저작자표시-비영리-동일조건변경허락 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.
- 이차적 저작물을 작성할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

동일조건변경허락. 귀하가 이 저작물을 개작, 변형 또는 가공했을 경우에는, 이 저작물을 동일한 이용허락조건하에서만 배포할 수 있습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer
Image Segmentation:
Region and Boundary Cue Integration
Using Semi-Supervised Learning

Ph.D. Dissertation

Kim, Tae Hoon
School of Electrical Engineering and Computer Science
Seoul National University
Abstract

In this dissertation, we propose a new segmentation framework to efficiently integrate region and boundary cues using semi-supervised learning (SSL). Following this concept, we present two generative models for interactive segmentation and one spectral segmentation for unsupervised segmentation.

We first consider the problem of multi-label, interactive segmentation when a set of scribbles labeling the pixels is given. In contrast to most existing algorithms which deal with the inter-label discrimination, we address the problem of finding the generative model for each label. Particularly, in the generative image segmentation, two likelihood models based on SSL are introduced.

In Chapter 2, the likelihood that an unlabeled pixel has a specific label is defined as the relevance score of the unlabeled pixel with respect to the seeded pixels in the scribbles with that label. Here, this relevance score corresponds to the steady-state probability that a particle starting from the seeded pixels stays at the unlabeled pixel until convergence, computed by Random Walks with Restart (RWR), one of the SSL techniques. Since the steady-state probability considers the whole relationship between the unlabeled pixel and the seeded pixels, our RWR-based segmentation algorithm produces very good results under two difficult problems: the weak boundary problem and the texture problem. To improve the performance under the weak boundary problem, data-driven RWR (dRWR) that incorporates the edgeness
of each pixel into its restarting probability is designed. The consistency of relevance scores in the edge-bounded areas can be more emphasized by dRWR. Additionally, in order to reduce the dependency on the seed quantity and placement, we devise higher-order RWR (hRWR) that computes the steady-state probabilities in a bilayer graph whose nodes consists of the pixels and the over-segmented regions, generated by a unsupervised image segmentation algorithm such as Mean Shift. We can achieve the higher-order constraint that the pixels in the regions tend to have similar relevance scores by hRWR. By combining the ideas of dRWR and hRWR, we finally complete global RWR (gRWR). Experimental results with synthetic and natural images demonstrate the relevance and accuracy of our RWR-based segmentation algorithm.

In Chapter 3, we propose a novel likelihood model to strengthen the higher-order constraint in the segmentation algorithm. Although we can conceptually add the higher-order constraint in the RWR-based segmentation algorithm, it is difficult to prove the higher-order effect theoretically. And the higher-order effect is actually limited for segmentation of natural images. Therefore, to learn the likelihoods from labeled and unlabeled pixels, we design a new higher-order formulation additionally imposing the soft label consistency constraint whereby the pixels in the over-segmented regions tend to have the same label. In contrast with previous works which focus on the parametric model of the higher-order cliques for adding this soft constraint, we address a nonparametric learning technique to recursively estimate the region likelihoods as higher-order cues from the resulting likelihoods of pixels included in the regions. Therefore, the main idea of our algorithm is to design several quadratic cost functions of pixel and region likelihoods, that are supplementary
to each other, in a proposed multilayer graph and to estimate them simultaneously by a simple optimization technique. In this manner, we consider long-range connections between the regions that facilitate propagation of local grouping cues across larger image areas. Also, as abundant region candidates extracted by multiple oversegmentations are used, boundary cue can be implicitly imposed. The experiments on challenging data sets show that integration of higher-order cues in the multilayer graph quantitatively and qualitatively improves the segmentation results with detailed boundaries and reduces sensitivity with respect to seed quantity and placement.

We then consider a new unsupervised segmentation algorithm without user interaction. Spectral segmentation which uses the global information embedded in the spectrum of a given image’s affinity matrix is a major trend in image segmentation. Its overall quality mainly depends on how the affinity matrix is designed. Here, a novel affinity model based on SSL is introduced.

In Chapter 4, we address the problem of efficiently learning a full range of pairwise affinities gained by integrating region and boundary cues for spectral segmentation. We first construct a sparse multilayer graph whose nodes are both the pixels and the over-segmented regions. By applying the SSL strategy to this graph, the intra and interlayer affinities between all pairs of nodes can be estimated without iteration. These pairwise affinities are then applied into the spectral segmentation algorithms. In this work, two types of spectral segmentation algorithms are introduced: $K$-way segmentation and hierarchical segmentation. The former is to cluster all pixels and regions simultaneously into the $K$ visually coherent groups across all layers in a single multilayer framework of Normalized Cuts. The latter is to generate a hierarchy
of regions from contour information, obtained from spectral analysis of our affinity matrix, using a sequence of two transformations: Oriented Watershed Transform and Ultrametric Contour Map. Our algorithms provide high-quality segmentations which preserve object details by directly incorporating the full-range connections. Moreover, since our full affinity matrix is defined by the inverse of a sparse matrix, its eigen-decomposition can be efficiently computed. The experimental results on the BSDS and MSRC image databases demonstrate the superiority of our segmentation algorithms in terms of relevance and accuracy compared with existing popular methods.

**Key words:** Interactive segmentation, unsupervised segmentation, spectral segmentation, hierarchical segmentation, semi-supervised learning, random walks with restart, likelihood estimation, multilayer graph construction, region and boundary cue integration.

**Student number:** 2006-21186
# Contents

Abstract i

Contents vi

List of Figures xi

List of Tables xxiii

1 Introduction 1

1.1 Background and Research Issues ...................................... 1
1.2 Outline of the Dissertation ............................................ 5

2 RWR-Based Interactive Segmentation 9

2.1 Introduction .............................................................. 9
2.2 Proposed Generative Model ............................................. 13
2.3 RWR-Based Likelihood Estimation .................................... 16
  2.3.1 Basic RWR ....................................................... 16
  2.3.2 Using Boundary Cue: Data-Driven RWR ....................... 21
  2.3.3 Using Higher-Order Cue: Higher-Order RWR ................. 23
  2.3.4 Combining Boundary and Higher-Order Cues: Global RWR . 29
# CONTENTS

2.4 Experimental Results ............................................. 31
   2.4.1 Weak Boundary Problem ................................ 31
   2.4.2 Texture Problem ........................................... 33
   2.4.3 Quantitative comparisons ................................. 34
2.5 Conclusion ....................................................... 37

3 Robust Higher-Order Interactive Segmentation ................. 39
   3.1 Introduction .................................................. 39
   3.2 Generative Model for Segmentation ........................ 42
   3.3 Proposed Likelihood Estimation ............................. 46
      3.3.1 Basic Function ......................................... 46
      3.3.2 Incorporating Higher-Order Cues ...................... 47
      3.3.3 Using Multiple Over-Segmentations .................. 54
   3.4 Experimental Results ........................................ 57
      3.4.1 Parameter Settings ...................................... 58
      3.4.2 Segmentation Results ................................... 59
      3.4.3 Complexity Consideration .............................. 65
   3.5 Conclusion .................................................... 65

4 Full Affinity Model for Spectral Segmentation ............... 67
   4.1 Introduction ................................................ 67
   4.2 Previous Affinity Models .................................... 72
   4.3 Proposed Full Affinity Model .............................. 74
      4.3.1 Graph Design .......................................... 75
      4.3.2 Learning Full Affinities ............................... 76
      4.3.3 Spectral Analysis ..................................... 78
CONTENTS

4.4 Spectral Segmentation ........................................... 81
  4.4.1 K-way Segmentation ......................................... 83
  4.4.2 Hierarchical Segmentation ................................. 85
4.5 Experimental Results ............................................. 86
  4.5.1 Parameter Setting ............................................ 86
  4.5.2 Measurements ................................................ 90
  4.5.3 Results ....................................................... 91
4.6 Conclusion ..................................................... 101

5 Conclusion .......................................................... 103

Bibliography .......................................................... 107
List of Figures

2.1 Introducing interactive segmentation. Given the input images with the seeds (“Red” and “Green”) in (a), the segmentation results are obtained by (b) GC [1], (c) GD [2], (d) RW [3], and (e) Our algorithm, respectively. ................................................. 10

2.2 An example of the generative segmentation with just one label. Given the seeds with green initial label in (a), the likelihood map, where red represents greater, in (b) is computed using our algorithm. (c) is the resulting segmentation with a threshold level $10^{-8}$. The foreground label is assigned to the pixels with probability above the threshold. . 11

2.3 Overview of our proposed segmentation algorithm. Given the seeds in (b), the posterior probabilities (d),(e) and (f) are obtained by computing Eq. (2.1) for “red”, “green”, and “blue” labels, respectively. The segmentation result (c) is obtained by assigning each pixel the label having maximum posterior probability. ..................... 15
2.4 Illustration of RWR. Given an input image, we construct a graph $G_X$ whose nodes are a set of pixels $X$ (green circle) in (a). In the graph $G_X$, RWR has two types of particle movements: (b) “Transition” (violet line) and (c) “Restart” (violet dotted line), at each pixel $x_i$. 

2.5 Comparison of the relevance scores $\vec{u}_+$ with respect to the variation of the restarting probability $c$ in RWR. (a) Synthetic images with one pixel-seed $+$. (b),(c), and (d) Relevance scores $\vec{u}_+$, estimated by RWR with $c = 10^{-4}$, $c = 10^{-5}$, and $c = 10^{-6}$, respectively (log scale).

2.6 Example segmentations by RWR in synthetic images. (a) Input images with one pixel-seed selected for each color. (b), (c), and (d) Relevance scores $\vec{u}_+$, $\vec{u}_+$, and $\vec{u}_+$, where red represents a greater score, for “red”, “green”, and “blue” labels, respectively. (e) Segmentation results, where labels are represented by red, green, and blue colors, respectively.

2.7 Comparison of the relevance scores $\vec{u}_+$ with respect to the variation of the parameter $\mu$ in dRWR. (a) Synthetic image with one pixel-seed $+$. (b),(c), and (d) Relevance scores $\vec{u}_+$, estimated by dRWR with $\mu = c \approx 9.8 \times 10^{-6}$, $\mu = 10^{-2}$, and $\mu = 1$, respectively.

