all pairs of pixels*. We extend the spectral method for Normalized Cuts to this setting by combining the transition matrices of Markov chains associated with each graph. We also derive an efficient method that uses importance sampling for sparsifying the dense graph of appearance relationships. This leads to a practical algorithm for segmenting high-resolution images. The resulting method can segment challenging images without any filtering or pre-processing.

### 4.1 Drawbacks of the Traditional Graph Construction

The classical similarity criteria in spectral segmentation algorithms [SM00, MVM11] takes into account both the appearance similarity and distance between pairs pixels. This is expected, since in real segmentations nearby pixels that are similar in their appearance tend to belong to the same regions. A typical choice of pixel similarity function is shown in Eq. 2.39, recalled here:

$$w(i, j) = \exp\left(-\frac{\|I(i) - I(j)\|^2}{2\sigma_I^2}\right) \exp\left(-\frac{\|X(i) - X(j)\|^2}{2\sigma_X^2}\right). \quad (2.39)$$

This formulation, however, poses some practical and theoretical difficulties:

**Intepretability** Although this measure is *locally* intuitive, i.e. one expects pixels that are close enough and have similar enough appearances to be clustered together, its *global* interpretation in the context of the cut is less obvious. In other words, it is not straightforward to comprehend what is being minimized when one computes the minimum normalized cut on a graph constructed based on Eq. 2.39.

**Short Range and necessity of filtering techniques** This construction is essentially local in both spacial and color domains, since it assigns large weights to pairs of pixels that are spatially close and whose colors are similar. Consequently, (1) far away pixels that are similar appearance-wise and (2) nearby pixels with difference appearances are both assigned low weights in  $G$ . Due to (1), this construction ends up being unsuitable to deal with textured images, considering they present pixels whose intensities are correlated even at a distance. The same can be said about noisy images, an example of (2). A typical resolution to this issue is the use of filtering techniques [SM00], which results in extra computation overhead and boundary smoothing, making the overall segmentation algorithm less performing and slower.

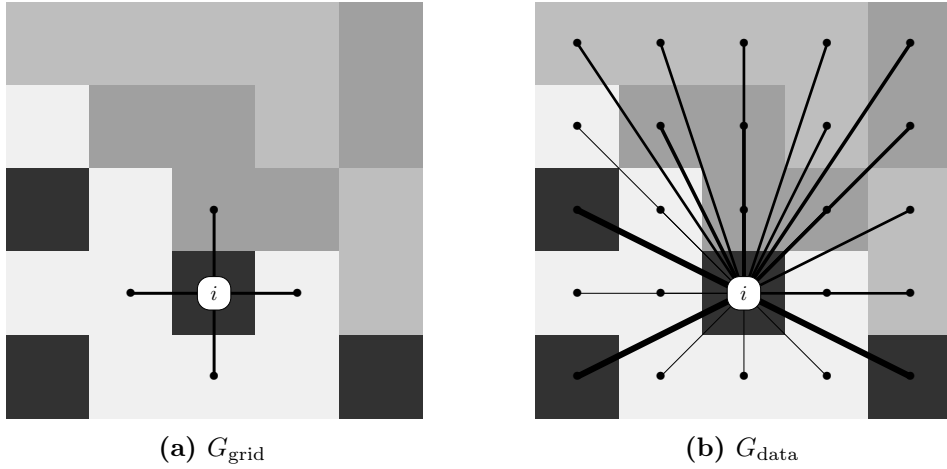
**Implementation** Efficiently constructing this graph is relatively challenging in practice for large values of  $\sigma_I$  and  $\sigma_X$ , even for small images, as discussed in Section 2.3.4. For example, a typical solution for large  $\sigma_X$ , which effectively enlarges the range of  $w$ , relies on image pre-processing using super-pixels in order to overcome this computational problem.

## 4.2 New Criteria for Image Segmentation

In order to address the above difficulties, we combine two normalized cut values to obtain a new criteria for image segmentation. We break the grouping cues (spatial proximity and appearance similarity) into two separate graphs,  $G_{\text{grid}}$  and  $G_{\text{data}}$ . Both graphs are defined over the same set of vertices, corresponding to the pixels in an image.

1. The graph  $G_{\text{grid}}$  is a grid over the image pixels, where each pixel is connected to the four neighboring pixels with an edge of weight 1. This graph encourages neighboring pixels to be grouped together, independent of their appearance.





**Figure 4.1:** The edges connecting to a pixel  $i$  in  $G_{\text{grid}}$  and  $G_{\text{data}}$ . In the each image, the thickness of each link is proportional to its weight.

2. The graph  $G_{\text{data}}$  is a fully connected graph that encourages pixels with similar appearance to be grouped together, independent of their location. The weights in  $G_{\text{data}}$  are based on appearance similarity of pixels, and do not depend on pixel locations,

$$w(i, j) = \exp\left(-\frac{\|I(i) - I(j)\|^2}{2\sigma^2}\right). \quad (4.1)$$

Figure 4.1 illustrates the graph construction of both  $G_{\text{grid}}$  and  $G_{\text{data}}$  on a image with two regions. Below we discuss how the normalized cut criteria for each of the two graphs can be interpreted.

#### 4.2.1 Spatial Information: $G_{\text{grid}}$

Let  $(A, B)$  be a cut in the grid graph. The cut defines a segmentation of the image into two regions, with a boundary  $\Gamma$  between them. The cut value,  $\text{Cut}(A, B|G_{\text{grid}})$ , counts the number of neighboring pixels that are in different regions. In general the cut value in the grid graph and similar graphs can be seen as a measure of the length of the boundary  $\Gamma$  (see [BK03]).

**Observation 1.**

$$\text{Cut}(A, B|G_{\text{grid}}) \approx \text{Len}(\Gamma).$$

This is a commonly used measure of spatial coherence in image segmentation problems (see, e.g., [MS89a]). Although the criteria  $\text{Cut}(A, B|G_{\text{grid}})$  leads to spatially coherent segmentations and is widely used in practice, it gives most preference to trivial solutions with a small (single pixel) region.

Using the previous observation and noting that  $\text{Vol}(S|G_{\text{grid}}) \approx 4|S|$  for  $S \subseteq V$  we can derive an expression for the value of a normalized cut in the grid graph.

**Observation 2.**

$$\text{NCut}(A, B|G_{\text{grid}}) \approx \frac{|V| \text{Len}(\Gamma)}{4 |A||B|}.$$

Minimizing this criteria encourages solutions where the boundary  $\Gamma$  between the two regions is short (to minimize  $\text{Len}(\Gamma)$ ) and where the two regions have similar size (to maximize  $|A||B|$ ). This criteria encourages spatially coherent regions without promoting trivial solutions where one region is very small.

#### 4.2.2 Global Appearance Information: $G_{\text{data}}$

Now we consider the weight of cuts and normalized cuts in  $G_{\text{data}}$ . For  $S \subseteq V$  we use  $g_S$  to denote a kernel density estimate defined by the pixel values in  $S$ ,

$$g_S(c) = \frac{1}{|S|} \sum_{i \in S} K(I(i) - c). \quad (4.2)$$

**Proposition 4.**

$$\text{Cut}(A, B|G_{\text{data}}) = (2\pi\sigma^2)^{\frac{d}{2}} |A||B| \langle g_A, g_B \rangle,$$

where  $d$  is the dimension of the pixel appearance vectors,  $g_A$  and  $g_B$  are density estimates defined using a Gaussian kernel, and  $\langle \cdot, \cdot \rangle$  denotes the standard inner product of functions.

*Proof.* We use the fact that the convolution of two Gaussians with equal variance is a

Gaussian with twice the variance,

$$\begin{aligned}
\sum_{i \in A, j \in B} w_{\text{data}}(i, j) &= \sum_{i \in A} \sum_{j \in B} \exp\left(-\frac{\|I(i) - I(j)\|^2}{2\sigma^2}\right) \\
&= (2\pi\sigma^2)^{\frac{d}{2}} \sum_{i \in A} \sum_{j \in B} \int_{-\infty}^{\infty} \left(\frac{1}{\pi\sigma^2}\right)^d \exp\left(-\frac{\|I(i) - c\|^2}{\sigma^2}\right) \exp\left(-\frac{\|I(j) - c\|^2}{\sigma^2}\right) dc \\
&= (2\pi\sigma^2)^{\frac{d}{2}} \int_{-\infty}^{\infty} \left(\sum_{i \in A} K_{\sigma^2}(I(i) - c)\right) \left(\sum_{j \in B} K_{\sigma^2}(I(j) - c)\right) dc \\
&= (2\pi\sigma^2)^{\frac{d}{2}} |A||B| \int_{-\infty}^{\infty} g_A(c)g_B(c)dc = (2\pi\sigma^2)^{\frac{d}{2}} |A||B| \langle g_A, g_B \rangle, \tag{4.3}
\end{aligned}$$

Here  $K_{\beta}(\cdot)$  is a Parzen window of bandwidth  $\beta$ . □

The proposition above is related to the Laplacian PDF Distance in [JEPE05]. It is also related to the work in [TGV13] where a different graph construction is used to define global appearance models.

The weight of a cut in  $G_{\text{data}}$  will be minimized when the pixel values in the two regions have complementary support. Although this intuitively makes sense, the measure encourages regions to be unbalanced in size due to the term  $|A||B|$  multiplying  $\langle g_A, g_B \rangle$ .

In order to derive an expression for  $\text{NCut}(A, B|G_{\text{data}})$ , we first use a similar reasoning as in the proposition above to note that  $\text{Vol}(S|G_{\text{data}}) = (2\pi\sigma^2)^{(d/2)} |S||V| \langle g_S, g_V \rangle$ . Then, from the definition of the normalized cut we obtain the following result.

**Proposition 5.**

$$\text{NCut}(A, B|G_{\text{data}}) = \langle g_V, g_V \rangle \frac{\langle g_A, g_B \rangle}{\langle g_A, g_V \rangle \langle g_B, g_V \rangle}.$$

This criteria is minimized when (1) the distributions  $g_A$  and  $g_B$  have little overlap and (2) both have significant overlap with  $g_V$ . In particular it penalizes solutions where one region does not represent a significant amount of the image color data. Both behaviors are expected in realistic segmentations: (1) different regions in a scene tend to have different appearances and (2) the appearance of significant/large regions should heavily contribute the overall color distribution of the entire scene.

### 4.2.3 Combining Spatial and Appearance information

The normalized cut values in  $G_{\text{grid}}$  and  $G_{\text{data}}$  provide complementary measures for image segmentation. To combine the spatial and appearance cues we use a convex combination,

$$\text{MixNCut}(A, B) = (1 - \lambda) \text{NCut}(A, B|G_{\text{data}}) + \lambda \text{NCut}(A, B|G_{\text{grid}}). \quad (4.4)$$

The parameter  $\lambda \in [0, 1]$  controls the relative importance of the two normalized cut measures.

We interpret  $\text{MixNCut}(A, B)$  as a mixture of an appearance and a spatial term,

$$\text{MixNCut}(A, B) \approx (1 - \lambda) \left( \langle g_V, g_V \rangle \frac{\langle g_A, g_B \rangle}{\langle g_A, g_V \rangle \langle g_B, g_V \rangle} \right) + \lambda \left( \frac{|V| \text{Len}(\Gamma)}{4 |A| |B|} \right). \quad (4.5)$$

The first term encourages a partition of the image into regions with dissimilar color distributions, while the second term encourages a spatially coherent partition. Both terms are normalized and avoid biases towards solutions with small regions. Note that each term is normalized in a particular way that is natural and has appropriate dimensions for the individual measures.

## 4.3 Segmentation Algorithm

### 4.3.1 Spectral Method

In this section, we provide a heuristic to approximately find the cut  $(A, B)$  that minimizes  $\text{MixNCut}(A, B)$ . The main goal here is to make use the random walk interpretation of normalized cuts presented in Section 2.3.3 in problems involving multiple graphs.

Let  $G_1$  and  $G_2$  be two weighted graphs. Now we describe a spectral method for optimizing a convex combination of two normalized cut values,

$$\text{MixNCut}(A, B|G_1, G_2) = (1 - \lambda) \text{NCut}(A, B|G_1) + \lambda \text{NCut}(A, B|G_2). \quad (4.6)$$

The approach is based on the Markov chain and conductance interpretation of normalized cuts ([SM00, MS01]). Let  $W_1$  and  $W_2$  be the weighted adjacency matrices of the two graphs

while  $D_1$  and  $D_2$  are the diagonal degree matrices. Let,

$$P_1 = D_1^{-1}W_1, \quad P_2 = D_2^{-1}W_2, \quad P = (1 - \lambda)P_1 + \lambda P_2. \quad (4.7)$$

The matrices  $P_1$  and  $P_2$  define two Markov chains on  $V$ . The matrix  $P$  also defines a Markov chain on  $V$  where in one step we follow  $P_1$  with probability  $(1 - \lambda)$  and  $P_2$  with probability  $\lambda$ .

We compute the second largest eigenvector of  $P$  to find a cut  $(A, B)$  with small conductance. In our experiments, we use a Lanczos Process for the truncated eigenvector decomposition. We use  $k$ -means with  $k = 2$  to cluster the entries in the eigenvector into 2 clusters.

Finally, instead of combining two Markov chains, one could also compute a convex combination of the Laplacians of  $G_{\text{grid}}$  and  $G_{\text{data}}$  and proceed with the normalized cut of that new combined graph. Note however that the desired interpretable *per graph* normalization would not be present in that case. Furthermore, as our experiments can attest, combining the Markov chains produces better empirical segmentation results than the using the convex combination of Laplacians.

### 4.3.2 Graph Sparsification

When the matrix  $P$  is sparse we can compute the required eigenvector much more quickly. The grid  $G_{\text{grid}}$  is sparse but  $G_{\text{data}}$  is dense. We sparsify the graph using a random sampling approach.

The approach described here is complementary to other methods that have been used to speed up the computation of eigenvectors for clustering. One such method is based on Nystrom approximation [FBCM04]. Another approach involves power iteration [LC10].

Let  $G$  be a weighted graph. To construct a sparse graph  $G'$  we independently sample  $m = O(|V|)$  edges (with replacement) from  $G$ , with probabilities proportional to the edge weights. The weight of each sampled edge is set to 1 (adding up weights if there is repetition). With this approach the expected value of a cut  $(A, B)$  in  $G'$  equals the value of the cut in  $G$  up to a scaling factor of  $(m/\text{Vol}(V|G))$ . Moreover, if  $m$  is sufficiently large then with high

probability every cut in  $G'$  has weight close to the cut value in  $G$  (up to a scaling factor of  $(m/\text{Vol}(V|G))$ ) (see, e.g., [WS11]). In the experimental section, we parametrize  $m = \alpha|V|$  with  $\alpha > 0$ .

To implement this approach efficiently for  $G_{\text{data}}$  we need to sample edges with probability proportional to their weights  $w(i, j)$  *without* enumerating all possible edges. We use an importance sampling method as a practical alternative.