2.8 Example segmentations by dRWR in Fig. 2.6(a). (a) Edge maps. (b), (c), and (d) Relevance scores $\vec{u}_+$, $\vec{u}_+$, and $\vec{u}_+$ for “red”, “green”, and “blue” labels, respectively. (e) Segmentation results. In (e), the labels of pixels in the area with black color are undecidable.
2.9 Illustration of hRWR. In the bilayer graph $G$, the nodes are the pixels $\mathcal{X}$ (green circle) and the regions $\mathcal{Y}$ (violet circle). The regions $\mathcal{Y}$, whose boundaries are drawn in red color overlaid on the original image, are generated by Mean Shift [4] in (a). (b) shows the edges connected to one region with blue lines. In $G$, hRWR has (c) “Jump” move (brown dotted lines) as well as “transition” and “restart” moves at each layer. ................................................................. 25

2.10 Comparison of the relevance scores $\vec{u}+$ with respect to the variation of the parameter $\gamma$ in hRWR. (a) Synthetic image with one pixel-seed $+$. (b),(c), and (d) Relevance scores $\vec{u}+$, estimated by hRWR with $\gamma = 10^{-1}$, $\gamma = 10^{-2}$, and $\gamma = 10^{-4}$, respectively. .................................................. 28

2.11 Example segmentations by hRWR in Fig. 2.6(a). (a) Over-segmentations. (b), (c), and (d) Relevance scores $\vec{u}+$, $\vec{u}+$, and $\vec{u}+$ for “red”, “green”, and “blue” labels, respectively. (e) Segmentation results. ............... 28

2.12 Example segmentations by gRWR in Fig. 2.6(a). (a), (b), and (c) Relevance scores $\vec{u}+$, $\vec{u}+$, and $\vec{u}+$ for “red”, “green”, and “blue” labels, respectively. (d) Segmentation results. ............................... 30

2.13 Comparison of RWR-based segmentation in real images. (a) Input images with scribbles. (b) Edges (black color) + Region boundaries (red color). (c), (d), (e), and (f) Segmentation results by RWR, dRWR, hRWR, and gRWR, respectively. ................................. 31
2.14 Comparison of our algorithm with GC and RW for finding weak boundaries. Given two labels ("green" and "blue") and original images in (a) (Top: image created with a black circle with four erased sections. Bottom: image created with a $3 \times 3$ black grid with four erased sections.), (b),(c) and (d) are the segmentation results of GC, RW and our algorithm respectively. 

2.15 Comparison of our algorithm with GC, GD, and RW on synthetic textured images. (b),(c),(d) and (e) are the segmentation results of GC, GD, RW, and our algorithm, respectively.

2.16 Comparison of our algorithm with GC and RW on natural images. (b),(c) and (d) are the segmentation results of GC, RW, and our algorithm, respectively. $a_o$ is the accurate rate in (2.27). (e) Ground-truth segmentations.

2.17 Example segmentations using the tri-maps on the Microsoft GrabCut database. (a) Test image with tri-map. Segmentation results by (b) GC [5], (c) RW [3], and (e) Our algorithm gRWR.

3.1 Using the higher-order cues for interactive segmentation. The first row shows an image with one pixel-seed selected for each label. (b)-(e) Segmentation results by RW [3] with pairwise potentials, gRWR [6] with higher-order constraint, Robust $P^n$ model [7] with parametric higher-order potentials, and our method with nonparametric higher-order cues, respectively.
3.2 Example segmentations using the GMM-based likelihood maps. (a) Input images with the seeds selected for each label. (b) and (c) Likelihoods estimated by GMM for “red” and “green” labels, respectively. (d) Segmentation results by assigning each pixel the label with the largest likelihood. (e) Refined segmentation results by adding the label smoothness constraint in the prior distribution (optimized by graph cuts). 

3.3 Example RWR-based segmentations in Fig. 3.2(a). (a) Edges (black color) + Region boundaries (red color). (b) and (c) Likelihoods estimated by gRWR which use the edgeness and the over-segmented regions in (a), where red represents a greater likelihood, for “red” and “green” labels, respectively (log scale). (d) Segmentation results by assigning each pixel the label with the largest likelihood.

3.4 Example segmentations by solving Eq. (3.6) in natural images. (a) Input images with the seeds selected for each label. (b) and (c) Likelihoods \( \{\hat{q}_k\} \) for “red” and “green” labels, respectively. (e) Segmentation results by assigning each pixel the label with the largest likelihood.
3.5 An illustration of the proposed graph $G$. In $\mathcal{V}$, a node denotes a pixel $x_i \in \mathcal{X}$ (green circle) or a region $y_n \in \mathcal{Y}$ (violet circle). The regions $\mathcal{Y}$, whose boundaries are drawn in red color overlaid on the original image, are generated by an unsupervised image segmentation algorithm in (a). The undirected edges $\mathcal{E}$ represent the pairwise relationship between the nodes. (b) shows the edges connected to one region with violet lines. The edge weights are defined by the color differences between two nodes as in (2.4), (2.16), and (3.7) according to the node types.  

3.6 Example segmentations by solving Eq. (3.14) with one over-segmentation of each image in Fig. 3.4(a). (a) Over-segmentations. (b) and (c) Likelihoods $\{\vec{q}_k\}$ for “red” and “green” labels, respectively. (e) Segmentation results by assigning each pixel the label with the largest likelihood. 

3.7 Multilayer graph using multiple over-segmentations. A set of regions $\mathcal{Y}$ consists of the sum of the regions obtained from multiple over-segmentations in (a). (b) and (c) show the edges connected to one same region and one same pixel with violet lines, respectively. 

3.8 Example segmentations by solving Eq. (3.14) with three over-segmentations in Fig. 3.4(a). (a) Multiple over-segmentations, where red represents larger probability that the edge is a part of object boundary. (b) and (c) Likelihoods $\{\vec{q}_k\}$ for “red” and “green” labels, respectively. (e) Segmentation results by assigning each pixel the label with the largest likelihood.
3.9 An example of multiple over-segmentations. (a) shows an input image with scribbles with two red and green initial labels. (b)-(d) represent multiple over-segmentations, generated by mean shift segmentation algorithm with different parameters \((h_s, h_r)\). (e)-(g) give the resulting segmentations based on only one over-segmentation (b)-(d), respectively. (h) shows our final segmentation with all regions (b)-(d).

3.10 Examples of the segmentations with respect to the variation of \(\kappa\) in (3.8). From the images (a) with scribbles with two red and green labels, (b)-(d) give the resulting segmentations according to \(\kappa\). \((\epsilon = 0.2)\)

3.11 Examples of the segmentations with respect to the variation of \(\epsilon\) in (3.10). From the images (a) with scribbles with two red and green labels, (b)-(d) give the resulting segmentations according to \(\epsilon\). \((\kappa = 0.002)\)

3.12 Example segmentations using the tri-maps on the Microsoft GrabCut database. (a) Test image with tri-map. Segmentation results by (b) GMM-GC [5], (c) gRWR [6], (d) Robust \(P^n\) model [7], and (e) Our algorithm.

3.13 Description of the sensitivity check with respect to seed quantity and placement. (a) Test image with an initial tri-map (b). (c) Results of the sensitivity check by varying the perturbed seed quantity: 1%, 10%, 30%, 50% of total seed quantity. (d)-(g) Standard segmentations produced from the initial tri-map, and (h)-(k) Perturbed segmentations produced from the randomly chosen 1% seeds by GC, RW, Robust \(P^n\) Model, and our algorithm respectively.
3.14 Comparison of our algorithm with the state-of-the-art algorithms for finding multiple patterns or objects with a few scribbles. (a) Input images with scribbles with two red and green labels. Segmentation results by (b) RW [3] with pairwise potentials, (c) Robust $P^n$ model [7] with parametric higher-order potentials, and (d) Our algorithm. 63

3.15 Visual comparison of our algorithm with the state-of-the-art algorithms in natural images. (a) Inputs with scribbles with two red and green labels. Segmentation results by (b) RW [3], (c) Robust $P^n$ model [7], and (d) Our algorithm. 64

4.1 Learning pairwise affinities. (a) Image with a user-given pixel point $+$. (b)-(d) Affinity images between $+$ and other pixels in the image, where red represents a greater affinity. These affinities are obtained by the general color+boundary affinity model used in MNCut [8], our proposed model, and the ground-truth affinity function from human annotations, respectively. 69

4.2 Introducing $K$-way segmentation. (a) Original image. (b)-(d) $K$-way partitioning by NCut [9], MNCut [8], and our algorithm FNCut with the boundaries drawn in red, respectively. 70

4.3 Introducing hierarchical segmentation. (a)-(c) Hierarchical segmentations of an input image in Fig. 4.2(a) by Canny-OWT-UCM, gPb-OWT-UCM [10], and our algorithm fPb-OWT-UCM, respectively. (d) Human boundaries. Darker color means greater edgeness. 70
4.4 Examples of affinity images by the previous affinity models. (a) Image with a user-given pixel (+). (b)-(d) Affinities between + and all other pixels by Eq. (4.1)-(4.3), respectively. 72

4.5 Our multilayer graph $G = (V, E)$. In (a), the graph nodes $V$ consist of pixels $X$ and regions $Y = \{Y_1 \cup ... \cup Y_L\}$. The undirected edges $E$ represent the relationship between pairs of nodes. (b) and (c) show the examples of edges (violet lines) connected to one pixel and to one region, respectively. 74

4.6 Examples of our affinity images using SSL [11]. (a) Image with one pixel + selected. (b)-(d) Three over-segmentations, obtained by varying the MShift parameters. (e)-(h) and (i)-(l) Affinity images $\vec{\pi}_+$ by $\psi_{no}$ in (4.7) and by $\psi_{un}$ in (4.9) with none, one (b), two (b)-(c), and three (b)-(d) over-segmentations, respectively. 79

4.7 Comparison of spectral analysis with respect to the sparse matrix $\Omega$ and our full affinity matrix $\Pi$. (a) Test images. (b) The average running times to solve the generalized eigenvalue problem in (4.12) by MATLAB 7.13 on a quad-core 3.3GHz desktop as a function of the number of pixels ($K = 5$). (c)-(d) and (e)-(f) The 2-3th smallest eigenvectors in (4.12) based on $\Omega$ and $\Pi$, respectively. 82

4.8 Overview of our $K$-way segmentation algorithm FNCut. (a) Test image. (b)-(e) The four smallest eigenvectors $\{\vec{v}_1, ..., \vec{v}_4\}$ in (4.12). The first eigenvector in (b) is very close to a constant vector. (f)-(h) The $K$-way segmentation results ($K = 2, 3, and 4$). 83
LIST OF FIGURES

4.9 Example of multilayer segmentation by FNCut. (a) Test image. (b)-(d) The three over-segmentations by varying the parameters of the MShift algorithm. (e)-(h) Our K-way segmentations at each layer (a)-(d), respectively ($K = 20$).

4.10 Overview of our hierarchical segmentation algorithm fPb-OWT-UCM. (a) Test image. (b) Maximal response of contour signals by fPb in (4.17). (c) Region tree in the UCM. (d)-(h) Segmentations produced by thresholding the UCM in (c) at level 0.01, 0.05, 0.1, 0.3, and 0.5, respectively.

4.11 Comparison of over-segmentations by GBIS [12] and MShift [4] for spectral segmentation. (a) Test images. (b)-(d) and (g)-(i) The three different over-segmentations by GBIS and MShift, respectively. (e) and (j) K-way segmentations using the regions in (b)-(d) and (g)-(i), respectively (Top: $K = 10$, Bottom: $K = 20$). (f) and (k) Hierarchical segmentations using the regions in (b)-(d) and (g)-(i), respectively.

4.12 Examples of our segmentation results using different region-based layers $\mathcal{Y}$ produced by MShift in the multilayer graph $G$. (a) Test images. (b) Ground-truth boundaries. (c)-(e) and (f)-(h) The three over-segmentations in accordance with the different sets of the MShift parameters $(h_s, h_r) = \{(5, 7), (10, 15), (15, 25)\}$ and $(h_s, h_r) = \{(5, 7), (7, 5), (7, 7)\}$, respectively. (i)-(k) and (m)-(o) 20/30/40-way segmentations using the region-based layers (c)-(e) and (f)-(h), respectively. (l) and (p) Hierarchical segmentations using the region-based layers (c)-(e) and (f)-(h), respectively.
4.13 Examples of our affinity images with respect to the variation of parameters $\tau$ (top row, $c = 10^{-5}$) and $c$ (bottom row, $\tau = 10^{-3}$) in (4.6) and (4.11), respectively. ............................... 89

4.14 Boundary benchmarks on the BSDS300 (Left) and BSDS500 (Right) datasets. In the box, several segmentation approaches are ranked according to their maximum F score in (4.18) with respect to human ground-truth boundaries. Iso-F curves are shown in green. Average agreement between human subjects is indicated by the green dot. . . . 93

4.15 Example segmentations by our algorithms: FNCut, fPb-OWT-UCM, and cPb-OWT-UCM. (a) Test images. (b), (d) and (f) Segmentation results by FNCut, fPb-OWT-UCM, and cPb-OWT-UCM at the optimal dataset scale (ODS) in terms of F in (4.18), respectively. (c) and (e) UCMs produced by fPb-OWT-UCM and cPb-OWT-UCM, respectively. (g) Human boundaries in multiple ground-truth segmentations. 94

4.16 Statistics of our $K$-way segmentation method FNCut on Covering, PRI, and VI over (a) BSDS300 dataset and (b) BSDS500 dataset with $K = \{5, 10, ..., 40\}$, compared with the general spectral methods NCut and MNCut. ................................. 95

4.17 Visual comparison of our algorithm FNCut with the general $K$-way spectral methods NCut [9] and MNCut [8] on the MSRC database. (a) Test images. (b)-(d) Segmentation results by NCut, MNCut, and FNCut, respectively. ................................. 96
4.18 Visual comparison of our algorithms fPb-OWT-UCM and cPb-OWT-UCM with the state-of-the-art hierarchical segmentation method gPb-OWT-UCM [13] on the BSDS500 dataset. (a) Test images. (b), (d), and (f) UCMs produced by gPb-OWT-UCM, fPb-OWT-UCM, and cPb-OWT-UCM, respectively. (c), (e), and (g) Segmentation results by gPb-OWT-UCM, fPb-OWT-UCM, and cPb-OWT-UCM at the optimal dataset scale (ODS) in terms of the Covering measure in (4.20), respectively. (h) Human boundaries.
List of Tables

2.1 Segmentation accuracy $a_o$ in the Microsoft GrabCut database. . . . 36

3.1 Segmentation accuracy $a_o$ in the Microsoft GrabCut database. . . . 60

4.1 Boundary benchmarks on the BSDS300 and BSDS500 datasets. Results for ten different segmentation algorithms are given. The F scores when choosing an optimal scale for the entire dataset (ODS) or per image (OIS), as well as the average precision (AP). The best three results are highlighted in colors: red, green, and blue in descending order. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 92

4.2 Region benchmarks on the BSDS300 dataset. sPb is the boundary detector based on spectral analysis of sparse affinity matrix such as the weight matrix $\Omega$. . . . . . . . . . . . . . . . . . . . . . . . . . . 98

4.3 Region benchmarks on the BSDS500 dataset. . . . . . . . . . . . . . 99
Chapter 1

Introduction

1.1 Background and Research Issues

Image segmentation is a fundamental process in many image, video, and computer vision applications [14, 15]. It is often used to partition an entire image, such that each partition belongs to a meaningful object or its part. It is a critical step towards object recognition and image understanding. Its application area varies from industrial quality control to medicine, robot navigation, geophysical exploration, military applications, etc. In all these areas, the quality of the final results depends largely on the quality of the segmentation.

The image segmentation problem is still an important research field, and many algorithms have been proposed in the literature (see the survey [16–18]). In general, one can distinguish unsupervised and interactive (or supervised) segmentation algorithms.

1) Unsupervised image segmentation is to cluster the pixels into meaningful groups without user interaction. The main research directions for this include mode-
seeking [4, 19], deterministic annealing [20], stochastic clustering [21, 22], mixture model [23, 24], rate distortion [25], graph-based model [9, 12, 26], contour-based model [27, 28], and other variational methods [29, 30]. In most researches, the image segmentation problem is described as assigning a label to every pixel in a specific globalization framework. However, the appearance-based local methods such as Mean Shift (MShift) [4] are still popular for obtaining over-segmented regions with detailed boundaries. Thus, combining advantages of local and global approaches is beneficial in image segmentation [7, 31]. The unsupervised segmentation algorithms require less skill from a user, since they do not use any user interaction. However, they still do not produce satisfactory results [32].

2) In order to do a semantically meaningful image segmentation, it is essential to take a priori information about the image into account. Such information, for example, can be provided by the user through a set of scribbles labeling the pixels in an image. This issue has been addressed in the literature as interactive image segmentation. Recently, several interactive segmentation approaches have been proposed. There are three types of interactive segmentation algorithms according to the user inputs. The first type is that the segmentation is obtained based on pieces of the desired boundary, such as the intelligent scissors [33]. The second type is that an initial boundary that is closed to the desired boundary is given, such as Active Contour [34] and Level Set [35]. Finally, the third type is that the user provides an initial labeling of some pixels. Nowadays one focuses on the supervised image segmentation of the third type. The main research directions for this are based on Graph Cuts (GC) [5, 36] to find the minimum cut between the foreground and background seeds via a maximum flow computation, Random Walker (RW) [3] to
determine the labels via the seed propagation on a weighted graph, and Shortest Path (GD) \([2,37]\) to assign each pixel the foreground label if there is a shorter path from that pixel to a foreground seed than to any background seed. Recently these algorithms were all placed into a common framework that allows them to be seen as instances of a more general seeded segmentation algorithm \([38-40]\). However, since the user typically needs to specify the scribbles on the image to achieve a desirable result, it is difficult for any user to provide the appropriate scribbles in a consistent and perceptually coherent manner.

In natural images, many problems make segmentation difficult, such as determining faint object boundaries, separating highly textured areas, detecting repeated patterns, and recognizing elongated objects. Also, in interactive segmentation, the segmentation quality should be less dependent on the scribble quantity and placement for reducing user effort. As a popular way to solve these difficulties, many segmentation algorithms use two basic cues of the pixels in relation to their local neighborhood: region and boundary cues \([41]\). Region and boundary cues provide the continuity of the pixels whose attributes such as color and texture are similar and the discontinuity of the pixels across the boundaries, respectively. However, since the region-based algorithms always provide closed contour regions, two different parts whose attributes differ by only a small amount may be mistaken as a same region. Also, in the boundary-based algorithms, if an image is noisy, edge detection may result in spurious edges. Therefore, both the region- and boundary-based segmentation algorithms often fail to produce accurate segmentation \([42]\). With the aim of improving the segmentation, a large number of new algorithms which integrate region and boundary cues have been proposed over the last few years. The main characteristics of these algorithms are as follows.
1) First, after the initial segmentation is produced using region or boundary cue, post-processing is performed by referring another cue. For example, the regions generated by the region-based algorithm are merged in order to eliminate false boundaries which do not correspond with boundary cue [42, 43]. Similarly, suppressing or prolonging a boundary is decided according to region cue in the initial result obtained from the boundary-based approach [44, 45]. Also, a region-growing procedure is used to obtain an initial estimate of a target region, which is then combined with salient edge information to achieve a more accurate representation of the target boundary [46–48].

2) Second, the embedded integration strategy usually consists of using edge information, previously extracted, within a region-based segmentation algorithm. For example, boundary information is used to define a new decision criterion which controls the growth of the region [49, 50] or the seed points from which regions are grown [51–53]. The aim of this integration strategy is to use boundary cue as the means of avoiding the common problems of region-based segmentation algorithms.

3) Finally, boundary cue is implicitly added by considering the relationship between the pixels and the regions, previously extracted by multiple over-segmentations, in the region-based segmentation. In [7], using the parametric model to force label consistency in the regions can emphasize the correlation between the pixels inside the regions as higher-order cue. Also, the discontinuity between the over-segmented regions can provide implicit boundary information as boundary cue.

We follow the segmentation framework of third type that starts by generating the over-segmented regions. Unlike the previous parametric models [7], we try to nonparametrically integrate region and boundary cues using the graph-based semi-
supervised learning techniques [11] [54] that make use of both labeled and unlabeled data. We first construct a multilayer graph. In this graph, each layer consists of the pixels or over-segmented regions. Then, in order to compute the edge weight between any two nodes, their colors are used as the region cues. Also, by linking only between each pixel and its corresponding regions, the boundaries of the over-segmented regions are implicitly defined as the boundary cues. Finally, by semi-supervised learning in this graph, region and boundary cues can be propagated to a large image area. This propagation is helpful to consider intrinsic image structure for image segmentation.

1.2 Outline of the Dissertation

In this dissertation, we propose a new segmentation framework to efficiently integrate region and boundary cues using semi-supervised learning (SSL). Following this concept, we present two generative models for interactive segmentation and one spectral segmentation for unsupervised segmentation.

In Chapter 2 and 3, we consider the problem of multi-label, interactive segmentation when a set of scribbles labeling the pixels is given. In contrast to most existing algorithms which deal with the inter-label discrimination, we address the problem of finding the generative model for each label. Particularly, in the generative image segmentation, two likelihood models based on SSL are introduced.

In Chapter 2, the likelihood that an unlabeled pixel has a specific label is defined as the relevance score of the unlabeled pixel with respect to the seeded pixels in the scribbles with that label. Here, this relevance score corresponds to the steady-state probability that a particle starting from the seeded pixels stays at the unlabeled
pixel until convergence, computed by Random Walks with Restart (RWR), one of the SSL techniques. Since the steady-state probability considers the whole relationship between the unlabeled pixel and the seeded pixels, our RWR-based segmentation algorithm produces very good results under two difficult problems: the weak boundary problem and the texture problem. To improve the performance under the weak boundary problem, data-driven RWR (dRWR) that incorporates the edgeness of each pixel into its restarting probability is designed. The consistency of relevance scores in the edge-bounded areas can be more emphasized by dRWR. Additionally, in order to reduce the dependency on the seed quantity and placement, we devise higher-order RWR (hRWR) that computes the steady-state probabilities in a bilayer graph whose nodes consist of the pixels and the over-segmented regions, generated by an unsupervised image segmentation algorithm such as Mean Shift. We can achieve the higher-order constraint that the pixels in the regions tend to have similar relevance scores by hRWR. By combining the ideas of dRWR and hRWR, we finally complete global RWR (gRWR).

In Chapter 3, to strengthen the higher-order constraint in the estimated likelihoods, we propose a new higher-order formulation additionally imposing the soft label consistency constraint whereby the pixels in the over-segmented regions tend to have the same label. In contrast with previous works which focus on the parametric model of the higher-order cliques for adding this soft constraint, we address a nonparametric learning technique to recursively estimate the region likelihoods as higher-order cues from the resulting likelihoods of pixels included in the regions. Therefore, the main idea of our algorithm is to design several quadratic cost functions of pixel and region likelihoods, that are supplementary to each other, in a proposed multilayer
graph and to estimate them simultaneously by a simple optimization technique. In this manner, we consider long-range connections between the regions that facilitate propagation of local grouping cues across larger image areas. Also, as abundant region candidates extracted by multiple over-segmentations are used, boundary cue can be implicitly imposed.