First, partition  $V$  into  $L$  ( $\approx 1000$  in practice) sets  $S_1, \dots, S_L$  with low appearance variance. We do this greedily, starting with a single set and repeatedly partitioning the set with highest variance into two using the  $k$ -means algorithm. Let  $m_i$  be the mean appearance of pixels in  $S_i$  and

$$q(a, b) = |S_a||S_b| \exp\left(-\frac{\|m_a - m_b\|^2}{2\sigma^2}\right). \quad (4.8)$$

To sample an edge for  $G'$  first select a random pair  $S_a$  and  $S_b$  with probability proportional to  $q(a, b)$ . Then select  $i \in S_a$  and  $j \in S_b$  uniformly at random. Let  $Z_q = \sum_{a', b' \in \{1, \dots, L\}} q(a', b')$ . Finally, add the edge  $\{i, j\}$  to  $G'$  with weight  $w'(i, j) = Z_q|S_a||S_b|w(i, j)/(mq(a, b))$ . This weighting ensures the expected value of a cut in  $G'$  equals the value of the same cut in  $G$ :

$$\begin{aligned} & \mathbb{E}[\text{Cut}(A, B|G')] \\ &= \mathbb{E}\left[\sum_{i \in A} \sum_{j \in B} w'(i, j)\right] \\ &= \mathbb{E}\left[\sum_{i \in A} \sum_{j \in B} \sum_{k=1}^m \frac{Z_q|S_a||S_b|w(i, j)}{mq(a, b)} \mathbb{1}[\{S_a, S_b\} \text{ and } \{i, j\} \text{ selected for the } k\text{-th edge}]\right] \\ &= \sum_{i \in A} \sum_{j \in B} \frac{Z_q|S_a||S_b|w(i, j)}{mq(a, b)} \mathbb{E}\left[\sum_{k=1}^m \mathbb{1}[\{S_a, S_b\} \text{ and } \{i, j\} \text{ selected for the } k\text{-th edge}]\right] \\ &= \sum_{i \in A} \sum_{j \in B} \frac{Z_q|S_a||S_b|w(i, j)}{q(a, b)} \mathbb{P}[\{S_a, S_b\} \text{ and } \{i, j\} \text{ selected}] \\ &= \sum_{i \in A} \sum_{j \in B} \frac{Z_q|S_a||S_b|w(i, j)}{q(a, b)} \mathbb{P}[i \text{ selected}|S_a] \mathbb{P}[j \text{ selected}|S_b] \mathbb{P}[\{S_a, S_b\} \text{ selected}] \\ &= \sum_{i \in A} \sum_{j \in B} \frac{Z_q|S_a||S_b|w(i, j)}{q(a, b)} \frac{1}{|S_a|} \frac{1}{|S_b|} \frac{q(a, b)}{Z_q} \\ &= \sum_{i \in A} \sum_{j \in B} w(i, j) = \text{Cut}(A, B|G) \end{aligned}$$

## 4.4 Numerical Experiments

### 4.4.1 Segmentation accuracy measure and hardware setting

To measure the accuracy of a segmentation we use the Jaccard Index, defined in Section 3.6.1. The algorithms in this section were implemented in MATLAB and run on the same hardware specifications of Section 3.6.

### 4.4.2 Sparsification algorithm for NCut

We compare our new segmentation method with the original normalized cut formulations, NCut, using the graph construction defined in Section 2.3.2. We sparsify this graph to scale the eigenvector computation to large images. Again, we accomplish this using importance sampling. Let  $H$  be the graph with weights defined by Equation 2.39. To sample one edge from  $H$ , first select a pixel  $i$  uniformly at random. Then, draw a location  $x$  from a Normal distribution centered at  $X(i)$  with variance  $\sigma_X^2$  and select the pixel  $j$  closest to that location. We add the edge  $\{i, j\}$  to  $G'$  with weight  $w'(i, j) = \exp(-\|I(i) - I(j)\|^2 / 2\sigma_I^2)$ . We repeat this process  $m$  times. Again, this choice of  $w'$  ensures the expected value of a cut in  $G'$  equals the value of the same cut in  $H$ . In practice we empirically noticed that setting  $m = 100|V|$  produces the best segmentation results, while keeping a low memory/computation cost.

### 4.4.3 Evaluation of NCut and MixNCut without sparsification

In order to demonstrate the efficacy of our multiview graphical approach compared to the traditional graph construction given in Eq. 2.39, we run both NCut and MixNCut without the graph sparsification step of each algorithm. Figure 4.2 shows the segmentation performance of each method under various combinations of their parameters. The tested images were selected such that they have different region sizes and textural appearances.

These results demonstrate the effect of low values for either  $\sigma_I$  or  $\sigma_X$  for the normalized cut method. In this setting, either the first or the second term in Eq. 2.39 is very small, potentially equalling zero, numerically. Therefore, the graph  $H$  is very sparse, which hinders the algorithm's segmentation performance. On the other hand, when both parameters increase past a threshold, the segmentation results abruptly and substantially improve,

attaining an approximately constant Jaccard value for any remaining  $(\sigma_I, \sigma_X)$  pair. At that point, the effect of either parameter on the final result becomes less clear. Furthermore, the algorithm heavily lessens its performance in textured images, notably when the regions sizes are unbalanced.

Our method is able to achieve good results in all scenarios, including the textured and/or unbalanced images. We also note that varying the parameters  $\sigma$  and  $\lambda$  produce clear implications on the final segmentation result. In other words, differently from NCut, the variation in each parameter also generates a smooth variation in our algorithm's performance, which is very helpful when tuning them. On the practical side, these results also suggest that our method performs well in the regime of high  $\lambda$  (between 0.8 and 0.97) and low  $\sigma$ .

#### 4.4.4 Impact of edge sampling on MixNCut

Figure 4.3 shows how our algorithm performs under different values of  $m = \alpha|V|$ , i.e. number of sampled edges of  $G_{\text{data}}$ , when sparsified according to the algorithm in Section 4.3.2. Our method attains satisfactory results for  $\alpha \geq 2$  and achieves almost perfect segmentations for values above  $\alpha = 4$ . Furthermore, it has the lowest processing time<sup>1</sup> around  $\alpha = 2$ . Having that in mind, the number of sampled edges used to sparsify  $G_{\text{data}}$  was set to  $m = 2|V|$  for the experiments with MixNCut in the following sections.

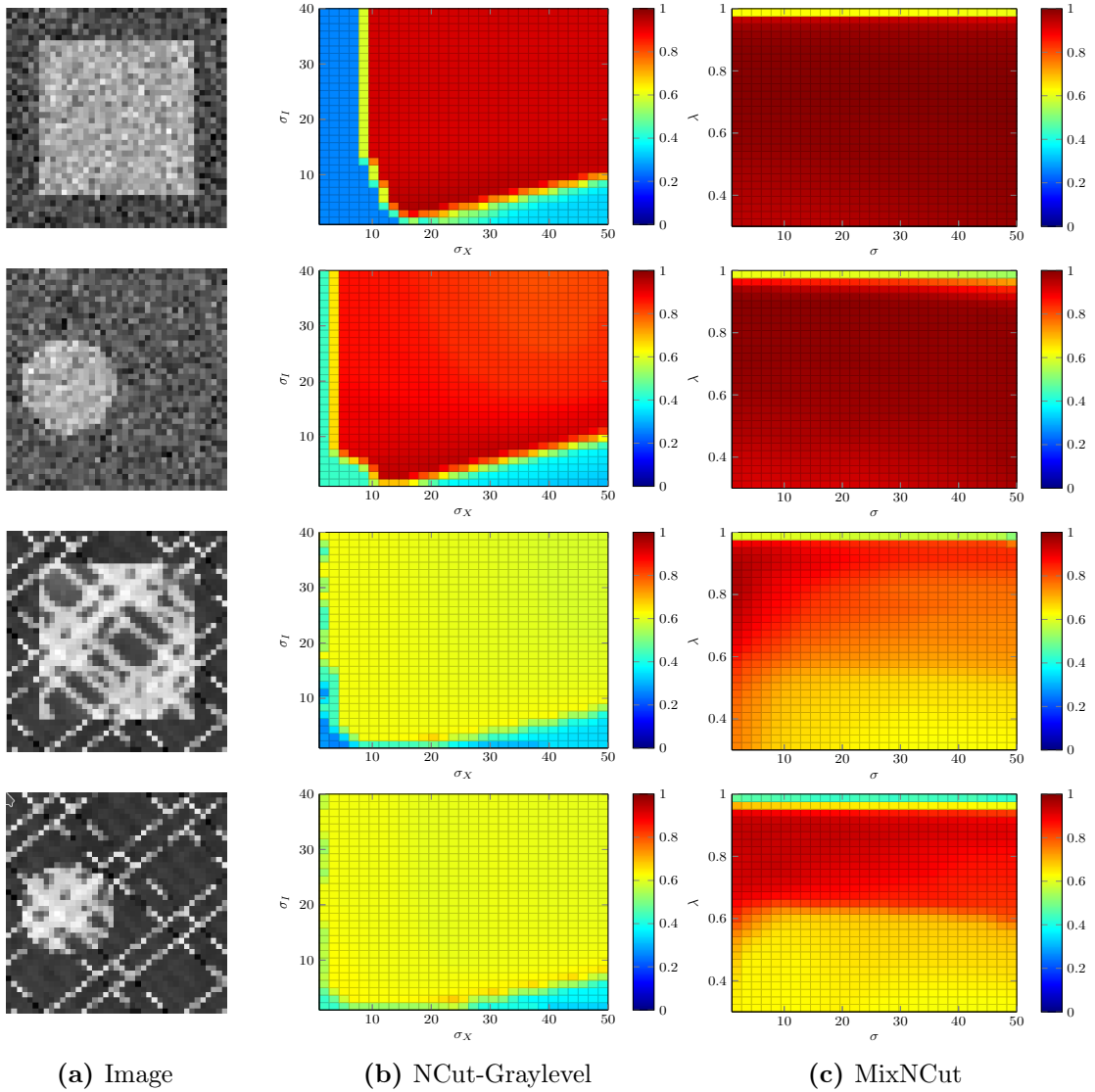
#### 4.4.5 The role of $\lambda$ on MixNCut

Figure 4.4 depicts the visual impact of varying  $\lambda$  in our method. For low  $\lambda$ 's, MixCut outputs a segmentation where fine image structures are kept while preserving some image artifacts. As it increases, the computed eigenvector gets blurred, which causes the final segmentation to have smoother boundaries. As already mentioned in the text, these results demonstrates the role that  $\lambda$ , and therefore  $G_{\text{grid}}$ , plays as enforcing consistency within each region, whereas  $G_{\text{data}}$  promotes the fitting of the color data in each region.

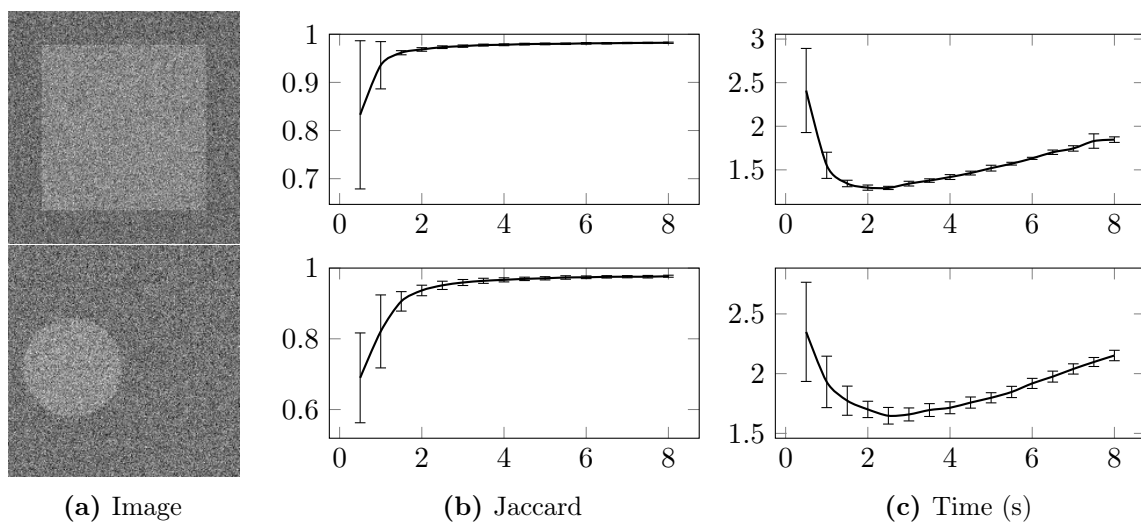
---

<sup>1</sup>The increase in computation time for values of  $\alpha$  smaller than 2 is due to the slow convergence of the Lanczos algorithm in that regime.

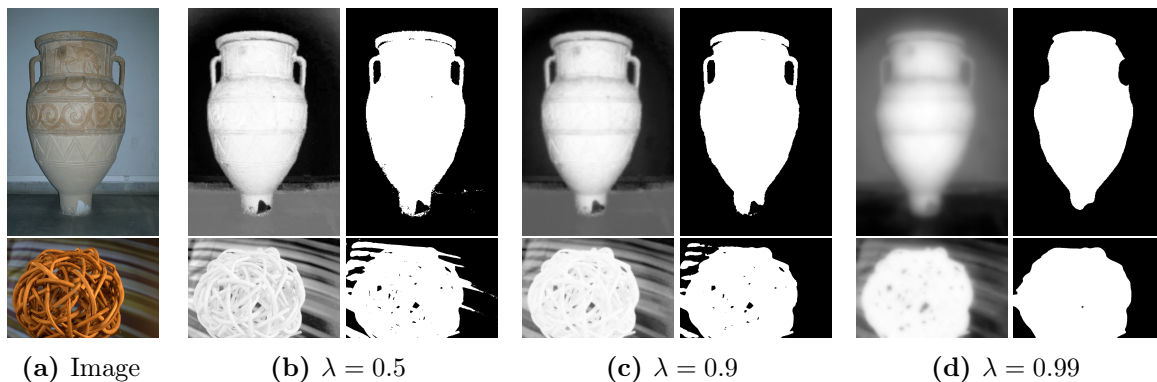




**Figure 4.2:** Evaluation of NCut-Graylevel and MixNCut performances without sampling. Column (a) show the test images of size  $40 \times 40$ , with different levels of noise. In column (c) we show the value of  $J$  for a range of values for  $\sigma_I$  and  $\sigma_X$ . In (c), we show evaluate the segmentation performance of MixNCut over various combinations of  $\sigma$  and  $\lambda$ . In these experiments NCut averaged  $6.31 \pm 5.28$ s of processing time and MixCut,  $1.35 \pm 0.25$ s.



**Figure 4.3:** Evaluation of sampling performance for various  $\alpha$ . Column (a) shows two different test images of size  $200 \times 200$ . Column (b) shows the average and standard deviation of  $J$  segmentation measure for 100 runs of MixCut on the respective test image. Column (c) shows the average time and standard deviation of the same runs. Here, we set  $\lambda = 0.95$  and  $\sigma = 1$ .



**Figure 4.4:** MixNCut results when varying  $\lambda$ . Column (a) shows the input image. Columns (b)-(d) show the computed eigenvectors (on the left) and segmentations (on the right) given by MixNCut for various values of  $\lambda$  and  $\sigma = 1$ .