In Chapter 4, we consider a new unsupervised segmentation algorithm without user interaction. Spectral segmentation which uses the global information embedded in the spectrum of a given image’s affinity matrix is a major trend in image segmentation. Its overall quality mainly depends on how the affinity matrix is designed. In this work, we address the problem of efficiently learning a full range of pairwise affinities gained by integrating region and boundary cues for spectral segmentation. We first construct a sparse multilayer graph whose nodes are both the pixels and the over-segmented regions. By applying the SSL strategy to this graph, the intra and interlayer affinities between all pairs of nodes can be estimated without iteration. These pairwise affinities are then applied into the spectral segmentation algorithms. In this work, two types of spectral segmentation algorithms are introduced: $K$-way segmentation and hierarchical segmentation. The former is to cluster all pixels and regions simultaneously into the $K$ visually coherent groups across all layers in a single multilayer framework of Normalized Cuts. The latter is to generate a hierarchy of regions from contour information, obtained from spectral analysis of our affinity matrix, using a sequence of two transformations: Oriented Watershed Transform and Ultrametric Contour Map. Our algorithms provide high-quality segmentations which preserve object details by directly incorporating the full-range connections. Moreover, since our full affinity matrix is defined by the inverse of a sparse matrix, its eigen-decomposition can be efficiently computed.
Finally, we conclude the dissertation in Chapter 5.
Chapter 2

RWR-Based Interactive Segmentation

2.1 Introduction

In this chapter, we deal with two important difficulties of the segmentation in natural images: the weak boundary problem and the texture problem. The first problem is to find the broken edges when they are parts of a consistent boundary. The second problem is to separate the textures in the highly cluttered image. In fact, such situations often arise in natural images. In these cases, the segmentations become ambiguous without user-provided inputs, and the interactive image segmentation algorithms are often preferred. Therefore, we address the interactive image segmentation problem.

Recently, several interactive segmentation approaches have been proposed. One of the popular approaches is the Graph Cuts method (GC) [1]. Since GC treats this minimum cut criterion, it often causes small cut problem when the contrast
Figure 2.1: Introducing interactive segmentation. Given the input images with the seeds (“Red” and “Green”) in (a), the segmentation results are obtained by (b) GC [1], (c) GD [2], (d) RW [3], and (e) Our algorithm, respectively.

is low or the number of seed pixels is small, as shown in the top image of Fig. 2.1(b). In [2], the interactive segmentation algorithm based on geodesic distance from the seeds (GD) was proposed. However, since it does not consider the global relationship between two pixels, it is not reliable to use the simple geodesic distance as the relevance measure between two pixels such as in Fig. 2.1(c). Particularly, when the weak boundary problem occurs in a uniform region, the shortest path distances from any pixel to both scribbles are exactly same, as shown in the top image of Fig. 2.1(c). Thus, in this case, GD does not work. Another approach is the Random Walker image segmentation algorithm (RW) proposed by Grady [3]. After the first arrival probability that a particle starting at a pixel first reaches one of the seeds with each label is computed, that pixel takes one label with maximum probability. It was shown in [3] that RW has better performance under difficult conditions than GC. However, the first arrival probability defined in [3] has some
2.1. INTRODUCTION

Figure 2.2: An example of the generative segmentation with just one label. Given the seeds with green initial label in (a), the likelihood map, where red represents greater, in (b) is computed using our algorithm. (c) is the resulting segmentation with a threshold level $10^{-8}$. The foreground label is assigned to the pixels with probability above the threshold.

limitations. Since the particle starting at a pixel must first arrive at the border of pre-labeled region, it only considers the local relationship between the pixel and that border. Therefore, the information of seeds inside the pre-labeled region is ignored in absence of higher-order interactions. Also, this probability depends on the number of seeds. If the seeds with only one label numerically grow under the weak boundary problem, the first arrival probability of that label is increased without regard to the whole relation between a pixel and the seeds. These limitations explain why RW still suffers from the two problems: the weak boundary problem and the texture problem such as in Fig. 2.1(d). Most recently, the segmentation approach defined by an $l_{\infty}$ norm was proposed in [38]. Since RW with constraints was used as a regularization method for yielding a unique solution, this approach still has the limitations of RW.

Most previous interactive image segmentation algorithms focus on the inter-label discrimination, not finding the generative model for each label. Although they tried to solve the weak boundary problem and the texture problem, these two problems in
natural images are still the most challenging issues in image segmentation. In this chapter, we propose a new generative image segmentation algorithm based on the Random Walks with Restart (RWR) [54] that can solve the weak boundary problem and texture problem effectively [6] [55]. The key contributions of our proposed algorithm are as follows.

1) We introduce a generative model for image segmentation. From basic decision theory [56], it is known that generative methods are better inference algorithms. In contrast to most existing models which focus on the inter-label discrimination, we address the problem of finding the generative model for each label like [57]. For example, we can consider just one-label segmentation problem as shown in Fig. 2.2. Our model can produce the segmentation result with an optimal threshold level as shown in Fig. 2.2(c). This is possible since the likelihood probability can be generated using our generative model as depicted in Fig. 2.2(b).

2) We design a generative image segmentation approach using the steady-state probability of RWR as a part of the likelihood term. Since the likelihood of a pixel is defined as the average of all the steady-state probabilities between that pixel and the seeds with same label, our algorithm can reduce dependence on the number of seeds under the weak boundary problem, such as in Fig. 2.1(f). RWR, similar to graph-based semi-supervised learning [11], is a very successful technique for defining the relevance relation between two nodes in graph mining [54, 58–60]. It has good performance on many other applications: Cross-model correlation discovery [54], Center-piece subgraph discovery [59], Content based image retrieval [58], Neighborhood formulation [60] and etc. Since this steady-state probability of RWR considers the whole relationship between two pixels, it naturally reflects the effects of texture.
3) We introduce the variants of RWR for estimating the likelihoods for each label. First, data-driven RWR (dRWR) that incorporates the edgeness of each pixel into its restarting probability is designed. Since the label consistency in the edge-bounded areas is emphasized in dRWR, partitioning an image containing the weak boundaries is possible. We then devise higher-order RWR (hRWR) that achieves the higher-order effects by constructing a bilayer graph whose nodes are the over-segmented regions as well as the pixels. Since the pixels inside each region tend to have same labels in hRWR, the segmentation results are less sensitive to the scribble properties: size and position. Finally, by combining the ideas of dRWR and hRWR, we complete global RWR (gRWR).

4) Under two challenging problems: the weak boundary problem and the texture problem, our algorithm produces very good segmentation results on synthetic and natural images. It has better performance than GC, GD, and RW.

This chapter is organized as follows. In Section 2.2, we introduce our proposed generative image segmentation algorithm. In Section 2.3, we then explain the RWR-based likelihood estimation in detail. The experimental results are shown in Section 2.4. Finally, we discuss our approach and give conclusions in Section 2.5.

### 2.2 Proposed Generative Model

Let us consider the image segmentation as a labeling problem in which each pixel $x_i \in \mathcal{X} = \{x_1, \ldots, x_N\}$ is to be assigned one label $l_k \in \mathcal{L} = \{l_1, \ldots, l_K\}$. From basic decision theory [56], we know that the most complete characterization of the solution is expressed in terms of the set of posterior probabilities $\{p(l_k|x_i)\}$. Once we know these probabilities, it is straightforward to assign $x_i$ the label having the
largest probability. In a generative approach, we model the joint probability $p(x_i, l_k)$ of pixels and labels. This can be done by computing the label prior probability $p(l_k)$ and the pixel likelihood $p(x_i|l_k)$ separately. The required posterior probability $p(l_k|x_i)$ is obtained using Bayesian rules:

$$p(l_k|x_i) = \frac{p(x_i|l_k)p(l_k)}{\sum_{k'=1}^{K} p(x_i|l_{k'})p(l_{k'})}, \quad (2.1)$$

where the sum in the denominator is taken over all labels.

Let $S_k (\subset X)$ be a set of user-given pixels, called seeds, with label $l_k$. Then the likelihood $p(x_i|l_k)$ can be obtained by

$$p(x_i|l_k) = \sum_{s \in S_k} p(x_i|x_s, l_k)p(x_s|l_k) = \frac{1}{|S_k|} \sum_{s \in S_k} u_{is}, \quad (2.2)$$

where $u_{is} = p(x_i|x_s, l_k)$. The likelihood $p(x_i|l_k)$ of each pixel $x_i$ is modeled by a mixture of probabilities $\{u_{is}\}_{x_s \in X_k}$ with respect to the seeds $S_k$, and each seed $x_s \in S_k$ has a probability $p(x_s|l_k)$ for the label $l_k$. The probability $u_{is}$ indicates the relevance score between a pixel $x_i$ and a seed $x_s \in S_k$. Recently, defining the relevance score between two nodes is one of the fundamental issues in graph mining. One very successful technique is based on RWR [54]. Thus, in this work, we propose to use the steady-state probability defined by RWR. Compared with traditional graph distances (such as shortest path, maximum flow), this steady-state probability can capture the whole relationship between two nodes by identifying all possible paths between them. The seed distribution $p(x_s|l_k)$ is defined by a uniform distribution, $1/|S_k|$. Since the likelihood $p(x_i|l_k)$ is computed by the average of the relevance distributions of all the seeds $S_k$ with the label $l_k$, our algorithm is less dependence on the number of seeds. Now, we briefly describe the process of our image segmentation algorithm. First, we construct a weighted graph in an image.
2.2. PROPOSED GENERATIVE MODEL

Figure 2.3: Overview of our proposed segmentation algorithm. Given the seeds in (b), the posterior probabilities (d), (e) and (f) are obtained by computing Eq. (2.1) for “red”, “green”, and “blue” labels, respectively. The segmentation result (c) is obtained by assigning each pixel the label having maximum posterior probability.

Then, we define $u_{is}$ as the steady-state probability that a particle starting from a seed $x_s \in S_k$ stays at a pixel $x_i$ in this graph. After computing this steady-state probability using RWR, we can estimate the likelihood $p(x_i|l_k)$ in (2.2) and, finally assign the label $r_i$ with maximum posterior probability in (2.1) to each pixel as follows:

$$r_i = \arg \max_{l_k} p(l_k|x_i) = \arg \max_{l_k} p(x_i|l_k),$$ \hspace{1cm} (2.3)

since the prior probability $p(l_k)$ in (2.1) is assumed uniform.

Fig. 2.3 shows the overall process of our algorithm from the seeds to the calculation of each label posterior probability $p(l_k|x_i)$ and the resulting segmentation. It starts with three initial seed labels: “red”, “green”, and “blue” as shown in Fig.
CHAPTER 2. RWR-BASED INTERACTIVE SEGMENTATION

Figure 2.4: Illustration of RWR. Given an input image, we construct a graph $G_X$ whose nodes are a set of pixels $\mathcal{X}$ (green circle) in (a). In the graph $G_X$, RWR has two types of particle movements: (b) “Transition” (violet line) and (c) “Restart” (violet dotted line), at each pixel $x_i$.

2.3(b). After computing the likelihood for each label, we generate the posterior probabilities as shown in Fig. 2.3(d),(e) and (f). Using the decision rule in (2.3), each pixel is assigned the label that has the maximum probability. Finally, we obtain the segmentation result in Fig. 2.3(c), where the object boundary is drawn in red color overlaid on the original image.

2.3 RWR-Based Likelihood Estimation

To estimate the likelihood $p(x_i|l_k)$ in (2.2), the relevance scores $\vec{u}_s = [u_{i_s}]_{i=1,...,N_X}$ should be estimated for each seed $x_s \in \mathcal{S}_k$. Here we explain how to compute the relevance score $u_{i_s}$ using the basic RWR framework and its variants in detail.

2.3.1 Basic RWR

We first construct a graph $G_X = (\mathcal{V}_X, \mathcal{E}_X)$ whose nodes $\mathcal{V}_X$ are the $N_X$ pixels $\mathcal{X}$, such as in Fig. 2.4(b). The edges $\mathcal{E}_X$ are connected between pairs of pixels, generally in the 4-neighborhood system. An edge $e_{ij} \in \mathcal{E}_X$ between two pixels $x_i$ and $x_j$ has the
following weight $w_{ij}^X$:

$$w_{ij}^X = \exp (-\theta_g \| \vec{g}_i - \vec{g}_j \|), \quad (2.4)$$

where $\vec{g}_i$ is the color vector of pixel $x_i$ in Lab color space and $\theta_g$ is a constant that controls the strength of $w_{ij}^X$.

To compute the relevance score of any pixel $x_i$ for the seed $x_s \in S_k$, RWR operates in the graph $G_X$ as follows.

**Main Idea:** Let us consider a particle that starts from the seed $x_s$. This particle transmits along the available edges every time ("transition" move in Fig. 2.4(b)), except that, before its transfer, with probability $c$ ($0 < c < 1$), it goes back to the seed $x_s$ ("restart" move in Fig. 2.4(c)). Then, the steady-state probability $u_{is}$ that the particle starting from $x_s$ stays at the pixel $x_i$ is what we want, the relevance score of $x_i$ with respect to $x_s$ in (2.2).

**Formulation:** Let $u_{is}^{(t)}$ be the probability that the particle starting from the seed $x_s$ arrives at the pixel $x_i$ after the $t$ iterations. In RWR, the arrival probability vector $\vec{u}_s^{(t)} = [u_{is}^{(t)}]_{N_X \times 1}$ is computed as follows:

$$\vec{u}_s^{(t)} = (1 - c) \mathbf{P}_X \vec{u}_s^{(t-1)} + c \vec{b}_s, \quad (2.5)$$

where $\vec{b}_s = [b_{is}]_{N_X \times 1}$ is a vector indicating the seed $x_s$ and its $i$-th element $b_{is}$ is 1 if $i = s$ and 0 otherwise. The transition matrix $\mathbf{P}_X$ is the adjacency matrix $\mathbf{W}_X = [w_{ij}^X]_{N_X \times N_X}$ in (2.4) column-normalized:

$$\mathbf{P}_X = [p_{ij}^X]_{N_X \times N_X} = \mathbf{W}_X \mathbf{D}_X^{-1}, \quad (2.6)$$

where $\mathbf{D}_X = \text{diag}([d_1^X, ..., d_{N_X}^X])$, whose $j$-th diagonal element is $d_j^X = \sum_{i=1}^{N_X} w_{ij}^X$, is the degree matrix. After convergence $t \to \infty$, the steady-state probability vector
\vec{u}_s = [u_{is}]_{N_X \times 1}\text{ is:}

\begin{align*}
\vec{u}_s &= (1 - c)\mathbf{P}_X \vec{u}_s + c\vec{b}_s = c(\mathbf{I}_X - (1 - c)\mathbf{P}_X)^{-1}\vec{b}_s, \quad (2.7)
\end{align*}

where \( \mathbf{I}_X \) is the identity matrix of size \( N_X \times N_X \).

**Relaxation:** The vector \( \vec{u}_s \) in (2.7) can be interpreted as the solution to minimize the following cost function \( F_{\text{RWR}} \):

\begin{align*}
F_{\text{RWR}} &= \sum_{i,j=1}^{N_X} w_{ij} \left| \frac{u_{is}}{d^X_i} - \frac{u_{js}}{d^X_j} \right|^2 + \tau \sum_{i=1}^{N_X} \frac{1}{d^X_i} |u_{is} - b_{is}|^2, \quad (2.8)
\end{align*}

where \( \tau = \frac{c}{1-c} \). The first term is the smoothness constraint in which good scores should not change too much between neighboring pixels. Since this term is defined as local variation by the difference in normalized scores between two end-pixels of each edge, the score variation between close-by pixels can be discretized. The second term in \( F_{\text{RWR}} \) is the fitting constraint in which good scores should not change too much from the initial assignment. A positive parameter \( \tau \) (or \( c \)) specifies the relative amount of smoothness and fitting constraints.

**Interpretation:** In a RWR view, \( \vec{u}_s \) in (2.7) consists of the affinity scores of all pixels \( \mathcal{X} \) with respect to the seed \( x_s \in \mathcal{S}_k \). In other words, each score \( u_{is} \) implies the likelihood that \( x_i \) has the same label \( l_k \) which was assigned to \( x_s \in \mathcal{S}_k \). It can be reformulated as follows:

\begin{align*}
\vec{u}_s &= c(\mathbf{I}_X - (1 - c)\mathbf{P}_X)^{-1}\vec{b}_s \\
&= c \sum_{t=0}^{\infty} (1 - c)^t \mathbf{P}_X^t \vec{b}_s. \quad (2.9)
\end{align*}

\( \vec{u}_s \) is defined as the weighted sum of all vector \( \mathbf{P}_X^t \vec{b}_s, t = 0, ..., \infty \). Note that \( \mathbf{P}_X^t \) is the \( t \)-th order transition matrix, whose \((i, j)\) element can be interpreted as the total probability for a particle that begins at \( x_j \) to end up at \( x_i \) after \( t \) iterations, considering all possible paths between two pixels. By varying the number of iterations \( t \), we
2.3. RWR-BASED LIKELIHOOD ESTIMATION

Figure 2.5: Comparison of the relevance scores \( \vec{u}_+ \) with respect to the variation of the restarting probability \( c \) in RWR. (a) Synthetic images with one pixel-seed \(+\). (b), (c), and (d) Relevance scores \( \vec{u}_+ \), estimated by RWR with \( c = 10^{-4} \), \( c = 10^{-5} \), and \( c = 10^{-6} \), respectively (log scale).

explicitly explore relationship at different scales in the image, and as \( t \) increases, we expect to find coarser structure. Therefore, RWR gives the texture effects by considering all paths between two pixels at all scales (any iteration number \( t = 0, ..., \infty \)) in the image. Since close-by pixels are likely to have high similarity value, as \( t \) increases, \( P^X_{bs} \) has lower weight \( c(1-c)^t \). The resulting vector \( \vec{u}_s \) can be solved by using a linear method of matrix inversion. Although it requires more memory space, fast computation is possible if the matrix is sparse. Since the 4-neighborhood system is used in this work, the transition matrix \( P^X \) is highly sparse. Therefore, Eq. (2.7) can be calculated fast using the linear system solver implemented by the MATLAB division operator ‘\( \backslash \)’. However, if the number of nearest neighbors is chosen to be a fixed large number, the complexity of matrix inversion is very high. In this case, some approximation methods such as Fast RWR technique [61] can be used.

The range of particle propagation is varied according to the restarting probability \( c \), as shown in Fig. 2.5. If \( c \) is decreased, the probability that the particle travels over a larger area is increased. This means that by varying \( c \), we can control the
extent of the label information of seed $x_s \in S_k$ in the image. Since small $c$ is required for large-size image, it is decided in inverse proportion to the number of pixels $N_X$ as follows.

$$c = 1/N_X.$$  \hspace{1cm} (2.10)

Fig. 2.6 shows four example segmentations using RWR. Although the likelihoods for each label are well varied according to image configuration, the pixels near $x_s$ may have excessively high likelihoods. Therefore, the segmentation errors often occur.
such as in the top row of Fig. 2.6.

2.3.2 Using Boundary Cue: Data-Driven RWR

In the basic RWR framework, all pixels $X$ should have the same restarting probability $c$. If the locally adaptive restarting probability of each pixel is derived, intrinsic image structure can be well reflected in RWR. Therefore, we devise new modified dRWR that can incorporate the nonuniform restarting probabilities.

**Main Idea:** dRWR operates the same as RWR except that each pixel $x_i$ can have the different restarting probability $c_i$ that the particle at $x_i$ goes back to the seed $x_s$. In this work, the probability $c_i$ is assigned using the boundary information of pixel $x_i$ and the relationship between the scribble $S_k$ containing $x_s$ and other scribbles $S_{\sim k} = \{S - S_k\}$ as follows.

$$
c_i = \begin{cases} 
1 & \text{if } x_i \in S_{\sim k} \\
\mu & \text{if } \psi(x_i) = 1 \\
c & \text{otherwise}
\end{cases}, \quad (2.11)
$$

where $c < \mu \leq 1$ and $\psi(x_i)$ indicates an edge indicating function that provides the binary output with a value 1 if the pixel $x_i$ is on the edge and 0 otherwise. If $x_i$ is included in $S_{\sim k}$, the relevance score between $x_i$ and the seed $x_s \in S_k$ should be low. Thus, by setting the restarting probability $c_i = 1$, we can make the steady-state probability $u_{is}$ close to 0. Also, as the particle on the edge returns with the large restarting probability $\mu$, the consistency of relevance scores in the edge-bounded areas can be emphasized. In other cases, $c_i$ is same to $c$ in (2.10).

**Formulation:** After the $t$ iterations, the arrival probability vector $u_s^{(t)}$ in dRWR is
formulated as follows:

\[ \vec{u}_{s}^{(t)} = \mathbf{P}_X (\mathbf{I}_X - \mathbf{C}_X) \vec{u}_{s}^{(t-1)} + \overrightarrow{c}_{s}^{(t-1)} \vec{b}_s, \]  

\[ (2.12) \]

where \( \mathbf{C}_X = \text{diag}(\vec{c}) \) is the matrix whose diagonal elements are the restarting probabilities \( \vec{c} = [c_i]_{N_X \times 1} \) in (2.11). The total restarting probability \( c_s^{(t)} \) that the particle returns to the seed \( x_s \) at the iteration \( t \) is formulated as

\[ c_s^{(t)} = \sum_{i=1}^{N_X} c_i u_{is}^{(t)} = \vec{c}^T \vec{u}_{s}^{(t)}. \]  

\[ (2.13) \]

By inserting \( c_s^{(t)} \) in (2.13), Eq. (2.12) can be transformed into

\[ \vec{u}_{s}^{(t)} = \mathbf{P}_X (\mathbf{I}_X - \mathbf{C}_X) \vec{u}_{s}^{(t-1)} + \overrightarrow{b}_s (\vec{c}^T \vec{u}_{s}^{(t-1)}) \]  

\[ = \left( \mathbf{P}_X (\mathbf{I}_X - \mathbf{C}_X) + \overrightarrow{b}_s \vec{c}^T \right) \vec{u}_{s}^{(t-1)}. \]  

\[ (2.14) \]

After convergence \( t \rightarrow \infty \), the steady-state probability vector \( \vec{u}_{s} \) in dRWR is defined as

\[ \vec{u}_{s} = \left( \mathbf{P}_X (\mathbf{I}_X - \mathbf{C}_X) + \overrightarrow{b}_s \vec{c}^T \right) \vec{u}_{s}. \]  

\[ (2.15) \]

Thus, the vector \( \vec{u}_{s} \) corresponds to the eigenvector of \( \left( \mathbf{P}_X (\mathbf{I}_X - \mathbf{C}_X) + \overrightarrow{b}_s \vec{c}^T \right) \) with the eigenvalue 1. Since this matrix is very sparse, its eigen-decomposition is efficiently computed.