#### 4.4.6 Experiments in Real Images

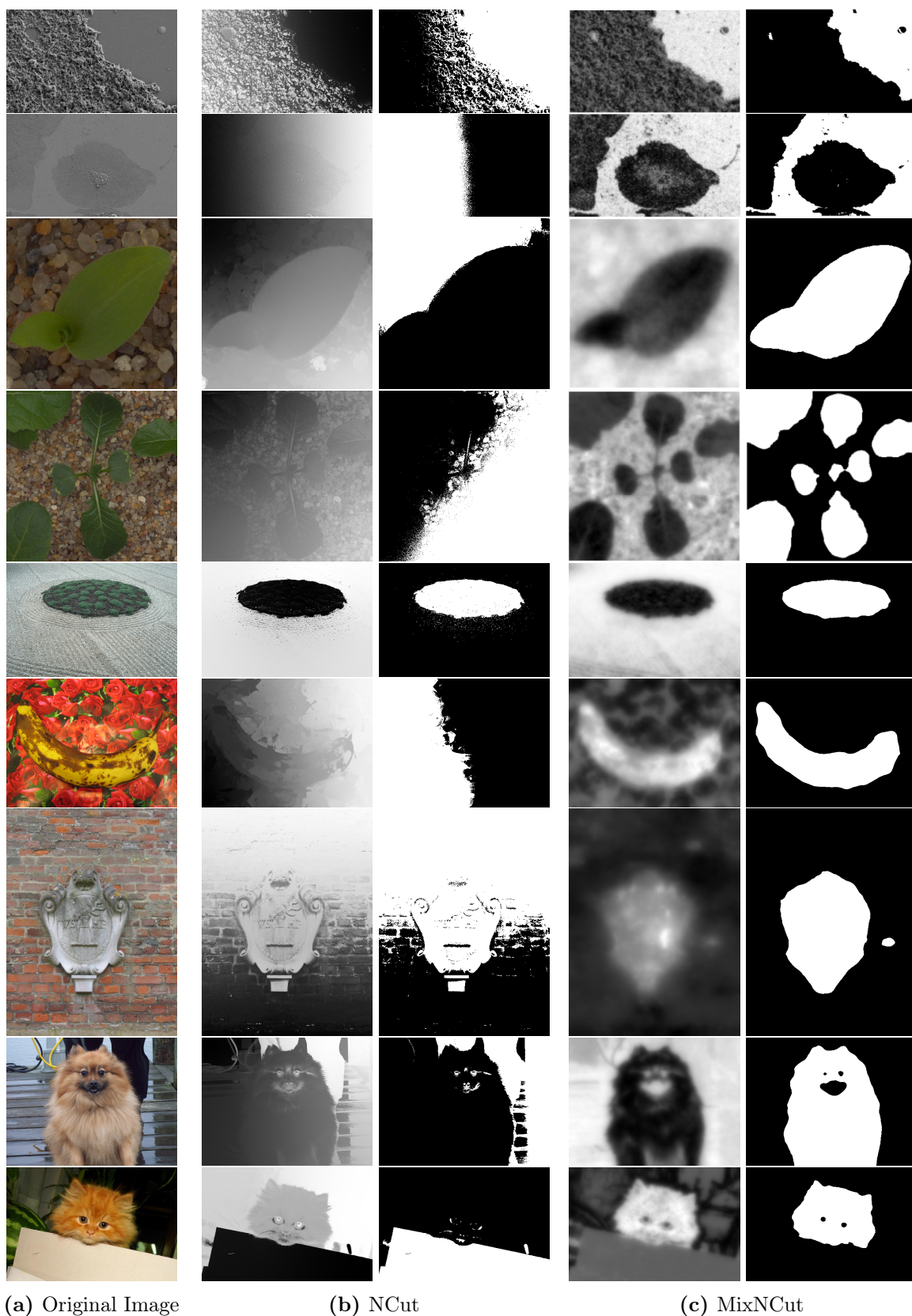
We tested our method on real images from a variety of datasets including the Berkeley Segmentation Dataset [MFTM01], the Plant Seedlings Dataset [GJJ<sup>+</sup>17], the Grabcut dataset [RKB04], the PASCAL VOC dataset [EEVG<sup>+</sup>15] and a Scanning Electron Microscope (SEM) dataset [AMCC18]. Figure 4.5 shows some of the results we obtained, comparing the original normalized cuts formulation with our new approach. We can see in these examples how the new approach can segment challenging images in a variety of settings, often outperforming the original normalized cuts formulation.

Figure 4.6 illustrates segmentation results using MixNCut to partition an image into 3 regions. In this case we follow the approach suggested in [NJW02] and [MS01], using  $k$ -means with  $k = 3$  to cluster the pixels using the second and third largest eigenvector of the transition matrix  $P$  in equation (4.7).

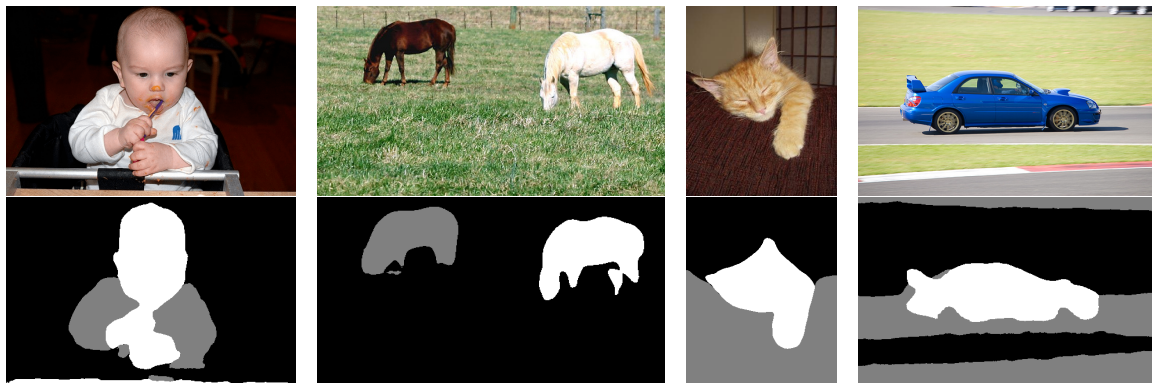
For each example in these figures, we ran the algorithms using different parameter values (specified in the next section), and show the best result among the different runs.

We also proceed as in Section 3.6.6 and evaluated the proposed MixNCut method’s performance on the SED1 and SWINSEG datasets, also described in Section 3.6.6. Here, we resized the images in the latter database to half of their original size for runtime and memory constraints. We tested MixNCut and NCut for the same set of parameters described on the previous paragraph, selecting the best parameter combination based on its  $F$ -score,  $F_\beta$  for  $\beta = 0.3$ . We again also consider swapping the labels of each segmentation as a possible result.

Table 4.1 shows the segmentation results as on SED1 reported in [WWL<sup>+</sup>18], using  $F_\beta$  as an evaluation metric, and in [AGBB11], using  $F_1$  instead, and compare them to our methods. Again, we use these two different measures in order to compare to existing evaluations in the literature. Note that our implementation of NCut improves on the performance of original segmentation algorithm proposed in [SM00]. Note too, that MixNCut is able to achieve segmentation results that are on par with some Deep Learning based methods, such as the ones introduced in [LY16] and [LY15].



**Figure 4.5:** Segmentation results comparing NCut and MixNCut on real images. Column (a) shows the input images. Column (b) shows the eigenvector found by the original NCut formulation on the left and the segmentation result on the right. Column (c) shows the eigenvector found by the new MixNCut formulation on the left and the segmentation result on the right.



**Figure 4.6:** Segmentation results using the proposed method for images with more than 2 regions.

Table 4.2 shows the segmentation results for the proposed methods on the SWINSEG. Note that here we are not able to use the reported results from [DSLW17] for comparison because of our choice for a reduced image size.

#### 4.4.7 Experiments in Synthetic Images

For a quantitative evaluation we used images with Brodatz textures [Bro66]. To generate input images, we mixed pairs of textures using different ground-truth segmentation patterns and resized the result to  $320 \times 320$  pixels.

We compare our method to NCut, using either graylevel intensities or “texture features”, where we use the magnitudes of the response of 12 Gabor filters (3 wavelengths and 4 orientations) to define appearance vectors for each pixel. We also contrast our proposed algorithm with the texture segmentation methods mentioned in Section 2.2. They include LSWD, FBS, PNMF and ORTSEG. In all these methods we try different combinations of their parameters and select the ones that performed the best in our data based on their  $J$  value. We run our comparisons using the implementations provided by their authors. As explained in Section 2.2, FBS and PNMF resort to Gabor filtering in their algorithms, whereas LSWD and ORTSEG make use of local image histograms.

Furthermore, we also compare our method with the Multi-view Spectral Clustering (MVSC) algorithm [ZB07]. It applies the normalized cuts algorithm on a convex combination of two

**Table 4.1:** Comparative Segmentation Performance on the SED1 Database.

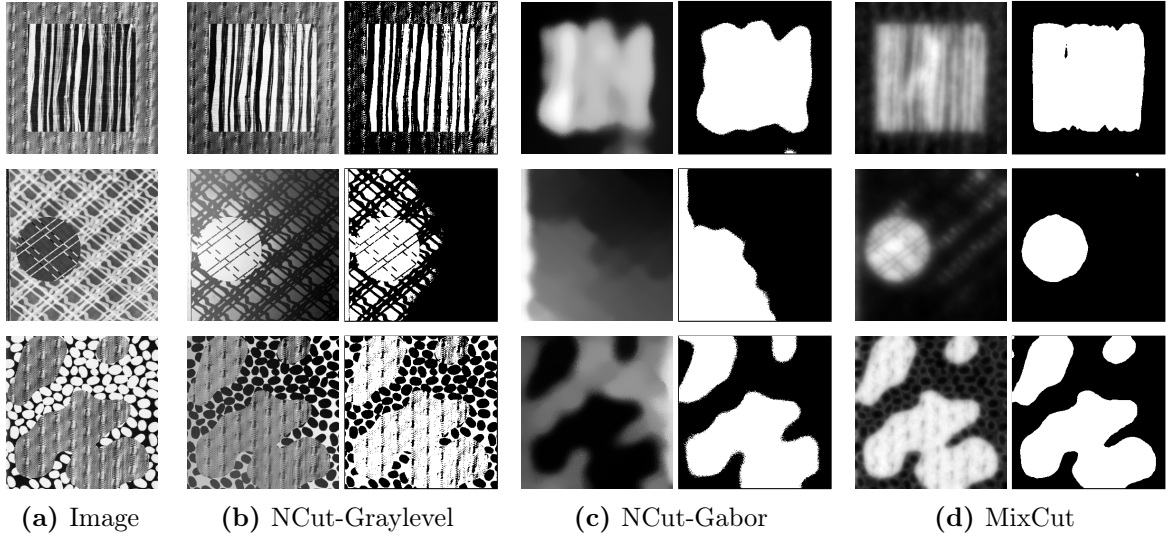
<b>Method</b>	<b>Measure</b>	
	$F_\beta$	$F_1$
MixNCut	0.8660	0.87
NCut	0.8188	0.81
[WWL <sup>+</sup> 18]	0.8811	–
[LY16]	0.8546	–
[LY15]	0.8194	–
[TLRY15]	0.7675	–
[ZLWS14]	0.7889	–
[AGBB11]	–	0.86
[GSBB03]	–	0.83
[SM00]*	–	0.72
[CM02]	–	0.57

\* Implementation of NCut as reported in [AGBB11]

**Table 4.2:** Comparative Segmentation Performance on the SWINSEG Dataset.

<b>Method</b>	<b>Measure</b>		
	$F_1$	Precision	Recall
MixNCut	0.8316	0.8975	0.8065
NCut	0.7952	0.8799	0.7568
[YLM <sup>+</sup> 09]	0.79	0.98	0.69
[YLM <sup>+</sup> 10]	0.26	0.90	0.19
[SWCX17]	0.77	0.68	0.93
[GAL16]	0.62	0.47	0.99
[DSLW17]	0.83	0.95	0.76

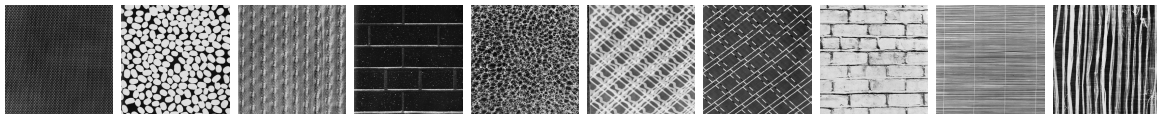




**Figure 4.7:** Comparing NCut-Graylevel, NCut-Gabor, and MixNCut on textured images. Column (a) shows the input images. Column (b) shows the eigenvector found by the original NCut formulation on the left and the segmentation result on the right. Column (c) shows the eigenvector found by NCut with Gabor features on the left and the segmentation result on the right. Column (d) shows the eigenvector found by the new MixNCut formulation on the left and the segmentation result on the right.

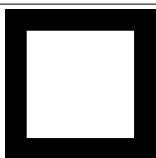
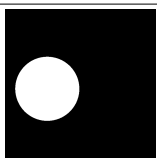
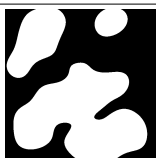
Laplacian matrices representing different graphs/views. For MVSC, we input  $w_{\text{grid}}$  and  $w_{\text{data}}$  to their algorithm and tested their results using the same values of  $\lambda$  and  $\sigma$  as in MixNCut, picking the best result segmentation according to their  $J$  value.

Qualitatively, Figure 4.7 shows some of the input images along with the computed eigenvectors and segmentations arising from our proposed MixNCut method and the tested NCut formulations. The poor segmentation performance of NCut defined over the pixel graylevels (Figure 4.7b) can be attributed to its inability to handle textures. Since it only processes the pixels intensities within a certain radius, the long range pixel relations that are typical in textures and important in their modeling are missed. This issue is partially solved when gabor features are considered, but it has the drawback of over-smoothing region boundaries (first and third examples in Figure 4.7c). In fact, in some extreme cases, it misses an entire small region (second row of Figure 4.7c). On the other hand, the new MixNCut method defined directly in terms of “raw” pixel values finds near optimal segmentations in all of these examples, preserving well the region boundaries and outperforming both baselines. This mainly due to its capacity to model the global image information without relying on



**Figure 4.8:** Brodatz Patterns used in the synthetic experiments

**Table 4.3:** Evaluation of different segmentation methods on textured images. The table summarizes accuracy and running time of each method on images with different ground-truth segmentations.

Method	<i>J</i> value			Time (s)
				
MixNCut	$0.906 \pm 0.080$	$0.876 \pm 0.107$	$0.842 \pm 0.133$	11.27
NCut-Graylevel	$0.541 \pm 0.128$	$0.470 \pm 0.126$	$0.538 \pm 0.130$	9.53
NCut-Gabor	$0.800 \pm 0.173$	$0.779 \pm 0.197$	$0.661 \pm 0.193$	13.14
MVSC	$0.731 \pm 0.273$	$0.766 \pm 0.214$	$0.656 \pm 0.284$	11.19
LSWD	$0.852 \pm 0.129$	$0.794 \pm 0.198$	$0.828 \pm 0.119$	67.08
ORTSEG	$0.853 \pm 0.149$	$0.668 \pm 0.274$	$0.826 \pm 0.135$	1.56
FBS	$0.850 \pm 0.151$	$0.778 \pm 0.214$	$0.804 \pm 0.146$	0.10
PNMF	$0.907 \pm 0.092$	$0.733 \pm 0.142$	$0.857 \pm 0.085$	14.59

filtering methods.

For the quantitative experiments, we use all pairings of the 10 textures in Figure 4.8 with three different ground-truth segmentations shown in Table 4.3 to generate three sets of images. We compute the mean accuracy of each method on each set of images using several parameter combinations ( $\sigma_I \in \{20, 30, \dots, 400\}$  and  $\sigma_X \in \{20, 30, \dots, 100\}$  for NCut;  $\lambda \in \{.98, .99, .995\}$  and  $\sigma \in \{.1, 1, 10, 30\}$  for MixNCut). Table 4.3 summarizes the best mean accuracy obtained with each method on each set of inputs. The table also shows the average running time of each method. We see the new MixNCut approach obtains high accuracy on all ground-truth patterns, outperforming the other methods, specially the spectral ones.



## 4.5 Conclusion

We introduced a new spectral method for image segmentation that can segment challenging images while working directly with “raw” pixel values, without any pre-processing or filtering. The approach is based on a novel combination of appearance and spatial grouping cues using two different graphs. We use a dense graph to capture the appearance of regions. This leads to non-parametric models of region appearance. We also introduced a technique that can be used to sparsify the resulting graph to ease the computational burden of spectral segmentation. Our results show that long range interactions can capture the appearance of complex regions and significantly improve the performance of graph-based segmentation methods. The proposed method is practical and it can be applied to different types of images (natural, biomedical, textures, etc.).

# CHAPTER 5

## Penalized Normalized Cuts

In this chapter, we generalize the original Normalized Cuts framework to allow for the incorporation of prior knowledge about how pixels should be grouped into the segmentation pipeline. This additional information comes from two sources: (1) user data about pixels that should be assigned to certain regions (seeds) or about the expected region appearances and (2) segmentation cues concerning each region’s color statistics. In order to accommodate these new features, a plug-and-play penalty function is added to the original Normalized Cuts formulation without requiring the explicit construction of a large, dense matrix. Although many strategies have been proposed to add seed information to the Normalized Cuts-based segmentation pipeline [EOK11, MVM11, CC15], this is, to the best of our knowledge, the first algorithm to add global appearance data and other segmentation cues to the original spectral framework. Our promising preliminary results show that the proposed methods out-perform the traditional spectral clustering algorithm and can be successfully used in image segmentation tasks.

### 5.1 Prior work and its limitations

Since the conception of the Normalized Cuts criteria and algorithm [SM00], many attempts have been made to generalize these concepts to include additional constraints. In particular, this body of work has been divided into algorithms that include these constraints in a hard or in a soft manner. For the former, two important papers are: the work of [YS01], which added linear constraints of the form  $U\mathbf{x} = 0$  to the spectral pipeline, and the ideas in [EOK11], that generalized that result to problems with constraints like  $U\mathbf{x} = b$ .