Fig. 2.7 shows that with a larger \( \mu \), the pixels in the edge-bounded areas have more similar relevance scores. To make the segmentation results discretized on the edges, we decide \( \mu = 1 \) in our experiments. Fig. 2.8 shows four example segmentations by dRWR. Compared with the results by RWR in Fig. 2.6(e), the segmentation results with a clear distinction on the edges in Fig. 2.8(e) can be obtained using the edge information in Fig. 2.8(a). Even if the weak boundaries exist, dRWR works well. However, if the complete edge-bounded area does not
2.3. RWR-BASED LIKELIHOOD ESTIMATION

Figure 2.7: Comparison of the relevance scores $\vec{u}^+$ with respect to the variation of the parameter $\mu$ in dRWR. (a) Synthetic image with one pixel-seed +. (b),(c), and (d) Relevance scores $\vec{u}^+$, estimated by dRWR with $\mu = c \approx 9.8 \times 10^{-6}$, $\mu = 10^{-2}$, and $\mu = 1$, respectively.

contain any seeded pixel, it is difficult to decide the label of that area by dRWR such as the area with black color in Fig. 2.8(e).

2.3.3 Using Higher-Order Cue: Higher-Order RWR

RWR can consider only pairwise connections, not higher-order. Also, if the number of pixels $N_X$ is very large, using long-range connections cause high computational complexity. To make up for these weak points, hRWR is proposed as follows.

Instead of the only pixel-based layer $G_X$, we construct a bilayer graph $G = (\mathcal{V}, \mathcal{E})$ whose nodes $\mathcal{V}$ are the $N_X$ given pixels $X$ and the $N_Y$ over-segmented regions $\mathcal{Y} = \{y_n\}_{n=1,...,N_Y}$, generated by an unsupervised segmentation algorithm such as Mean Shift [4], as shown in Fig. 2.9(b). The edges $\mathcal{E} = \mathcal{E}_X \cup \mathcal{E}_Y \cup \mathcal{E}_{XY}$ are linked by different criteria according to the node types. An undirected edge exists, if the one of the following conditions is satisfied.

1) In the pixel-based layer, an intralayer edge $e_{ij}^X \in \mathcal{E}_X$ between two neighboring pixels $x_i, x_j \in \mathcal{X}$ exists with the weight $w_{ij}^X$ in (2.4).
Figure 2.8: Example segmentations by dRWR in Fig. 2.6(a). (a) Edge maps. (b), (c), and (d) Relevance scores $\vec{u}_+^+$, $\vec{u}_+^+$, and $\vec{u}_+^+$ for “red”, “green”, and “blue” labels, respectively. (e) Segmentation results. In (e), the labels of pixels in the area with black color are undecidable.

2) In the region-based layer, another intralayer edge $e_{nm}^Y \in \mathcal{E}_Y$ between two regions $y_n, y_m \in \mathcal{Y}$ which share a common boundary is connected with the following weight $w_{nm}^Y$:

$$w_{nm}^Y = \exp \left(-\theta_g \|\vec{g}_n - \vec{g}_m\|\right),$$

(2.16)

where $\vec{g}_n$ is the mean color vector of the inner pixels of region $y_n$. Using the links across regions can consider long-range connections.

3) An interlayer edge $e_{in}^{XY} \in \mathcal{E}_{XY}$ between a pixel $x_i \in \mathcal{X}$ and its corresponding
2.3. RWR-BASED LIKELIHOOD ESTIMATION

Figure 2.9: Illustration of hRWR. In the bilayer graph $G$, the nodes are the pixels $X$ (green circle) and the regions $Y$ (violet circle). The regions $Y$, whose boundaries are drawn in red color overlaid on the original image, are generated by Mean Shift [4] in (a). (b) shows the edges connected to one region with blue lines. In $G$, hRWR has (c) “Jump” move (brown dotted lines) as well as “transition” and “restart” moves at each layer.

Region $y_n \in Y$ is added with the following weight $w_{in}^{XY}$ ($= w_{ni}^{YX}$):

$$w_{in}^{XY} = 1.$$ \hspace{1cm} (2.17)

Since the correlation between the pixels in the same region is more emphasized through the interlayer connections between the region and its inner pixels in (2.17), these connections provide implicit higher-order effects.

**Main Idea:** hRWR operates the same as RWR except that at each pixel $x_i$ or region $y_n$, the particle jumps along the interlayer edges $E_{XY}$ in $G$ with probability $\alpha_i$ or $\bar{\alpha}_n$ (“jump” move in Fig. 2.9(c)). Namely, if the particle stays at any pixel $x_i$, it moves to one region corresponding to $x_i$ in the opposite layer with probability $\alpha_i$ (“jump”), to the seed $x_s$ with probability $c_i$ (“restart”) or to the neighboring pixels in the same layer with probability $1 - c_i - \alpha_i$ (“transition”). Similarly, the
particle at any region $y_n$ moves to the inner pixels of $y_n$ in the opposite layer with probability $\alpha_n$ ("jump"), to the seed $x_s$ with probability $\bar{c}_n$ ("restart") or to the adjacent regions in the same layer with probability $1 - \bar{c}_n - \alpha_n$ ("transition"). Two restarting probabilities $c_i$ and $\bar{c}_n$ in the pixel- and region-based layer are decided, respectively, as follows:

$$(c_i, \bar{c}_n) = (c, \bar{c}),$$

where $c$ in (2.10) and $\bar{c} = 1/N_Y$. Also, two jumping probabilities $\alpha_i$ and $\bar{\alpha}_n$ for the pixel $x_i$ and the region $y_n$ are set, respectively, as follows:

$$(\alpha_i, \bar{\alpha}_n) = \begin{cases} 
(0, 0) & \text{if } x_i \in y_n \in \{\bar{S}_k \cap \bar{S}_\sim k\} \\
(\gamma, \gamma) & \text{otherwise}
\end{cases},$$

where $\bar{S}_k$ and $\bar{S}_\sim k = \{\bar{S} - \bar{S}_k\}$ are the sets of seeded regions containing the seeded pixels in $S_k$ and $S_\sim k$, respectively. In $\mathcal{Y}$, some regions may be incorrect. If the region $y_n$ corresponding to the pixel $x_i$ contains the seeded pixels of several label types, that region is incomplete. Thus, by setting $\alpha_i = \bar{\alpha}_n = 0$, we block the interlayer link between $x_i$ and its corresponding region $y_n$. In other cases, with a larger $\gamma$, the relationship between the pixels and their corresponding regions gets much closer.

**Formulation:** Let $\vec{u}^{(t)}_s = [\vec{u}^{(t)}_{ns}]_{N_Y \times 1}$ be the probability vector that the particle starting from the pixel $x_s$ arrives at the regions $\mathcal{Y}$ after the $t$ iterations. In hRWR, the total arrival probability vector $\vec{u}^{(t)}_s = [\vec{u}^{(t)}_{s}; \vec{u}^{(t)}_{\bar{s}}]$ is formulated as follows:

$$\vec{u}^{(t)}_s = \mathbf{P}(\mathbf{I} - \mathbf{C}) \vec{u}^{(t-1)}_s + \vec{c}^{(t-1)} \vec{b}_s,$$

where the identity matrix $\mathbf{I}$ is the same size as $\mathbf{P}$ and $\vec{b}_s = [\vec{b}_s; \bar{0}_{N_Y \times 1}]$ is the extended vector indicating the seeded pixel $x_s$. The matrix $\mathbf{C} = \text{diag}(\vec{c})$ has the diagonal elements $\vec{c} = [c_1, ..., c_{N_X}, \bar{c}_1, ..., \bar{c}_{N_Y}]^T$ which are the restarting probabilities for all
pixels $\mathcal{X}$ and regions $\mathcal{Y}$. The matrix $\mathbf{P}$ is:

$$
\mathbf{P} = \begin{bmatrix}
\mathbf{P}_X(I_X - \mathbf{A}_X) & \mathbf{P}_{XY} \mathbf{A}_Y \\
\mathbf{P}_{YX} \mathbf{A}_X & \mathbf{P}_Y(I_Y - \mathbf{A}_Y)
\end{bmatrix},
$$

(2.21)

where the matrices $\mathbf{A}_X = \text{diag}([\lambda_1, ..., \lambda_{N_X}])$ and $\mathbf{A}_Y = \text{diag}([\bar{\lambda}_1, ..., \bar{\lambda}_{N_Y}])$ have the diagonal elements $\lambda_i = \frac{a_i}{1 - c_i}$ and $\bar{\lambda}_n = \frac{\bar{a}_n}{1 - \bar{c}_n}$, respectively. It consists of two intralayer transition matrices $\mathbf{P}_X$ in (2.6) & $\mathbf{P}_Y = [p^Y_{nm}]_{N_Y \times N_Y}$, and two interlayer jump matrices $\mathbf{P}_{XY} = [p^{XY}_i]_{N_X \times N_Y}$ & $\mathbf{P}_{YX} = [p^{YX}_i]_{N_Y \times N_X}$. The elements of $\mathbf{P}_Y$, $\mathbf{P}_{XY}$, and $\mathbf{P}_{YX}$ are defined by

$$
p^Y_{nm} = \frac{w^Y_{nm}}{\sum_{n=1}^{N_Y} w^Y_{nm}}
$$

$$
p^{XY}_i = \frac{u^{XY}_i}{\sum_{i=1}^{N_X} u^{XY}_i}
$$

(2.22)

where $w^Y_{nm}$ in (2.16) and $w^{XY}_i = w^{YX}_i$ in (2.17). The total restarting probability $\hat{c}_s^{(t)}$ that the particle returns to the seeded pixel $x_s$ at the iteration $t$ is formulated as

$$
\hat{c}_s^{(t)} = \sum_{i=1}^{N_X} c_i u^{(t)}_{is} + \sum_{n=1}^{N_Y} \bar{c}_n \bar{u}^{(t)}_{ns} = \bar{\mathbf{u}}^{(t)} \bar{\mathbf{c}}^T \mathbf{c}^T \bar{\mathbf{c}}.
$$

(2.23)

By inserting $\hat{c}_s^{(t)}$ in (2.23), the steady-state probability vector $\bar{\mathbf{u}}_s$ is:

$$
\bar{\mathbf{u}}_s = \left( \mathbf{P}(I - \mathbf{C}) + \bar{\mathbf{c}} \bar{\mathbf{c}}^T \right) \bar{\mathbf{u}}_s.
$$

(2.24)

Thus, the steady-state probabilities $\bar{\mathbf{u}}_s$ for the seeded pixel $x_s$ correspond to the eigenvector of $\left( \mathbf{P}(I - \mathbf{C}) + \bar{b}_s \bar{\mathbf{c}} \right)$ with the eigenvalue 1.