As it has been shown in later results, enforcing constraints in a hard way is not robust in typical real world scenarios [MVM11]. That observation motivated the development of techniques that encompass the additional constraints softly. The main works here are:

**Biased Normalized Cuts** The authors in [MVM11] proposed to modify a solution to the original Normalized Cuts algorithm towards being correlated with a predefined vector that embodies the additional constraints. Based on the theoretical work of [MOV12], their proposed algorithm was able incorporate the information about pixels that should be linked in the final spectral solution (seeds) with a negligible additional runtime to the traditional NCut algorithm. On the other hand, the Biased NCut algorithm is not able to handle multiple sets of seeds, such as foreground and background seeds, which hinders its application in certain image segmentation problems.

**Semi-supervised Normalized Cuts** In an attempt to create a algorithm that is able to tackle problems with both must-link seeds (pixels that should be linked in the final solution) and cannot-link seeds (pixels that should not be assigned to be together), the authors in [CC15] developed a soft constrained generalization of NCut called Semi-supervised NCut. In their work, they suggested a new cut criteria that incorporated both sets of seed constraints and proposed approximate minimizers via eigenvalue problem solvers. These solvers, on the other hand, require complex algebraic calculations that hinder its implementation and interpretability. Furthermore, the proposed method is hard to generalize in other to embody different kinds of constraints other than seeds.

In this chapter we propose a simple, easy to implement, and scalable, normalized cut criteria that is able incorporate different constraints in a plug-and-play fashion via a penalty function. We then derive four possible constraints that can be added in a soft manner to our methodology and show that they lead to interpretable segmentation objective functions. In particular, we show that the seed data can be used here to encompass both must-link and cannot-link pixel information.

## 5.2 Adding the Penalty

Let  $\phi(A, B|G) = \|J^\top \mathbf{y}\|_2^2$  be a quadratic function of the cut  $(A, B)$  of graph  $G$ , where  $\mathbf{y} \in \{-1, 1\}^n$  is the labeling vector representing  $(A, B)$  and  $J \in \mathbb{R}^{n \times m}$  is a matrix such that  $J^\top \mathbf{1} = \mathbf{0}$ . Our goal is to study the solutions of the following cut minimization problem:

$$\min_{A, B} \frac{\text{Cut}(A, B|G) - \alpha\phi(A, B|G)}{\text{Vol}(A|G)} + \frac{\text{Cut}(A, B|G) - \alpha\phi(A, B|G)}{\text{Vol}(B|G)}, \quad (5.1)$$

where  $\alpha > 0$  is a constant. In the above,  $\phi(A, B|G)$  acts as a penalty function and, depending on the choice of  $J$ , it will enable to addition of soft constraints and segmentation cues to the traditional Normalized Cut formulation in Section 2.3. We call the above minimization problem and clustering criterion of *Penalized Normalized Cuts*. Our methodology is based on the following proposition:

**Proposition 6.** *For  $J \in \mathbb{R}^{n \times m}$ , the minimization problem in Eq. 5.1 can be approximately solved via a generalized eigenvalue problem.*

*Proof.* We proceed similarly to what is described in Eqs. 2.33-2.37. Introducing the new variables  $v_A$  and  $v_B$  and defining  $\mathbf{d} = D\mathbf{1}$ , we can rewrite Eq. 5.1 in the following way:

$$\begin{aligned} \min_{\mathbf{x}, v_A, v_B} \quad & \left( \frac{v_A + v_B}{4v_A v_B} \right) \mathbf{y}^\top (L - \alpha J J^\top) \mathbf{y} \\ \text{s.t.} \quad & \mathbf{d}^\top \mathbf{y} = v_A - v_B, \quad \mathbf{d}^\top \mathbf{1} = v_A + v_B \\ & \mathbf{y} \in \{-1, +1\}^n \end{aligned} \quad (5.2)$$

Now, using the transformation in Eq. 2.35:

$$\mathbf{x} = \sqrt{\frac{v_A + v_B}{4v_A v_B}} \left( I - \frac{\mathbf{1}\mathbf{d}^\top}{\mathbf{d}^\top \mathbf{1}} \right) \mathbf{y} \quad (5.3)$$

and using the fact that  $J^\top \mathbf{1} = 0$ , we have:

$$\begin{aligned}
\min_{\mathbf{x}, v_A, v_B} \quad & \mathbf{x}^\top L \mathbf{x} - \alpha \|J^\top \mathbf{x}\|_2^2 \\
\text{s.t.} \quad & \mathbf{d}^\top \mathbf{x} = 0, \quad \mathbf{d}^\top \mathbf{1} = v_A + v_B, \quad \mathbf{x}^\top D \mathbf{x} = 1 \\
& \mathbf{x} \in \left\{ -\sqrt{\frac{v_A}{(v_A + v_B)v_B}}, \sqrt{\frac{v_B}{(v_A + v_B)v_A}} \right\}^n
\end{aligned} \tag{5.4}$$

A spectral relaxation of the above problem can be accomplished by minimizing only on  $\mathbf{x}$  and dropping the combinatorial constraint on it. That leads us to the following problem:

$$\begin{aligned}
\min_{\mathbf{x}} \quad & \mathbf{x}^\top L \mathbf{x} - \alpha \|J^\top \mathbf{x}\|_2^2 \\
\text{s.t.} \quad & \mathbf{1}^\top D \mathbf{x} = 0, \quad \mathbf{x}^\top D \mathbf{x} = 1,
\end{aligned} \tag{5.5}$$

From which we derive the following:

$$\left( L - \alpha J J^\top \right) \mathbf{x} = D \mathbf{x} \Leftrightarrow L \mathbf{x} = (D + \alpha J J^\top) \mathbf{x}. \tag{5.6}$$

The matrix  $D + \alpha J J^\top$  is Positive Definite and, therefore, the above problem constitutes a valid Generalized Eigenvalue Problem.  $\square$

Computationally, one could be reluctant of forming potentially dense matrices of the form  $J J^\top$ , due to its memory demand,  $O(n^2)$ , and the spectral decomposition runtime requirement,  $O(n^3)$ . However, for low  $m$ , the generalized eigenvector decomposition in Eq. 5.6 can be efficiently solved via traditional Power Methods, such as the Lanczos algorithm, that does not require  $J J^\top$  to be formed explicitly. That strategy was used in [CC15] when the authors confronted a similar issue.

In the next section, we explore the many ways  $J$  can be set in order to encompass several clustering and segmentation cues or constraints.

## 5.3 Possible Penalties and Related Segmentation Cues

### 5.3.1 Seeds

We start by the task of adding the constraints imposed by seed information to the formulation in Eq. 5.1. Let  $I$  be an image with  $n$  pixels. Let  $Fore$  be the set of pixels that are set to belong to the images foreground and, similarly,  $Back$  be the ones in the background. Define  $\mathbf{s} \in \mathbb{R}^n$  as follows:

$$\mathbf{s}(i) = \frac{n_s}{|Fore|} \mathbb{1}(I(i) \in Fore) - \frac{n_s}{|Back|} \mathbb{1}(I(i) \in Back), \quad (5.7)$$

for all pixels  $i \in \{1, \dots, n\}$ , where  $n_s = |Fore| + |Back|$ . Now let  $\mathbf{x} \in \{-1, 1\}^n$  be the labeling vector of a cut  $(A, B)$ . Without loss of generality, assume the pixels in  $A$  (resp.  $B$ ) are labeled as 1 (resp.  $-1$ ) in  $\mathbf{x}$ . Then we have that:

$$\begin{aligned} \mathbf{x}^\top \mathbf{s} &= n_s \left[ \left( \frac{|A \cap Fore|}{|Fore|} - \frac{|B \cap Fore|}{|Fore|} \right) + \left( \frac{|B \cap Back|}{|Back|} - \frac{|A \cap Back|}{|Back|} \right) \right] \\ &= n_s [(TPR - FNR) + (TNR - FPR)] \\ &= n_s [(2 \times \text{Sensitivity} - 1) + (2 \times \text{Specificity} - 1)], \end{aligned} \quad (5.8)$$

where the rates are defined using the convention that region  $A$  should be set as the foreground and  $B$  the background. Therefore, Sensitivity (resp., Specificity) concerns the amount of foreground (resp., background) seeds set to the foreground (resp., background) region.

Furthermore,  $\|\mathbf{x}^\top \mathbf{s}\|_2^2$  accounts for the square of the sum of Sensitivity + Specificity, which is maximized if both sensitivity and specificity is as high as possible. That setting is attained when the seeds are placed in different segments according to their types, foreground our background. Finally, since  $\mathbf{s}^\top \mathbf{1} = 0$ , our framework allows us to set  $J = \mathbf{s}$ .

### 5.3.2 Region Color Histograms

For an image  $I$  with  $L$  colors, consider the one-hot encoding  $H \in \mathbb{R}^{n \times L}$  of image's pixel colors, i.e,  $H(i, j) = \mathbb{1}\{\text{Pixel } I(i) = \text{Color } j\}$ . Let  $\mathbf{x}$  be an indicator vector of a segmentation  $(A, B)$  of  $I$  and  $\mathbf{h}_A$  and  $\mathbf{h}_B$  be the unnormalized color histograms of  $A$  and  $B$ , respectively.

We have that:

$$\|H^\top \mathbf{x}\|_2^2 = \|\mathbf{h}_A - \mathbf{h}_B\|_2^2, \quad (5.9)$$

Ideally, we would like to find a segmentation such that the histograms  $\mathbf{h}_A$  and  $\mathbf{h}_B$  are as distant as possible, i.e., that  $\|H^\top \mathbf{x}\|_2^2$  is maximized. Therefore, one is tempted to set  $J$  to  $H$  in our formulation of  $\phi(A, B)$ . However, this setting is not allowed, since  $H\mathbf{1} = \mathbf{h}$ , where  $\mathbf{h}$  is the unnormalized histogram of  $I$ , and therefore  $\mathbf{1} \notin \text{Null}(H^\top)$ .

On the other hand, we can then define

$$\tilde{H} = H - \frac{1}{n} \mathbf{1} \mathbf{h}^\top \quad (5.10)$$

and assure the above requirement, i.e,  $\tilde{H}^\top \mathbf{1} = \mathbf{0}$ , making  $J = \tilde{H}$  a valid matrix for  $\phi(A, B)$ .

We also have that:

$$\begin{aligned} \left(H - \frac{1}{n} \mathbf{1} \mathbf{h}^\top\right)^\top \mathbf{x} &= (\mathbf{h}_A - \mathbf{h}_B) - \left(\frac{|A| - |B|}{n}\right) \mathbf{h} \\ &= \frac{(n - |A| + |B|)\mathbf{h}_A + (-n - |A| + |B|)\mathbf{h}_B}{n} \\ &= \frac{2(|B|\mathbf{h}_A - |A|\mathbf{h}_B)}{n} = \frac{2|A||B|}{n} (\mathbf{p}_A - \mathbf{p}_B), \end{aligned} \quad (5.11)$$

where  $\mathbf{p}_\mathcal{R} = \mathbf{h}_\mathcal{R}/|\mathcal{R}|$  is the appearance model of region  $\mathcal{R} \in \{A, B\}$ . This leads to:

$$\|\tilde{H}^\top \mathbf{x}\|_2^2 = \left(\frac{2|A||B|}{n}\right)^2 \|\mathbf{p}_A - \mathbf{p}_B\|_2^2, \quad (5.12)$$

which is also a quantity whose maximization is desirable, since we want the segments' appearance models to be as different as possible, while assuring that  $|A| \approx |B|$ .

### 5.3.3 Region Mean Colors

Let  $C \in \mathbb{R}^{n \times L}$  be the matrix (resp, vector) of RGB colors (resp, graylevels) of  $I$ . With the binary assignment vector  $\mathbf{x}$ , we have:

$$\|C^\top \mathbf{x}\|_2^2 = \left\| \sum_{i \in A} I(i) - \sum_{i \in B} I(i) \right\|_2^2 = \||A|\mu_A - |B|\mu_B\|_2^2 \quad (5.13)$$

where  $\mu_{\mathcal{R}} = \frac{1}{|\mathcal{R}|} \sum_{i \in \mathcal{R}} I(i)$ . Again, note that  $J = C$  does not have the required property of  $J^\top \mathbf{1} = \mathbf{0}$ , so we set  $J = \tilde{C}$  with

$$\tilde{C} = C - \mathbf{1}\mu^\top, \quad (5.14)$$

where  $\mu \in \mathbb{R}^L$  is the average color of  $I$ . We also have that:

$$\begin{aligned} (C - \mathbf{1}\mu^\top)^\top \mathbf{x} &= (|A|\mu_A - |B|\mu_B) - \left( \frac{|A| - |B|}{n} \right) (|A|\mu_A + |B|\mu_B) \\ &= \frac{|A||B|}{n} (\mu_A - \mu_B), \end{aligned} \quad (5.15)$$

which leads to:

$$\|\tilde{C}^\top \mathbf{x}\|_2^2 = \left( \frac{2|A||B|}{n} \right)^2 \|\mu_A - \mu_B\|_2^2, \quad (5.16)$$

Therefore, by maximizing the above equation, we maximize the discrepancy between the mean colors of each region, while making segmented regions have similar sizes ( $|A| \approx |B|$ ).

### 5.3.4 Global Appearance Models

Let  $\theta_s(c) = P(I(i) = c | i \in \mathcal{R}_s)$  be the marginal probability for the color  $c$  of a pixel given that it belongs to the  $s$ -th ground truth region of  $I$ . In section 2.1.3, that is defined as the appearance model of  $\mathcal{R}_s$ . Assume that  $I$  has two regions  $\mathcal{R}_0$  and  $\mathcal{R}_1$  and that  $\theta_0(i)$  and  $\theta_1(i)$  are given. Define  $\mathbf{t} \in \mathbb{R}^n$  as follows:

$$\mathbf{t}(i) = \log \left( \frac{\theta_0(I_i)}{\theta_1(I_i)} \right) \quad (5.17)$$

Now, for a binary assignment vector  $\mathbf{x}$ , define  $l$  as:

$$\begin{aligned} l = \mathbf{x}^\top \mathbf{t} &= \sum_{i \in A} \log \left( \frac{\theta_0(I_i)}{\theta_1(I_i)} \right) + \sum_{i \in B} \log \left( \frac{\theta_1(I_i)}{\theta_0(I_i)} \right) \\ &= \log \left( \frac{\prod_{i \in A} \theta_0(I_i)}{\prod_{i \in A} \theta_1(I_i)} \right) + \log \left( \frac{\prod_{i \in B} \theta_1(I_i)}{\prod_{i \in B} \theta_0(I_i)} \right). \end{aligned} \quad (5.18)$$

Finding  $A$  that maximizes the first term above corresponds to an assignment that maximizes the likelihood of the colors in  $A$  being iid generated from the marginal distribution of  $\mathcal{R}_0$ , while minimizing that they were generated from the marginal distribution of  $\mathcal{R}_1$ . The opposite happens to  $B$  in the second term. In sum, the above metric finds  $A$  and  $B$  that



best match the statistics provided by the appearance models  $\theta_0$  and  $\theta_1$ , considering that their colors were generated iid from these models.

Notice, however, that  $\mathbf{t}^\top \mathbf{1} \neq 0$  as previously required in order to be set to  $J$ . To solve this, let  $J = \tilde{\mathbf{t}}$  with

$$\tilde{\mathbf{t}} = \mathbf{t} - \nu \mathbf{1}, \quad (5.19)$$

where  $\nu = \frac{1}{n} \sum_i \mathbf{t}(i)$  is the mean of  $\mathbf{t}$ . Furthermore:

$$(\mathbf{t} - \nu \mathbf{1})^\top \mathbf{x} = l - \nu(|A| - |B|) \text{ and } \|\tilde{\mathbf{t}}^\top \mathbf{x}\|_2^2 = (l - \nu(|A| - |B|))^2 \quad (5.20)$$

Maximizing the above metric also maximizes Eq. 5.18 and encourages the segmented regions have similar sizes ( $|A| \approx |B|$ ), since  $\nu$  is constant for the image.

### 5.3.5 Combining Cues and Summary

One can also set  $J$  to be a combination of various segmentation cues and constraints. Let  $J_1, \dots, J_K$  be  $K$  matrices used to encode segmentation cues and constraints, such as the ones discussed in the previous sections. Then, for a segmentation  $(A, B)$  of graph  $G$ , labeled according to the binary vector  $\mathbf{x}$ , defining:

$$\phi(A, B|G) = \sum_{k=1}^K \beta_k \mathbf{x} J_k J_k^\top \mathbf{x}, \quad (5.21)$$

for  $\beta_1, \dots, \beta_k > 0$ , leads to a cut criteria that encompasses all the desired segmentation cues and constraints.

Table 5.1 summarizes the proposed penalties for the various segmentation cues studied here.

## 5.4 Preliminary Results

In this section, we present some synthetic and real examples of proposed methods performance. Here, we show that despite their simplicity and generality, they attain desirable clustering and segmentation results in a scalable runtime. On the other hand, these results are only preliminary. A further investigation of these algorithms' performance in different settings

**Table 5.1:** Summary of proposed penalties for the Penalized Normalized Cut formulation

Constraint/Cue	$J$ (definition)	Measure maximized
Seeds	$\mathbf{s}$ (Eq. 5.7)	Seed Sensitivity + Specificity
Histogram Disparity	$\tilde{H}$ (Eq. 5.10)	$\ \mathbf{p}_A - \mathbf{p}_B\ _2^2$
Color Disparity	$\tilde{C}$ (Eq. 5.14)	$\ \mu_A - \mu_B\ _2^2$
Appearance Model Fit	$\tilde{\mathbf{t}}$ (Eq. 5.19)	$\log\left(\frac{\prod_{i \in A} \theta_0(I_i)}{\prod_{i \in A} \theta_1(I_i)}\right) + \log\left(\frac{\prod_{i \in B} \theta_1(I_i)}{\prod_{i \in B} \theta_0(I_i)}\right)$

and an in-depth comparison of our methods to related approaches is still lacking and it is left as future work.

#### 5.4.1 Synthetic Experiments

To illustrate the impact of the incorporation of our penalties to the original NCut framework, we follow [CC15] and start with two synthetic experiments using points in 2D. In our first experiment, we sample 100 points uniformly at random from 3 regions of  $\mathbb{R}^2$ , here called  $S_1$ ,  $S_2$  and  $S_3$ , summing up a total of  $n = 300$  points. In experiment 2, we repeat the same process for 1000 points in each set. Figures 5.1a and 5.1b show the resulting datasets. We then construct a dense graph with weights  $w(i, j) = \exp(-\|p(i) - p(j)\|_2^2 / 2\sigma^2)$  and  $\sigma = 3$  for each experiment and compute the generalized eigenvector-solution for the traditional, non-penalized, Normalized Cuts problem in each graph, depicted in Figures 5.1c and 5.1d. Note that the algorithm correctly detects the datasets' constituent groups in both experiments. In each figure, we also show the elapsed time (ET) for each generalized eigenvector computation. For all the next implementations of the Penalized Normalized Cuts formulation (Eq. 5.1) in this section,  $\alpha$  is set to 100. The algorithms in this section were implemented in MATLAB and run on the same hardware specifications of Section 3.6.

In the following experiments, we also differentiate two strategies to compute the generalized eigenvectors in Eq. 5.6:

**Dense:** We compute matrices  $L$  and  $D + \alpha J J^\top$  and then perform the traditional Lanczos algorithm in order to compute the first 2 generalized eigenvectors of our problem. Here,

it is required for the solver to explicitly compute a dense  $n \times n$  matrix,  $JJ^\top$ , and perform matrix multiplications involving  $D + \alpha JJ^\top$ . In MATLAB code, we run `eigs(L, D+ alpha * J*J', 2, 'sm')`.

**Implicit:** Here we take advantage of the structure in  $D + \alpha JJ^\top$  to create a matrix function  $f(\mathbf{y})$  to replace the computation of  $D + \alpha JJ^\top$  present within the Lanczos method to solve the generalized eigenvalue problem in Eq. 5.6. Many approaches can be applied in this case. Here we choose to approach it by solving:

$$(D + \alpha JJ^\top)^{-1}L\mathbf{x} = \lambda\mathbf{x}. \quad (5.22)$$

For  $J \in \mathbb{R}^{n \times m}$  with  $m \ll n$ , the matrix inversion above can be efficiently computed via the Woodbury matrix identity [Gut46]:

$$(D + \alpha JJ^\top)^{-1} = D^{-1} - \alpha D^{-1}JOJ^\top D^{-1}. \quad (5.23)$$

where  $O = (D^{-1} + \alpha J^\top D^{-1}J)^{-1}$ . Now, recall  $L = D - W$  and let  $M = D^{-1}W$ ,  $v_1 = \alpha D^{-1}JO \in \mathbb{R}^{n \times m}$  and  $v_2 = (1 - M)^\top J \in \mathbb{R}^{n \times m}$ . For  $\mathbf{y} \in \mathbb{R}^n$  and using Eq. 5.23, we define  $f(\mathbf{y})$  as:

$$f(\mathbf{y}) = (D + \alpha JJ^\top)^{-1}L\mathbf{y} = -M\mathbf{y} - v_1v_2^\top\mathbf{y}. \quad (5.24)$$

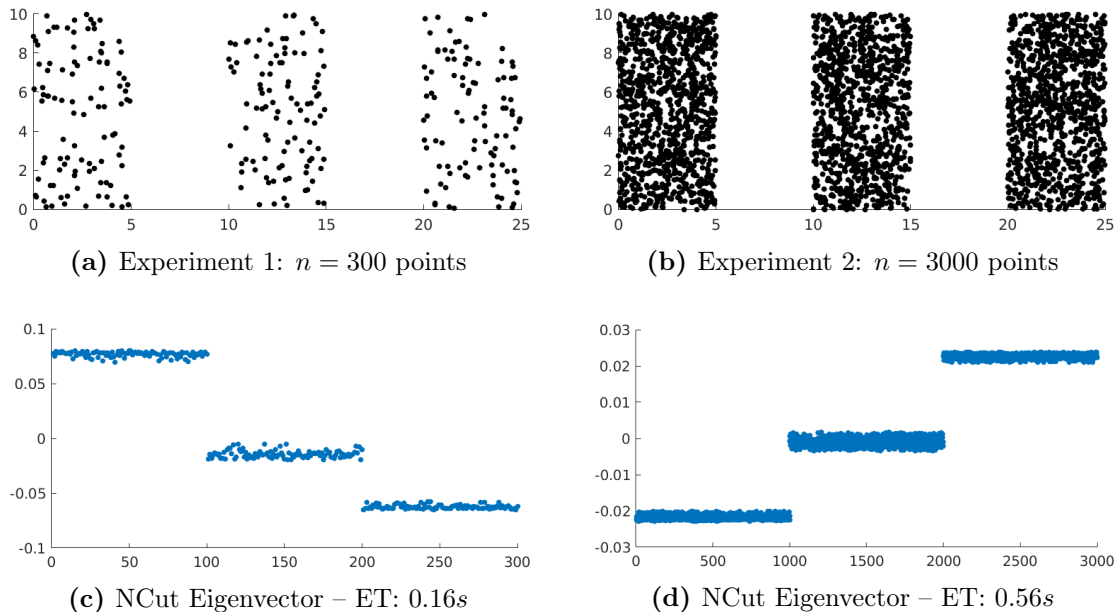
Having  $M$ ,  $v_1$  and  $v_2$  computed offline and assuming  $W$  sparse and  $m \ll n$ , the above expression (used within the Lanczos method) can be computed in time linear in  $n$ . In MATLAB, we code it as `eigs(@(x) M*x + v1*(v2'*x), n, 2, 'lm')`<sup>1</sup>.

## Experiments with seeds

In order to simulate the impact of using the seeds penalty from Section 5.3.1 in our algorithm, we set 10% of each dataset to assume a seed role. Figures 5.2a and 5.2c show the seeds selected from each dataset, circled in blue and red. In each set of each experiment, the

---

<sup>1</sup>According to MATLAB's documentation, in order to define  $f(\mathbf{y})$  as a linear function of  $\mathbf{y}$ , we cannot use the option for searching for the second smallest eigenvector ('sm'). Therefore, we negate  $f(\mathbf{y})$  in Eq. 5.24 and search for the second *largest* eigenvector ('lm' option), which entails the same result.



**Figure 5.1:** Datasets used in our synthetic experiments and generalized eigenvectors computed from the traditional NCut formulation. In each experiment, the leftmost set is  $S_1$ , the central one is  $S_2$  and the rightmost one is  $S_3$ .

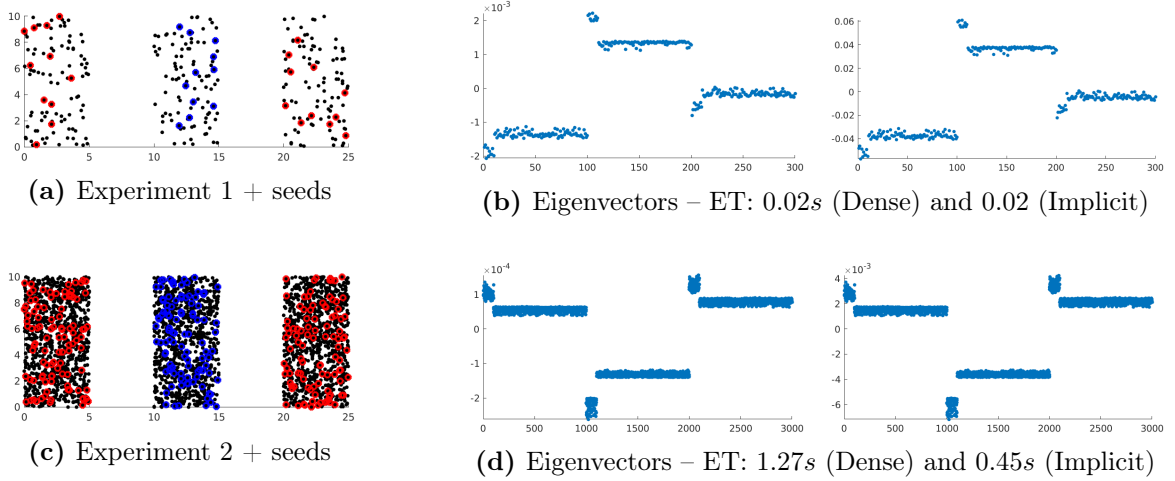
first 10th of its points are selected as seeds. In these examples, the seeds of the same color should be clustered together, while the ones of different colors are expected to be assigned to different clusters.

In Figures 5.2b and 5.2d, we show the resulting penalized eigenvectors according to the procedure in Section 5.3.1. In both experiments, our algorithm is able to better separate the sets marked with the red seeds from the ones marked with the blue ones, bringing the eigenvectors from Figure 5.1 closer to the expected result when seeds are being considered. Note that this outcome is not hindered by the unbalanced clusters in these experiments. Furthermore, although a perfect pixel assignment separating sets  $S_1$  and  $S_3$  from set  $S_2$  can be attained in both experiments by thresholding the eigenvectors at 0, the separation of the sets becomes more evident when more datapoints are considered. This suggests that as the dataset size increases our algorithm should be able to better cluster its points, but further experiments are necessary to support this claim.

Moreover, seeds of different kinds are expected to be more explicitly distinguishable from one another according to their corresponding eigenvector values, since our formulation encourages

their separation in different clusters. This effectively explains the behaviour of the first 10th of each set’s eigenvector values in both experiments. In some sense, one can intuit that the presence of the seeds “drags” their neighboring datapoints to their respective clusters in our proposed formulation.

We finally note that solving the generalized eigenvector problem without the explicit computation of dense matrices effectively reduces the overall computation time, bringing it closer to the original NCut runtime. This effect is particularly noticeable in larger datasets, as expected.



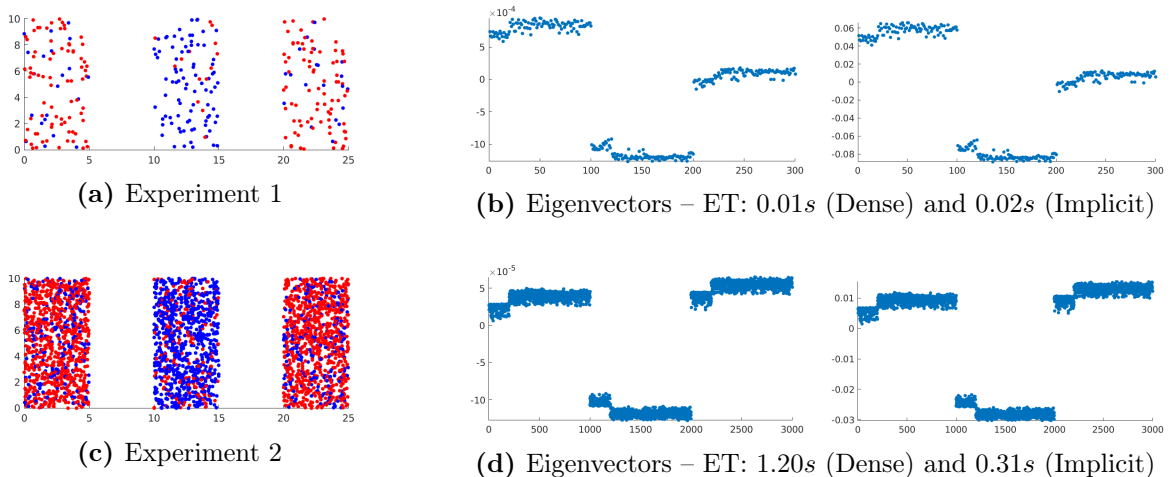
**Figure 5.2:** Results for NCut penalized with seeds information. In (a) and (c), we show the initial datasets with the selected seeds circled in red and blue. In (b) and (d), the computed eigenvectors are shown: on left, for the generalized eigenvalue problem with dense matrices and, on the right, the same problem but without the explicit computation of these matrices. The elapsed time of each solver is also presented.

### Experiments using the histogram disparity penalty

In order to explore the consequences of including the color histogram disparity cue explained in Section 5.3.2, we assign a color out of two possible to each datapoint in our experiments’ datasets. In each set, the first 20% of its datapoints is assigned to a color and the remaining 80% to the other color. We have sets  $S_1$  and  $S_3$  be of different color majority color than  $S_2$ . Figures 5.3a and 5.3c mark the different pixel colors as red and blue. Clearly, in both experiments, sets  $S_1$  and  $S_3$  should be assigned to a different cluster than that of  $S_2$  according to the histogram disparity cue. For these experiments, note that the specific choice of pixel colors is irrelevant, since their one-hot encoding is effectively the same for any selected pair

of colors.

In Figures 5.3b and 5.3d, the resulting penalized eigenvectors are shown along with the computation runtime. Similarly to the seeds case, our algorithm is also able to differentiate the desired clusters, despite their unbalance. Again, the increase of the dataset’s size is also observed to improve the overall separation between the clusters and the usage of the implicit computation of dense matrices also improves the algorithm’s runtime. On the other hand, the initial 20 % pixels in each set have their corresponding eigenvector value closer to zero than the rest of other pixels. This is expected since these pixels find more color-wise similarity in the pixels from the other cluster.