Fig. 2.10 shows that with a larger $\gamma$, the pixels inside each region have more similar relevance scores. However, the differences in relevance scores between the adjacent regions may be smaller. Thus, we empirically decide $\gamma = 10^{-4}$. Fig. 2.11 shows four example segmentations by hRWR. Compared with the results by RWR
CHAPTER 2. RWR-BASED INTERACTIVE SEGMENTATION

Figure 2.10: Comparison of the relevance scores $\vec{u}_+$ with respect to the variation of the parameter $\gamma$ in hRWR. (a) Synthetic image with one pixel-seed $+. (b),(c),$ and (d) Relevance scores $\vec{u}_+$, estimated by hRWR with $\gamma = 10^{-1}$, $\gamma = 10^{-2}$, and $\gamma = 10^{-4}$, respectively.

Figure 2.11: Example segmentations by hRWR in Fig. 2.6(a). (a) Over-segmentations. (b), (c), and (d) Relevance scores $\vec{u}_+$, $\vec{u}_+$, and $\vec{u}_+$ for “red”, “green”, and “blue” labels, respectively. (e) Segmentation results.
2.3. RWR-BASED LIKELIHOOD ESTIMATION

in Fig. 2.6(e), hRWR makes the relevance scores inside each region nearly uniform in Fig. 2.11(e) by using the higher-order relationship between that region and its inner pixels in Fig. 2.11(a). However, the relevance scores in the different regions may be not clearly discretized such as in Fig. 2.11(e).

2.3.4 Combining Boundary and Higher-Order Cues: Global RWR

By combining the ideas of dRWR and hRWR, we finally design global RWR (gRWR).

**Main Idea:** gRWR operates the same as hRWR except that each pixel \( x_i \) (or region \( y_n \)) has the data-driven restarting probability \( c_i \) (or \( \bar{c}_n \)) such as in dRWR. Therefore, two probabilities \( c_i \) and \( \alpha_i \) for the pixel \( x_i \) are decided as follows:

\[
(c_i, \alpha_i) = \begin{cases} 
(1, 0) & \text{if } x_i \in S_k \\
(c, 0) & \text{if } x_i \in y_n \in \{\mathcal{S}_k \cap \mathcal{S}_{k'}\} \\
(\mu, 0) & \text{if } \psi(x_i) = 1 \\
(c, \gamma) & \text{otherwise}
\end{cases}, \tag{2.25}
\]

\[
(\bar{c}_n, \bar{\alpha}_n) = \begin{cases} 
(1, 0) & \text{if } y_n \in \bar{\mathcal{S}}_k \\
(\bar{c}, 0) & \text{if } \bar{x}_n \in \{\bar{\mathcal{S}}_k \cap \bar{\mathcal{S}}_{k'}\} \\
(\bar{c}, \gamma) & \text{otherwise}
\end{cases}, \tag{2.26}
\]

where \( c \) and \( \bar{c} \) in (2.18). The data-driven restarting probability \( c_i \) is assigned, same in (2.11). Since the region information of the edge pixel \( x_i \) is meaningless, we additionally block the interlayer link between \( x_i \) and its corresponding region by setting \( \alpha_i = 0 \).

**Formulation:** The formulation of gRWR is exactly same to that of hRWR in (2.24) with the data-driven restarting and jumping probabilities in (2.25) and (2.26).
Fig. 2.12 shows four example segmentations by gRWR. The segmentation results in Fig. 2.12(d) have the advantages of dRWR in Fig. 2.8(e) and hRWR in Fig. 2.11(e). In the regions which do not contain any seeded pixel, gRWR can produce the relevance scores well by considering the long-range relationship between the regions. Fig. 2.13 demonstrates the superiority of gRWR in terms of accuracy, compared with dRWR and hRWR.
2.4. Experimental Results

We begin by analyzing the performance under two difficult problems: weak boundary problem and texture problem. We then compare the segmentation results obtained by the state of the art methods including GC [1], GD [2], and RW [3] on natural images and provide quantitative comparisons.

2.4.1 Weak Boundary Problem

The weak boundary problem is to find the broken edges when they are parts of a consistent boundary. In [3], RW shows better segmentation result than GC in low contrast with small number of seeds. Although GC and RW are capable of finding weak boundaries, our algorithm gives more intuitive outputs. In Fig. 2.14, our algorithm is compared with GC and RW in the weak boundary problem. We
Figure 2.14: Comparison of our algorithm with GC and RW for finding weak boundaries. Given two labels (“green” and “blue”) and original images in (a) (Top: image created with a black circle with four erased sections. Bottom: image created with a 3 × 3 black grid with four erased sections. ), (b),(c) and (d) are the segmentation results of GC, RW and our algorithm respectively.

used two synthetic examples: circle and 3 × 3 grid with four sections erased. Given the seeds (green and blue) in Fig. 2.14(a), the segmentations were obtained in Fig. 2.14(b)-(d). Fig. 2.14(b) shows clearly that GC has small cut problem. In Fig. 2.14(c), we can confirm that the segmentations of RW is substantially affected by the difference between the numbers of Green and Blue seeds. Namely, RW is sensitive to the number of seeds. In contrast, Fig. 2.14(d) shows that our algorithm is less dependent on the number of seeds and produces better segmentations, because the likelihood is computed by the average of the relevance scores of all the seeds.
Figure 2.15: Comparison of our algorithm with GC, GD, and RW on synthetic textured images. (b), (c), (d) and (e) are the segmentation results of GC, GD, RW, and our algorithm, respectively.

2.4.2 Texture Problem

In GC, GD, and RW, it is hard to separate the textured region without considering higher-order connections, because they deal with the minimum cut criterion, the shortest path distance, and the first arrival probability, respectively. Since these three algorithms do not consider the information of seeds that are inside the pre-labeled regions, it is not easy for them to take into account the effects of texture. On the other hand, our algorithm can reflect the texture information by using the steady-state probability of RWR, because RWR considers all possible paths between two nodes in a small neighborhood system. In spite of the use of a small neighborhood system, it captures the textural structure well and obtains object details. In Fig. 2.15, we used the synthetic images that consist of four or five different kinds of textures. This is the texture segmentation problem that one texture is extracted among them. The segmentation results in Fig. 2.15(e) show that our
algorithm produced more reliable texture segmentations on these synthetic textured images than GC, GD, and RW.

### 2.4.3 Quantitative comparisons

The previous two situations often arise in natural images. Now, we compare the segmentations obtained from GC, GD, RW, and our algorithm on natural images. We utilized the Berkeley Segmentation Dataset (BSDS300) [32] of natural images where human subjects provide foreground/background labels as the ground-truth segmentations. For quantitative comparisons, the similarity between the segmentation result and the preset ground-truth segmentation was measured using a normalized overlap $a_o$ [38]:

$$a_o = \frac{|R \cap R'|}{|R \cup R'|}$$  \hspace{1cm} (2.27)

where $R$ is the set of pixels assigned as the foreground from the segmentation result and $R'$ is that from the ground-truth segmentation. In this work, it was used as the accuracy measure of the image segmentation. For this experiments, we chose the natural images with highly textured (cluttered) regions or with similar color distributions between the foregrounds and backgrounds. In Fig. 2.16, the segmentations were produced from the three different algorithms on these natural images. Compared with the segmentations from GC, GD, and RW in Fig. 2.16, our algorithm has better segmentations qualitatively and quantitatively. The quantitative comparison confirms the relevance and accuracy of our algorithm.

We then demonstrate the quality of our proposed algorithm on the Microsoft GrabCut database \(^1\) which consists of 50 images with tri-maps and ground-truth segmentations.

---

\(^1\)http://research.microsoft.com/en-us/um/cambridge/projects/visionimagevideoediting/segmentation/grabcut.htm
2.4. EXPERIMENTAL RESULTS

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

\[ a_o = 0.629555 \quad a_o = 0.573076 \quad a_o = 0.747215 \]

\[ a_o = 0.741653 \quad a_o = 0.581065 \quad a_o = 0.874370 \]

\[ a_o = 0.652658 \quad a_o = 0.639281 \quad a_o = 0.750052 \]

\[ a_o = 0.796337 \quad a_o = 0.843284 \quad a_o = 0.872281 \]

\[ a_o = 0.566904 \quad a_o = 0.611782 \quad a_o = 0.892063 \]

Figure 2.16: Comparison of our algorithm with GC and RW on natural images. (b),(c) and (d) are the segmentation results of GC, RW, and our algorithm, respectively. \( a_o \) is the accurate rate in (2.27). (e) Ground-truth segmentations.
### CHAPTER 2. RWR-BASED INTERACTIVE SEGMENTATION

#### Table 2.1: Segmentation accuracy $a_o$ in the Microsoft GrabCut database.

<table>
<thead>
<tr>
<th>Segmentation model</th>
<th>Accuracy ($a_o$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GC [5]</td>
<td>0.9522</td>
</tr>
<tr>
<td>GD [2]</td>
<td>0.9517</td>
</tr>
<tr>
<td>RW [3]</td>
<td>0.9510</td>
</tr>
<tr>
<td>RWR</td>
<td>0.9444</td>
</tr>
<tr>
<td>dRWR</td>
<td>0.9538</td>
</tr>
<tr>
<td>hRWR</td>
<td>0.9535</td>
</tr>
<tr>
<td>gRWR</td>
<td><strong>0.9581</strong></td>
</tr>
</tbody>
</table>

Figure 2.17: Example segmentations using the tri-maps on the Microsoft GrabCut database. (a) Test image with tri-map. Segmentation results by (b) GC [5], (c) RW [3], and (e) Our algorithm gRWR.
2.5. **CONCLUSION**

We present a novel generative image segmentation model in the Bayesian framework. More importantly, we provide a new interpretation of RWR for image segmentation. Although RW [3] is also based on the Random Walks concept, our work is conceptually different from RW, and produces significant improvement in performance as shown in the experiments. The key differences between RW and our work are "First arrival probability vs. Average probability". In [3], the score between a pixel and each label is defined by the first arrival probability that a particle starting at a pixel reaches to a seed. On the other hand, in our work it is defined as the average probability that a particle starting at one of the seeds stays at a pixel.

Our approach has several advantages for image segmentation. First, owing to the generative segmentation model based on RWR, it can obtain segmentations with just single label. Second, it is less dependent on the number of seeds, because the likelihood is computed by the average of the relevance scores of all the seeds.