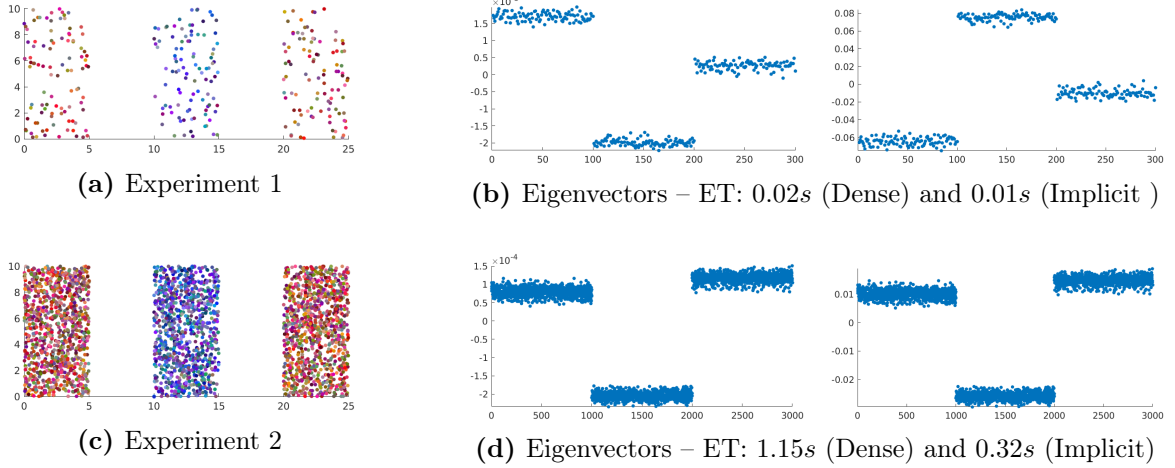
**Figure 5.3:** Results for Ncut penalized with the histogram disparity cue. In (a) and (c), one color out of two possible is assigned to each datapoint. In (b) and (d), the generalized eigenvectors computed using the histogram disparity cue are shown in the same manner as described in Figure 5.2.

### Experiments using the mean color disparity penalty

For the experiments making use of the mean color disparity cue, we assign a RGB color to each datapoint. In order to choose each color, we initially set all points in  $S_1$  and  $S_3$  to red (RGB vector  $[1, 0, 0]$ ) and the point in  $S_2$  to blue (RGB vector  $[1, 0, 1]$ ). Then, for each datapoint, we uniformly at random select a color in the RGB space and add it to its initially assigned RGB value at the proportion of 2 to 1 for the new color. We then normalize it to right RGB range. Figures 5.4a and 5.4c depict the resulting datasets.

Figures 5.4b and 5.4d show the resulting generalized eigenvectors from the problem setting

explained in Section 5.3.3. Here we observe phenomena that are similar to what was described in prior sections, with a particular emphasis on the success of our algorithms to produce the desired clustering according to the region’s mean color disparities, even on an unbalanced setting.

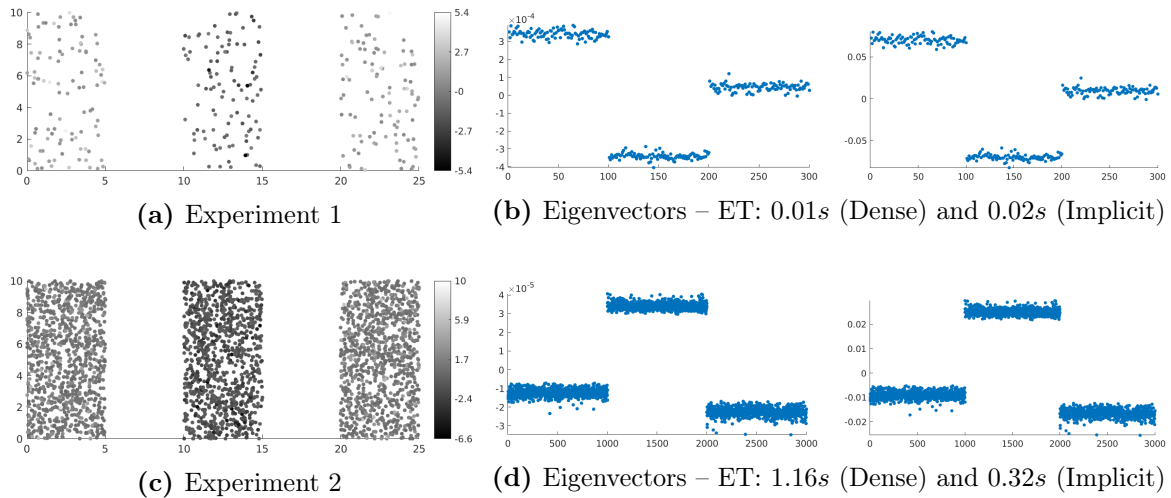


**Figure 5.4:** Results for NCut penalized with the color disparity cue. In (a) and (c), a random RGB color around blue or red is assigned to each datapoint. In (b) and (d), the generalized eigenvectors computed using the mean color disparity cue are shown in the same manner as described in Figure 5.2.

### Experiments with appearance models

Finally, to investigate the influence of the usage of user-provided appearance models in our framework, we assign to each datapoint the probability values of belonging to each of the two possible clusters. We initially set the probability for sets  $S_1$  and  $S_3$  to belong to the first cluster at 1 and to belong to the other cluster at 0. The opposite takes place for set  $S_2$ . Then, we add a uniformly drawn sample in  $(0, 1)$  to each probability value and normalized them accordingly. Figures 5.5a and 5.5c depict the log ratio of these probability values (the vector  $\mathbf{t}$  on Section 5.3.4) for all datapoints.

Figures 5.5b and 5.5d show the generalized eigenvectors computed from our framework when it makes use of the provided appearance models. The same observations from the previous sections can be drawn here, showing again an example of success and pointing to the scalability of our method within this application.



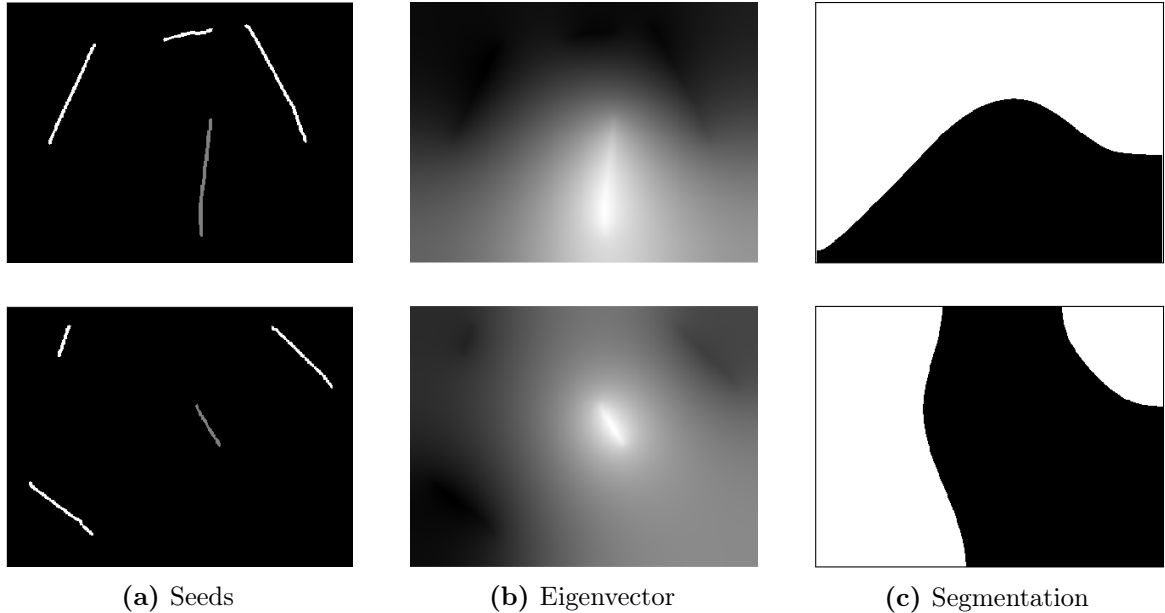
**Figure 5.5:** Results for for NCut penalized with the appearance model data. In (a) and (c), a appearance model probability value is assigned to each datapoint according to the set it belongs to. On the image, their log-ratio values are depicted. In (b) and (d), the generalized eigenvectors computed using the appearance model data are shown in the same manner as described in Figure 5.2.

### 5.4.2 Real experiments

In this section we show the effect of using the seed data as a penalty for normalized cuts when segmenting real scenes. We also present some preliminary result of the penalized NCut for the histogram and mean color disparity cues and for the appearance model data constraint on images.

In order to test the effect of image seeds in segmentation, we start by investigating its behaviour on blank images, i.e., here defined as an image whose color data is discarded, but the spatial relations between pixels are kept. In terms of graph Laplacian, in blank images only the grid graph (explained in Section 4.2.1) is constructed and used for clustering purposes. This scenario is important since we can understand how the seeds affect the final segmentation “prior” to the usage of usage of the color data. In Figure 5.6, we visually demonstrate the effect of the seed data in blank images under the penalized normalized cut framework. The presence of seeds induce an eigenvector with “valleys” and “peaks” around the locations of the seeds. This confirms the propensity of our method to assign foreground and background seeds to distinct partitions, which is reflected on distant eigenvector values. The unseeded locations also get values that interpolate those of the seeds, which is expected



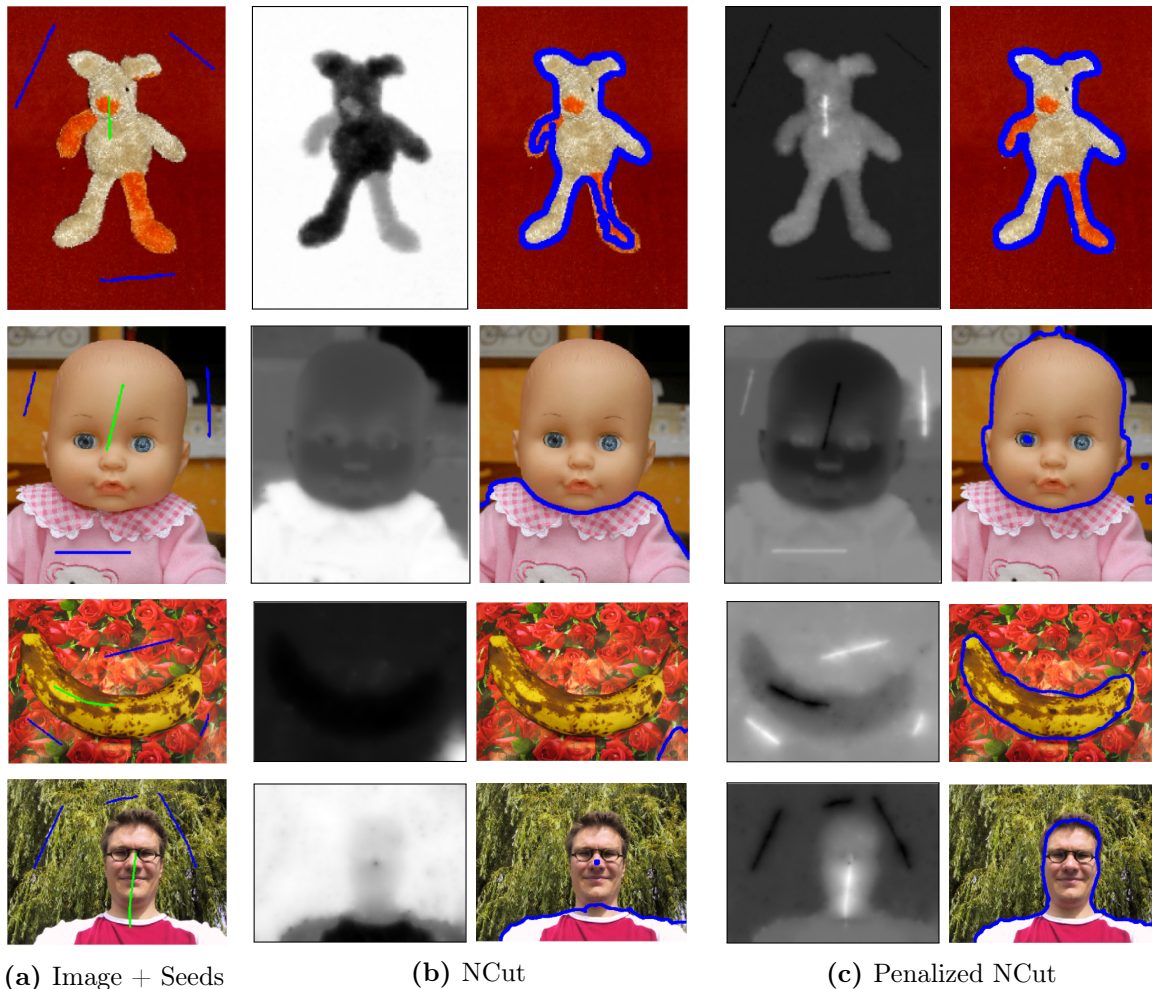


**Figure 5.6:** Segmentation results of a blank image when seeds are added to it. In (a) the proposed seeds are shown in white (background seeds) and dark gray (foreground seeds). The remaining black pixels are unseeded. In (b), we show the resulting penalized normalized cut eigenvector when the seeds penalty is applied ( $\alpha = 1$ ). The resulting segmentation when the eigenvector in (b) is clustered via  $K$ -means is depicted in (c). Image frames were added for visibility. The seed images are from the Grabcut dataset [RKB04].

in seed-aided segmentation. Finally, the resulting segmentations observe the seed presence, which is also expected in our applications. In that Figure, we set  $\alpha = 1$  and used  $K$ -means for clustering the eigenvector values to compute the resulting segmentation.

Figure 5.7 shows the outputs of our methods when using both the spatial and color data to compose the image’s Graph Laplacian  $L$ . In this case, we constructed the final graph as a combination of  $G_{\text{grid}}$  from Section 4.2.1 and  $G_{\text{data}}$  from Section 4.2.2 with  $\lambda = 0.5$  and  $\sigma = 1$ . We again set  $\alpha = 1$  in our penalized framework. Figure 5.7 also presents the resulting eigenvectors and segmentations when no penalty is added<sup>2</sup>. Our method is able to greatly improve the segmentations generated by the “unaided” method. This is already reflected on its eigenvectors, where the desired image regions and boundaries are more clear and unequivocal than those in the non-penalized eigenvectors. Further experimentation, however,

<sup>2</sup>The segmentation results in Chapter 4 were achieved after some parameter tuning. Here we deliberately chose simple parameters when constructing  $L$  in order to make the effect of the penalty more explicit.



**Figure 5.7:** Segmentation results of a real image when seeds are added to it. In (a) the proposed seeds are shown in blue (background seeds) and green (foreground seeds) on the original image. In (b), on the left we show the normalized cut eigenvector without the use of the seed information and on the right the final segmentation. In (c), on the left we show the penalized normalized cut eigenvector when the seeds penalty is applied ( $\alpha = 1$ ) and on the right we show the final segmentation. Images are from the Grabcut dataset [RKB04].

is necessary to understand the effect of  $\alpha$  in this process.

For the quantitative results, we compare the penalized normalized cuts segmentation solutions to the standard normalized cut on all the images of the Grabcut dataset [RKB04]. In both cases, the base graph is such that its Laplacian matrix is again the convex combination of the Laplacian matrices of  $G_{\text{grid}}$  and  $G_{\text{data}}$ , which requires setting the parameters  $\lambda$  and  $\sigma$ . For each method, we set  $\lambda = 0.99$  and  $\sigma = 1$ , as these were the parameters the methods performed the best under according to the foreground detection Jaccard index. As always,

**Table 5.2:** Comparative Segmentation Performance on the Grabcut Dataset for the penalized and the traditional normalized cuts algorithms.

Method	Measure				
	Jaccard	$F_\beta$	Precision	Recall	Acc
Penalized Normalized Cuts	0.4788	0.6442	0.6593	0.7289	0.2309
Normalized Cuts	0.3923	0.5113	0.5193	0.7064	0.3287

we allowed for a permutation of region labels when computing these measures. For the Penalized NCut method, we kept  $\alpha = 1$  in all experiments. We also used the brush strokes provided by [GRC<sup>+</sup>10] as our segmentation seeds. We resized all the image to half of their original sizes for memory saving purposes.

In Table 5.2, we show a performance comparative of our the best choices for both aided and unaided normalized cuts methods in terms of  $F_\beta$  (Eq. 3.22), Precision/Recall (Eq. 3.23) and the Jaccard index (Eq. 3.20) and the Segmentation Accuracy, Acc, defined as

$$\text{Acc} = \frac{1}{|\Omega|} \sum_{i \in \Omega} \mathbb{1}_{S(i)=R(i)}, \quad (5.25)$$

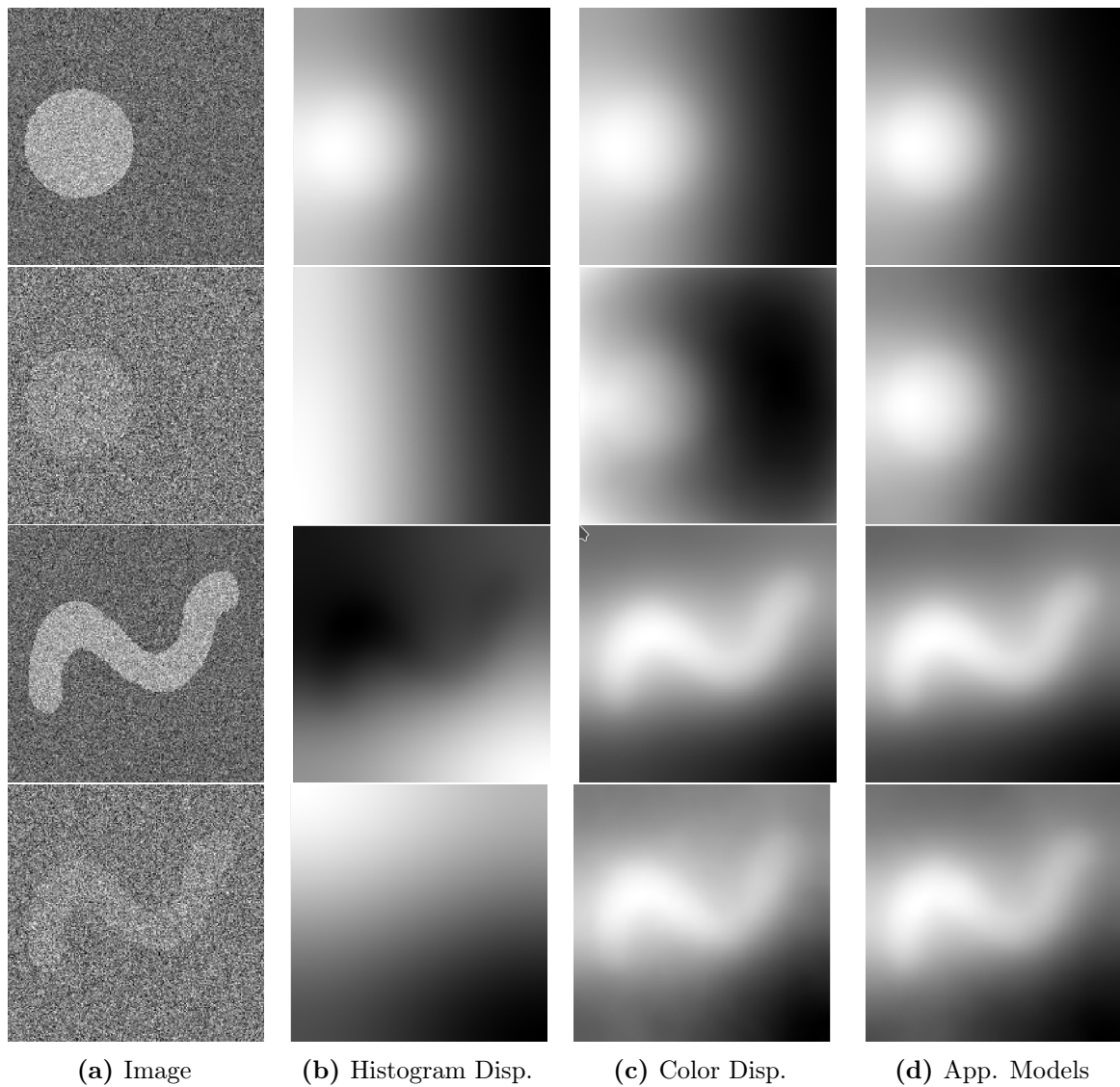
where  $S$  and  $R$  are the estimated and ground-truth segmentations. The overall segmentation algorithm performance, expressed in the  $F_\beta$ , Jaccard and Acc measures, is improved by the addition of the seeds. This is expected since the seeds improve the segmentation algorithm’s ability to localize the foreground regions, increasing both Precision and Recall values and, therefore, the overall  $F$ -score. The comparatively high value of Recall for NCut stems from the many segmentations it produced with very small backgrounds (results similar to the banana image in Figure 5.7). In those cases, the seeds helped locating the foreground and also balancing the region sizes.

In Figure 5.8 we show some results of the penalized normalized cuts for the remaining segmentation penalties. The images of size  $200 \times 200$  pixels were generated from iid Gaussian samples of different means according to GT2 and GT4 in Figure 3.8. In these experiments,  $\alpha = 1$  and the base graph  $G$  consists only of the image’s grid graph, as done in Figure 5.6,

but we do use its color data in the penalties. The foreground region of each image is visible in most resulting eigenvectors, even when it's challenging to detect it on the original image. This provides evidence that the penalty promotes an eigenvector that is biased towards the desired segmentation outcome. Notice also that color disparity is more effective at detecting the foreground than histogram disparity, specially when the noise level is high. This can be used as evidence that the former performs better than the latter in images, which is convenient since the color disparity cue requires a less computationally demanding matrix multiplication in its eigenvector solver.

## 5.5 Conclusion

In this chapter we proposed a scheme to promote constrained spectral image segmentation via a penalized version of the traditional Normalized Cut criteria. We showed the new cut objective function can be approximately minimized via a generalized eigenvalue solver. In its turn, the generalized eigenvectors can also be found efficiently without requiring the computation and storage of large dense matrices. Our work demonstrated that at least four different constraints and segmentation cues can be added to this penalty-based segmentation framework: (1) seed data, (2) region histogram disparity cue, (3) mean region color disparity cue and (4) user-provided appearance model data. These constraints can also be used together as an all encompassing penalty. For all these methods, we presented synthetic experiments that provided evidence for their performance in hard clustering scenarios. We showed that the results were easily interpretable and scalable and that they visually improve as the number of data points increased. Finally we compared the proposed algorithm's performance in some seed-based image segmentation problems to a non-penalized normalized cut solution. The results indicated that the penalty addition was beneficial to the final segmentation solution. Despite these results, a deeper investigation on the impact of different parameters and the comparison of our proposed solvers to established segmentation algorithms is lacking. Furthermore, the application of penalties of (2), (3) and (4) kinds on real images is missing, which makes their usage performance in hard non-synthetic scenarios inconclusive. These experimental deficiencies will be addressed in future work.



**Figure 5.8:** Penalized Normalized Cut Eigenvectors from different segmentation cues/penalties. In (a) we show the original images corrupted with Gaussian noise. The eigenvectors computed from the histogram disparity, color disparity and appearance models cues are shown in (b), (c) and (d), respectively.

# CHAPTER 6

## Conclusion

In this theses we proposed methods that advanced the understanding and improved the performance of traditional approaches to graph-based image segmentation. In sum, we advanced solutions to two problems faced in that community:

- Is it possible to estimate non-parametric appearance models directly from an image without cumbersome user intervention?
- How does one make design an similarity graph for spectral image segmentation that is both interpretable and whose cut performs texture segmentation well even when constructed on raw pixel intensity values?

### 6.1 Contributions

#### 6.1.1 Contributions to appearance modelling

The main results in our work in appearance modeling for image segmentation (Chapter 3) are two fold:

- Departing from typical assumptions on image formation, we described a set of algebraic expressions that correlate an image's appearance models to higher order statistics of the its pixel values.
- Two appearance estimators for images with two natural regions can be derived from that set of expressions, making it possible to approach unsupervised non-parametric appearance modeling and segmentation without iterative procedures. The first estimation algorithm was based on solving a linear system of quadratic equations and the

second had a eigenvalue decomposition at its core.

- One appearance estimation algorithm for images with multiple regions was proposed. It requires the estimation of third order statistics to be used in a method of moments estimator, typical of document topic modeling problems. To our knowledge, it was the first time this strategy was used in image appearance modeling.

Our experiments also demonstrate that the proposed algorithms work well in practice and attain satisfactory segmentation in challenging scenarios. This was particularly noticeable on images whose appearance models were random probability vectors. Our methods were able to distinguish their constituent regions even when it was impossible to do so visually. Furthermore, we showed that this approach was able to segment challenging textured images without any filtering-based preprocessing. Finally, our experiments demonstrated that the proposed methods were computationally efficient and fast compared to other related algorithms. Overall, this work shed light to the benefit of second and third order color statistics to segmentation algorithms based on appearance modeling, suggesting that their use can great light improve their performance in challenging imaging scenarios.

### 6.1.2 Contributions to spectral image segmentation

The work developed in Chapters 4 and 5 advanced the following contributions the spectral image segmentation:

- We developed a data structure that relies on two simple graphs constructed from the original image without any preprocessing. Each graph corresponds to a grouping cue and cuts on them are related to interpretable measures of segmentation quality. The first graph, a sparse grid graph, assess the spatial coherence in the segmented regions and the second graph, a dense graph computed on the image color space, capture the appearance disparity between regions. Our method then searches for low conductivity sets of a random walk composed of both graphs, where the walker is allowed to switch between graphs with a certain probability. This highly interpretable graph construction is, to the best of our knowledge, innovative with the spectral segmentation literature.
- In order to deal with the computationally inefficient dense graph, we suggested a graph

sparsification algorithm based on importance sampling of the edges from the original graph. We also added a color partitioning step to the sparsification pipeline. The overall method was able to efficiently process high resolution color images.

- Our method demonstrated that, contrary to the commonly held assumption, it is possible to attain high quality segmentation results on textured images without any filtering. In fact the proposed segmentation algorithm, which works directly on the raw pixels, attains a very high texture segmentation performance when compared to typical filtering based spectral solvers. The eigenvectors output by our algorithm are able to better preserve segmentation borders, once they are not oversmoothed as in filtering dependent techniques, without losing in region contrast.
- Additionally, we also proposed a graph sparsification algorithm based on importance sampling for traditional Normalized Cuts weight graph. To our knowledge, it was the first time such an approach was used to reduce the computational burden of the spectral image segmentation on its classical formulation.
- Our methods were also shown to greatly outperform the traditional normalized cut formulation, even in the setting where the sparsification procedures are not necessary. This work suggested that long range interactions can capture the appearance of complex regions and significantly improve the performance of spectral segmentation methods and of graph-based segmentation methods, broadly speaking.
- We proposed a new generalization of Normalized Cuts to incorporate certain kinds of penalties. This new framework, here named Penalized Normalized Cuts, also leads to a simple generalized eigenvalue problem that can be made scalable through a clever usage of typical power methods for eigenvector computation, such as the Lanczos algorithm.
- We presented four different ways to set the penalty for Penalized Normalized Cuts leading to the embodiment of four important sources of expert knowledge about segmentations:
  - Seed data: User-provided pixel region assignments prior to the segmentation estimation.



- Histogram disparity cue: the anticipation that different regions should have dissimilar color distributions.
- Mean color disparity cue: the assumption that different regions should have dissimilar mean colors.
- Appearance Model data: User-provided expected color distributions for each region.

For each of the above penalties, we showed that their cut criteria lead to interpretable objective functions in the context of image segmentation. We also provided preliminary synthetic and real experiments demonstrating their efficiency and scalability for spectral clustering and segmentation problems. To the best of our knowledge, these were the first algorithms to attempt incorporating expert knowledge beyond seed data to spectral segmentation algorithms.

## 6.2 Future Work

### 6.2.1 MRF based segmentation and appearance modeling

In Section 3.4, we proposed the use of a method initially developed for topic modeling problems in natural language processing to an image segmentation setting. In our future work, other methods from topic modeling will be explored to appearance estimation purposes. Of particular interest could be the use of Non-negative Matrix Factorization (NMF) for its simplicity and speed and Latent Dirichlet Allocation (LDA) for its generative aspect and its long-standing application on topic modeling problems. On the other hand, it could be of potential interest applying some methods in appearance estimation for image segmentation to topic modeling, drawing more theoretical and practical connections between these two fields.

A possible solution for the appearance estimation problem that was untapped in our work and can be matter for future work is the use of mixtures of Markov chains in image formation modelling. In few words, we can treat sequences of neighboring pixels in an image with  $K$  regions as samples of a mixture of  $K$  Markov chains with transition matrices

$\{M_1, M_2, \dots, M_K\}$ , appearances  $\{\theta_1, \theta_2, \dots, \theta_K\}$  and weights  $\{w_1, w_2, \dots, w_K\}$ . Here, we assume that these pixel sequences are always contained within a sole image region. Now we generate a sequence of  $t$  pixels as follows: sample a chain  $s$  with probability  $w_s$ , select a starting pixel color  $i$  with probability  $\theta_s(i)$ , then perform a random walk according to the transition matrix  $M_s$  and repeat it  $t - 2$  times<sup>1</sup>. In [GKV16], the authors showed that having sample sequences of length three from this process is enough to learn the transition matrices and the *weighted* appearances  $\{w_1\theta_1, w_2\theta_2, \dots, w_K\theta_K\}$ . Applied to our image segmentation problem, this algorithm does not assume the independence of neighboring pixels (Assumption 2), which is an advantage over our modeling in Chapter 3, and can be used in multi region appearance estimation. Future work will study the practical applicability of this algorithm on texture segmentation.

Finally, as discussed at the end of Chapter 3, some investigation of possible ways to adapt the Method of Moments (MoM) estimator to the segmentation of broader and more realistic image settings is still needed. Future work will focus on literature review on possible alternatives to the method of moments for topic modeling within the NLP community, to be used within the multi-region image segmentation setting. We will also attempt to derive new algebraic expressions for the image moments in order to find ways to improve the base MoM estimator. Moreover, some future work will convey a more thoroughly empirical experimentation on how the parameters in Method of Moments estimator behave and whether they could be optimized to produce better segmentations on realistic images.

---

<sup>1</sup>To illustrate this generative process in our imaging problem, take  $\beta$  from Eq. 3.2, assume that the probability that pixels  $x$  and  $y$  belong to different regions is negligible. Then we have that, for two colors  $i$  and  $j$ :

$$\begin{aligned} \beta(i, j) &= \sum_{s=1}^2 w_s P(I(x) = i, I(y) = j \mid x, y \in \mathcal{R}_s) \\ &= \sum_{s=1}^2 w_s P(I(x) = i \mid x \in \mathcal{R}_s) P(I(y) = j \mid I(x) = i; x, y \in \mathcal{R}_s) \\ &= \sum_{s=1}^2 w_s \theta_s(i) M_s(j, i), \end{aligned}$$

which is a mixture of Markov chains. Note that we didn't need to make use of Assumption 2 (independence at a distance) in this derivation.

## 6.2.2 Spectral image segmentation

A straightforward extension of the work developed in Chapter 4 is the study of kernel functions other than the Gaussian, such as the Epanechnikov, Cosine, Tricube, etc. Conducting such research and comparing the performance of these methods could also lead to new local similarity functions that perform better than the traditional formulation in Eq. 2.39.

Furthermore, instead of varying the weight function on these graph, one could also try different graph topologies. For instance, adding extra nodes to the graph, each representing a color in the image, has been successfully tried in MRF and Random Walker segmentation approaches [Gra05, TGVB13]. A study of the effect of this new graph construction in spectral algorithms is still lacking. Our preliminary work on this theme suggests that such an approach is computationally efficient in practice, since it does not require the creation of a dense graph and its further sparsification, and leads to an interpretable optimization problem.

As mentioned at the end of Chapter 5, a more complete experimentation on the proposed methods for Penalized Normalized Cuts is lacking and will be pursued as a future work. In particular, we will explore the effect of the penalty's balancing factor,  $\alpha$ , on the diverse synthetic experiments we proposed. We will also add results of each method on real images and do a throughout evaluation of their runtimes.

Finally, we will study new penalties that could be added to the penalized normalized cut idea. One possible option arises from the observation that the one-hot matrices discussed in Section 5.3.2 can be viewed as responses from each pixel to filters that only detect one particular color. This notion could be broadened to general texture filters, such as Gabor for example. We will also investigate how all these penalties can be added in a simple procedural way to our framework. Finally, on a more theoretical perspective, how each  $J$  is connected to the constraint/cue it entails.

## References

- [AFH<sup>+</sup>12] Anima Anandkumar, Dean P Foster, Daniel J Hsu, Sham M Kakade, and Yi-Kai Liu. A spectral algorithm for latent dirichlet allocation. In *Advances in Neural Information Processing Systems*, pages 917–925, 2012.
- [AGBB11] Sharon Alpert, Meirav Galun, Achi Brandt, and Ronen Basri. Image segmentation by probabilistic bottom-up aggregation and cue integration. *IEEE transactions on pattern analysis and machine intelligence*, 34(2):315–327, 2011.
- [AHK12] Animashree Anandkumar, Daniel Hsu, and Sham M Kakade. A method of moments for mixture models and hidden markov models. In *Conference on Learning Theory*, pages 33–1, 2012.
- [AMCC18] Rossella Aversa, Mohammad Hadi Modarres, Stefano Cozzini, and Regina Ciancio. NFFA-EUROPE - SEM dataset, 2018.
- [AMFM10] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):898–916, 2010.
- [ASS<sup>+</sup>12] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012.
- [BC06] Tijl De Bie and Nello Cristianini. Fast sdp relaxations of graph cut clustering, transduction, and other combinatorial problems. *Journal of Machine Learning Research*, 7(Jul):1409–1436, 2006.

- [Ben10] Frank W Bentrem. A q-ising model application for linear-time image segmentation. *Central European Journal of Physics*, 8(5):689–698, 2010.
- [BFL06] Yuri Boykov and Gareth Funka-Lea. Graph cuts and efficient nd image segmentation. *International Journal of Computer Vision*, 70(2):109–131, 2006.
- [BJ01] Yuri Y Boykov and M-P Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *IEEE International Conference on Computer Vision*, volume 1, pages 105–112, 2001.
- [BK03] Yuri Boykov and Vladimir Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *IEEE International Conference on Computer Vision*, 2003.
- [BMB16] Christos G Bampis, Petros Maragos, and Alan C Bovik. Projective non-negative matrix factorization for unsupervised graph clustering. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 1255–1258. IEEE, 2016.
- [Bro66] Phil Brodatz. *Textures: a photographic album for artists and designers*. Dover Pubns, 1966.
- [BS04] Steffen Bickel and Tobias Scheffer. Multi-view clustering. In *ICDM*, pages 19–26, 2004.
- [BVZ99] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. In *IEEE International Conference on Computer Vision*, volume 1, pages 377–384, 1999.
- [CC15] Selene E Chew and Nathan D Cahill. Semi-supervised normalized cuts for image segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1716–1723, 2015.
- [CCCD93] Vicent Caselles, Francine Catté, Tomeu Coll, and Françoise Dibos. A geometric model for active contours in image processing. *Numerische mathematik*,

- 66(1):1–31, 1993.
- [CM02] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002.
- [CMH<sup>+</sup>14] Ming-Ming Cheng, Niloy J Mitra, Xiaolei Huang, Philip HS Torr, and Shi-Min Hu. Global contrast based salient region detection. *IEEE transactions on pattern analysis and machine intelligence*, 37(3):569–582, 2014.
- [CS96] Jean-François Cardoso and Antoine Souloumiac. Jacobi angles for simultaneous diagonalization. *SIAM journal on matrix analysis and applications*, 17(1):161–164, 1996.
- [CV01] Tony F Chan and Luminita A Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, 2001.
- [CYW<sup>+</sup>08] Jingyu Cui, Qiong Yang, Fang Wen, Qiying Wu, Changshui Zhang, Luc Van Gool, and Xiaoou Tang. Transductive object cutout. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.
- [DGK04] Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. Kernel k-means: spectral clustering and normalized cuts. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 551–556, 2004.
- [DMFU10] Paulo AV De Miranda, Alexandre X Falcão, and Jayaram K Udupa. Synergistic arc-weight estimation for interactive image segmentation using graphs. *Computer Vision and Image Understanding*, 114(1):85–99, 2010.
- [DS05] Virginia R De Sa. Spectral clustering with two views. In *ICML workshop on learning with multiple views*, pages 20–27, 2005.
- [DSLW17] Soumyabrata Dev, Florian M Savoy, Yee Hui Lee, and Stefan Winkler. Night-time sky/cloud image segmentation. In *2017 IEEE International Conference*

*on Image Processing (ICIP)*, pages 345–349. IEEE, 2017.

- [EEVG<sup>+</sup>15] Mark Everingham, S.M. Ali Eslami, Luc Van Gool, Christopher K.I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015.
- [EOK11] Anders Eriksson, Carl Olsson, and Fredrik Kahl. Normalized cuts revisited: A reformulation for segmentation with linear grouping constraints. *Journal of Mathematical Imaging and Vision*, 39(1):45–61, 2011.
- [FBCM04] Charless Fowlkes, Serge Belongie, Fan Chung, and Jitendra Malik. Spectral grouping using the nystrom method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):214–225, 2004.
- [GAL16] Glenn Franco Barroso Gacal, Carlo Antioquia, and Nofel Lagrosas. Ground-based detection of nighttime clouds above manila observatory (14.64n, 121.07e) using a digital camera. *Applied Optics*, 55:6040–6045, 2016.
- [GG84] Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence*, (6):721–741, 1984.
- [GJJ<sup>+</sup>17] Thomas Mosgaard Giselsson, Rasmus Nyholm Jørgensen, Peter Kryger Jensen, Mads Dyrmann, and Henrik Skov Midtiby. A public image database for benchmark of plant seedling classification algorithms. *arXiv preprint arXiv:1711.05458*, 2017.
- [GKV16] Rishi Gupta, Ravi Kumar, and Sergei Vassilvitskii. On mixtures of markov chains. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 3449–3457. Citeseer, 2016.
- [GPS89] Dorothy M Greig, Bruce T Porteous, and Allan H Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society: Series B (Methodological)*, 51(2):271–279, 1989.

- [Gra05] Leo Grady. Multilabel random walker image segmentation using prior models. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 763–770. IEEE, 2005.
- [Gra06] Leo Grady. Random walks for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 28(11):1768–1783, 2006.
- [GRC<sup>+</sup>10] Varun Gulshan, Carsten Rother, Antonio Criminisi, Andrew Blake, and Andrew Zisserman. Geodesic star convexity for interactive image segmentation. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3129–3136. IEEE, 2010.
- [GSBB03] Meirav Galun, Eitan Sharon, Ronen Basri, and Achi Brandt. Texture segmentation by multiscale aggregation of filter responses and shape elements. In *ICCV*, volume 3, page 716, 2003.
- [Gut46] Louis Guttman. Enlargement methods for computing the inverse matrix. *The annals of mathematical statistics*, pages 336–343, 1946.
- [GW<sup>+</sup>18] Rafael C Gonzalez, Richard E Woods, et al. Digital image processing, 2018.
- [HC71] John M Hammersley and Peter Clifford. Markov fields on finite graphs and lattices. *Unpublished manuscript*, 46, 1971.
- [HYL08] Zhilan Hu, Hong Yan, and Xinggang Lin. Clothing segmentation using foreground and background estimation based on the constrained delaunay triangulation. *Pattern Recognition*, 41(5):1581–1592, 2008.
- [Isi25] Ernst Ising. Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik*, 31(1):253–258, 1925.
- [JEPE05] Robert Jenssen, Deniz Erdogmus, Jose Principe, and Torbjorn Eltoft. The laplacian PDF distance: A cost function for clustering in a kernel feature space. In *Advances in Neural Information Processing Systems*, pages 625–632, 2005.



- [KRD11] Abhishek Kumar, Piyush Rai, and Hal Daume. Co-regularized multi-view spectral clustering. In *Advances in neural information processing systems*, pages 1413–1421, 2011.
- [LC10] Frank Lin and William W Cohen. Power iteration clustering. In *ICML*, 2010.
- [Li09] Stan Z Li. *Markov random field modeling in image analysis*. Springer Science & Business Media, 2009.
- [LY15] Guanbin Li and Yizhou Yu. Visual saliency based on multiscale deep features. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5455–5463, 2015.
- [LY16] Guanbin Li and Yizhou Yu. Deep contrast learning for salient object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 478–487, 2016.
- [MBLS01] Jitendra Malik, Serge Belongie, Thomas Leung, and Jianbo Shi. Contour and texture analysis for image segmentation. *International journal of computer vision*, 43(1):7–27, 2001.
- [MCPTVG18] Kevis-Kokitsi Maninis, Sergi Caelles, Jordi Pont-Tuset, and Luc Van Gool. Deep extreme cut: From extreme points to object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 616–625, 2018.
- [MFM04] David R Martin, Charless C Fowlkes, and Jitendra Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE transactions on pattern analysis and machine intelligence*, 26(5):530–549, 2004.
- [MFTM01] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *International Conference on Computer Vision*, volume 2, pages 416–423, July 2001.

- [MGTH18] Pedro Mercado, Antoine Gautier, Francesco Tudisco, and Matthias Hein. The power mean laplacian for multilayer graph clustering. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84, pages 1828–1838, 2018.
- [MMF<sup>+</sup>14] Michael T McCann, Dustin G Mixon, Matthew C Fickus, Carlos A Castro, John A Ozolek, and Jelena Kovacević. Images as occlusions of textures: A framework for segmentation. *IEEE transactions on image processing*, 23(5):2033–2046, 2014.
- [MOV12] Michael W Mahoney, Lorenzo Orecchia, and Nisheeth K Vishnoi. A local spectral method for graphs: With applications to improving graph partitions and exploring data graphs locally. *The Journal of Machine Learning Research*, 13(1):2339–2365, 2012.
- [MS89a] David Mumford and Jayant Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on pure and applied mathematics*, 42(5):577–685, 1989.
- [MS89b] David Bryant Mumford and Jayant Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on pure and applied mathematics*, 1989.
- [MS01] Marina Meila and Jianbo Shi. Learning segmentation by random walks. In *Advances in Neural Information Processing Systems*, pages 873–879, 2001.
- [MSV95] Ravi Malladi, James A Sethian, and Baba C Vemuri. Shape modeling with front propagation: A level set approach. *IEEE transactions on pattern analysis and machine intelligence*, 17(2):158–175, 1995.
- [MVM11] Subhransu Maji, Nisheeth K Vishnoi, and Jitendra Malik. Biased normalized cuts. In *CVPR 2011*, pages 2057–2064. IEEE, 2011.
- [NBCE09] Kangyu Ni, Xavier Bresson, Tony Chan, and Selim Esedoglu. Local histogram based segmentation using the wasserstein distance. *International Journal of Computer Vision*, 84(1):97–111, 2009.

- [NJW01] Andrew Ng, Michael Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 14:849–856, 2001.
- [NJW02] Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, pages 849–856, 2002.
- [P<sup>+</sup>98] Patrick Perez et al. *Markov random fields and images*. IRISA, 1998.
- [PF98] Pietro Perona and William Freeman. A factorization approach to grouping. In *European Conference on Computer Vision*, pages 655–670. Springer, 1998.
- [Por05] Fatih Porikli. Integral histogram: A fast way to extract histograms in cartesian spaces. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 829–836. IEEE, 2005.
- [RKB04] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "Grabcut" interactive foreground extraction using iterated graph cuts. *ACM transactions on graphics (TOG)*, 23(3):309–314, 2004.
- [RRB17] Matteo Ruffini, Guillaume Rabusseau, and Borja Balle. Hierarchical methods of moments. In *Advances in Neural Information Processing Systems*, pages 1901–1911, 2017.
- [Set99] James Albert Sethian. *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, volume 3. Cambridge university press, 1999.
- [SM00] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.
- [SWCX17] Cunzhao Shi, Yu Wang, Wang Chunheng, and Baihua Xiao. Ground-based cloud detection using graph model built upon superpixels. *IEEE Geoscience and Remote Sensing Letters*, 14(5):719–723, 2017.

- [TAB14] Meng Tang, Ismail Ben Ayed, and Yuri Boykov. Pseudo-bound optimization for binary energies. In *European Conference on Computer Vision*, pages 691–707, 2014.
- [TBAMB15] Meng Tang, Ismail Ben Ayed, Dmitrii Marin, and Yuri Boykov. Secrets of grabcut and kernel k-means. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1555–1563, 2015.
- [TDP<sup>+</sup>18] Meng Tang, Abdelaziz Djelouah, Federico Perazzi, Yuri Boykov, and Christopher Schroers. Normalized cut loss for weakly-supervised cnn segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1818–1827, 2018.
- [TGVB13] Meng Tang, Lena Gorelick, Olga Veksler, and Yuri Boykov. Grabcut in one cut. In *IEEE International Conference on Computer Vision*, pages 1769–1776, 2013.
- [TLRY15] Na Tong, Huchuan Lu, Xiang Ruan, and Ming-Hsuan Yang. Salient object detection via bootstrap learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1884–1892, 2015.
- [TM98] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *Sixth international conference on computer vision (IEEE Cat. No. 98CH36271)*, pages 839–846. IEEE, 1998.
- [TMAB16] Meng Tang, Dmitrii Marin, Ismail Ben Ayed, and Yuri Boykov. Normalized cut meets mrf. In *European Conference on Computer Vision*, pages 748–765. Springer, 2016.
- [VKR09] Sara Vicente, Vladimir Kolmogorov, and Carsten Rother. Joint optimization of segmentation and appearance models. In *IEEE International Conference on Computer Vision*, pages 755–762, 2009.
- [vR79] CJ van Rijsbergen. Information retrieval, 2nd edbutterworths, 1979.

- [WS01] Song Wang and Jeffrey Mark Siskind. Image segmentation with minimum mean cut. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 1, pages 517–524. IEEE, 2001.
- [WS03] Song Wang and Jeffrey Mark Siskind. Image segmentation with ratio cut. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(6):675–690, 2003.
- [WS11] David P Williamson and David B Shmoys. *The design of approximation algorithms*. Cambridge University press, 2011.
- [WWL<sup>+</sup>18] Linzhao Wang, Lijun Wang, Huchuan Lu, Pingping Zhang, and Xiang Ruan. Salient object detection with recurrent fully convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1734–1746, 2018.
- [XPC<sup>+</sup>16] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas S Huang. Deep interactive object selection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 373–381, 2016.
- [XPC<sup>+</sup>17] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas Huang. Deep grabcut for object selection. *arXiv preprint arXiv:1707.00243*, 2017.
- [YLM<sup>+</sup>09] J. Yang, W. Lu, Y. Ma, W. Yao, and Q. Li. An automatic ground based cloud detection method based on adaptive threshold. *Journal of Applied Meteorological Science*, 20:713–721, 2009.
- [YLM<sup>+</sup>10] J. Yang, W. Lu, Y. Ma, W. Yao, and Q. Li. An automatic ground-based cloud detection method based on the local threshold interpolation. *Acta Meteorologica Sinica*, 68:1007–1017, 2010.
- [YS01] Stella X Yu and Jianbo Shi. Grouping with bias. In *NIPS*, pages 1327–1334, 2001.
- [YS03] Stella X Yu and Jianbo Shi. Multiclass spectral clustering. In *ICCV*, pages 313–319, 2003.

- [YWC15] Jiangye Yuan, Deliang Wang, and Anil M Cheriyyadat. Factorization-based texture segmentation. *IEEE Transactions on Image Processing*, 24(11):3488–3497, 2015.
- [YXSJ13] Qiong Yan, Li Xu, Jianping Shi, and Jiaya Jia. Hierarchical saliency detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1155–1162, 2013.
- [ZB07] Dengyong Zhou and Christopher JC Burges. Spectral clustering and transductive learning with multiple views. In *Proceedings of the 24th international conference on Machine learning*, pages 1159–1166, 2007.
- [ZLWS14] Wangjiang Zhu, Shuang Liang, Yichen Wei, and Jian Sun. Saliency optimization from robust background detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2814–2821, 2014.
- [ZZLY18] Linlin Zong, Xianchao Zhang, Xinyue Liu, and Hong Yu. Weighted multi-view spectral clustering based on spectral perturbation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.