Segmentations. Although most pixels have the initial labels as the seeds except for a narrow band around the objects, this database can evaluate the problem of extracting accurate object details. It is widely used for quantitative segmentation comparison [62] [39] with state-of-the-art algorithms. Table 2.1 presents the comparative evaluation of the interactive segmentation algorithms: GC [5], GD [2], RW [3], and our algorithms (RWR, dRWR, hRWR, gRWR). Our main algorithm gRWR quantitatively has better performance than the conventional methods. Fig. 2.17 illustrates the example segmentations on the Microsoft GrabCut database. This figure shows that our algorithm intuitively produces high-quality segmentation results.
Third, we can integrate region and boundary information globally in the RWR-based framework. Finally, it gives *qualitatively* and *quantitatively* better segmentations on natural images. Generally, large neighborhood system is needed for obtaining object details, because it captures image structure well. However, since it gives high computations, many efficient techniques, like multi-scale approach, have been proposed. Our RWR-based algorithm is an alternative solution, since it considers all possible paths between two nodes in small neighborhood system.

In our algorithm, the jumping and restarting probabilities were chosen empirically. However, they are not optimal for every image. If we can control them well, better segmentation results will be obtained. Thus, our future work will include the automatic selection of the optimal values of these parameters.
Chapter 3

Robust Higher-Order Interactive Segmentation

3.1 Introduction

Recently, interactive image segmentation algorithms, inspired by the user-inputs such as the scribbles which provide a partial labeling of the image, have gained popularity since these algorithms give the user the ability to affect the segmentation as necessary for a particular application. However, these algorithms should satisfy the following condition as well as the problem of extracting accurate object details. As a way to reduce the user effort, the segmentation should not be sensitive to where the seeds are positioned and how many seeds are used. To guarantee this condition without great memory and computation time costs, most popular algorithms find a set of the over-segmented regions, generated by an unsupervised segmentation method, as the resulting segmentation [63–65]. These algorithms are inspired by the hard constraint whereby pixels constituting a particular region should have the
Figure 3.1: Using the higher-order cues for interactive segmentation. The first row shows an image with one pixel-seed selected for each label. (b)-(e) Segmentation results by RW [3] with pairwise potentials, gRWR [6] with higher-order constraint, Robust $P^m$ model [7] with parametric higher-order potentials, and our method with nonparametric higher-order cues, respectively.

same label. They have the benefit of using more informative features extracted from the pixels within the regions. If, however, these regions are not consistent with boundaries in the image, there are radical difficulties in obtaining the exact solutions. In fact, such situations often arise in natural images.

To overcome this hard constraint, many algorithms to combine multiple segmentations of the same image have been proposed [7, 66, 67]. Unlike [66, 67] that heuristically merge multiple regions, the work of Kohli et al. [7] couples higher-order potential function defined in the regions with conventional unary and pairwise constraints by using higher-order CRFs in a principled manner. However, since this higher-order function is defined as the parametric model with respect to the number of pixels in the region not taking the dominant label, its parameters should
be learned in the training stage. In the RWR-based segmentation algorithm [6], the higher-order constraint can be conceptually added in the global RWR (gRWR) framework. However, it is difficult to prove the higher-order effect theoretically. And the higher-order effect is actually limited for segmentation of natural images. In this work, we propose a generative model which has the ability of theoretically utilizing nonparametric higher-order cues defined in the over-segmented regions for segmentation [68] [31]. The key contributions of our algorithm are as follows.

1) We introduce a generative model for interactive segmentation, similarly in [6]. In this framework, we design a quadratic cost function to estimate the pixel likelihoods for each label.

2) We propose a new higher-order cost function of pixel likelihoods to partly enforce the label consistency inside the regions generated by an unsupervised image segmentation algorithm such as the Mean Shift algorithm [4]. Unlike the previous parametric higher-order potentials such as the Robust $P^n$ model [7], we efficiently consider the pairwise relationship between the pixels and their corresponding regions in a multi-layer graph. In this work, the representative region likelihoods are defined as higher-order cues for estimating the pixel likelihoods. Also, using multiple region candidates provides implicit boundary cue for segmentation.

3) We address a nonparametric learning technique to recursively estimate the higher-order cues from the resulting likelihoods of pixels included in each region. In this manner, we consider long-range connections between the regions that facilitate propagation of local grouping cues across larger image areas, unlike previous works Robust $P^n$ model [7] and RWR-based model [6]. These connections give the same
CHAPTER 3. ROBUST HIGHER-ORDER INTERACTIVE SEGMENTATION

effects as nonparametric priori model.

4) Our algorithm is less sensitive to user inputs and gives high-quality segmentation results, as shown in Fig. 3.1.

This chapter is organized as follows. In Section 3.2, our proposed generative model is introduced. We then explain in detail how to estimate the pixel likelihoods from the higher-order cues in Section 3.3. The experimental results are given in Section 3.4. Finally, we discuss our approach in Section 3.5.

3.2 Generative Model for Segmentation

Given an image $I$, each pixel $x_i \in \mathcal{X} = \{x_1, ..., x_{N_X}\}$ is to be assigned by one label $l_k \in \mathcal{L} = \{l_1, ..., l_K\}$ in a segmentation problem. Let $r_i \in \mathcal{R} = \{r_1, ..., r_{N_X}\}$ denote the label of pixel $x_i$ among $\mathcal{L}$. In a generative model, the required posterior probability distribution $p(\mathcal{R} | \mathcal{X})$ is obtained by use of the Bayesian rule:

$$p(\mathcal{R} | \mathcal{X}) = \frac{p(\mathcal{X} | \mathcal{R}) p(\mathcal{R})}{p(\mathcal{X})} \propto p(\mathcal{X} | \mathcal{R}) p(\mathcal{R}),$$

(3.1)

where $p(\mathcal{X} | \mathcal{R})$ and $p(\mathcal{R})$ are the likelihood and prior distributions, respectively. This can be done by computing the likelihood distribution $p(\mathcal{X} | \mathcal{R})$ and the prior distribution $p(\mathcal{R})$ separately. For segmentation, the optimal labels $\mathcal{R}^*$ of $\mathcal{X}$ are obtained by maximum a posterior (MAP) estimation as follows:

$$\mathcal{R}^* = \arg\max_{\mathcal{R}} p(\mathcal{R} | \mathcal{X}).$$

(3.2)

It is usually expressed by energy minimization as follows.

$$\mathcal{R}^* = \arg\min_{\mathcal{R}} -\log p(\mathcal{R} | \mathcal{X}) = \arg\min_{\mathcal{R}} -\log p(\mathcal{X} | \mathcal{R}) - \log p(\mathcal{R})$$

$$\approx \arg\min_{\mathcal{R}} E_{\text{unary}}(\mathcal{X}; \mathcal{R}) + E_{\text{pairwise}}(\mathcal{R}).$$

(3.3)
3.2. GENERATIVE MODEL FOR SEGMENTATION

Figure 3.2: Example segmentations using the GMM-based likelihood maps. (a) Input images with the seeds selected for each label. (b) and (c) Likelihoods estimated by GMM for “red” and “green” labels, respectively. (d) Segmentation results by assigning each pixel the label with the largest likelihood. (e) Refined segmentation results by adding the label smoothness constraint in the prior distribution (optimized by graph cuts).

This energy formulation consists of two unary $E_{\text{unary}}$ and pairwise $E_{\text{pairwise}}$ potentials, which correspond to the likelihood and prior distributions, respectively. Eq. (3.3) can be optimized by the general optimizers: graph cuts [69], belief propagation [70], and etc.

In most previous segmentation algorithms, the conditional independence assumption holds if the likelihood distribution $p(\mathcal{X}|\mathcal{R}) \approx \prod_{i=1}^{N_{\mathcal{X}}} p(x_i|r_i)$ factorizes as the product of likelihoods over instances. Here, the likelihood $q_{ik} = p(x_i|r_i = l_k)$ that the
CHAPTER 3. ROBUST HIGHER-ORDER INTERACTIVE SEGMENTATION

Label \( r_i \) of pixel \( x_i \) is \( l_k \in \mathcal{L} \) is computed using the parametric model, learned from only the user-given, labeled pixels, called seeds, \( S \subset X \), such as Gaussian Mixture Model (GMM) [71]. However, since the likelihood \( q_{ik} \) is decided according to the properties of pixel \( x_i \) in this model, the likelihood smoothness between the neighboring pixels can not be guaranteed. Therefore, the label smoothness constraint is imposed in the prior distribution \( p(\mathcal{R}) \) using the additional parametric model such as potts model and higher-order model [7]. Fig. 3.2 shows some example segmentations using the likelihoods estimated by GMM, commonly used likelihood estimator. When the background has a similar color distribution to the foreground, GMM produces very noisy likelihoods for the background and the foreground, such as in Fig. 3.2(b) and (c). If the label smoothness constraint is added in the prior distribution, the segmentation results such as Fig. 3.2(e) can be improved. However, they still may have a mass of errors, since the likelihoods are fixed irrespective of the resulting labels \( \mathcal{R}^* \).

To make up for this weakness, we focus on estimating the likelihoods \( \vec{q}_k = [q_{ik}]_{N_X \times 1} \) by considering the smoothness constraint that the neighboring pixels have the similar likelihoods for each label if their colors are similar. This constraint is imposed into the quadratic cost function for estimating the likelihoods \( \vec{q}_k \). In the previous RWR-based algorithm [6], the likelihoods can be interpreted as the solution to minimize the quadratic cost function such as Eq. (2.8) when the restarting and jump probabilities are fixed for all nodes. However, since the smoothness constraint is based on the difference in normalized scores between two end-pixels of each edge and the fitting constraint is slightly emphasized in this function, it is difficult to control the variation in the likelihood distribution such as in Fig. 3.3. Also, the limited range of pairwise connections are considered. Therefore, in this work, we
### 3.2. GENERATIVE MODEL FOR SEGMENTATION

![Figure 3.3: Example RWR-based segmentations in Fig. 3.2(a).](a) Edges (black color) + Region boundaries (red color). (b) and (c) Likelihoods estimated by gRWR which use the edgeness and the over-segmented regions in (a), where red represents a greater likelihood, for “red” and “green” labels, respectively (log scale). (d) Segmentation results by assigning each pixel the label with the largest likelihood.

The quadratic cost function for conceptually estimating each likelihood $q_{ik}$ in range of $[0, 1]$. Additionally, the higher-order constraint that the pixels in a particular region should have the similar likelihoods can be theoretically imposed into the cost function and the propagation of local grouping cues across the whole image is possible. In our experiments, in order to prove the performance of our likelihoods, we assume that a prior distribution $p(R)$ in (3.1) is uniform.
3.3 Proposed Likelihood Estimation

3.3.1 Basic Function

We first define the quadratic cost function $J_X^k$ which consists of pairwise $E_{k,P}^X$ and unary $E_{k,U}^X$ terms as follows, similarly in [39].

**Main Idea:** Let $S_k (\subset X)$ denote a set of seeds with the label $l_k$. We start with the pixel-based graph $G_X$, same as in Sec. 2.3.1. By definition of the relationship between all pixels $X$ in the graph $G_X$, the quadratic cost function $J_X^k$ of pixel likelihoods $\vec{q}_k = [q_{ik}]_{N_X \times 1}$ with respect to the label $l_k$ is as follows.

$$J_X^k = E_{k,P}^X + E_{k,U}^X = \sum_{i,j=1}^{N_X} w_{ij}^X (q_{ik} - q_{jk})^2 + \sum_{i=1}^{N_X} \lambda_i^X d_i^X (q_{ik} - q_{ik}^*)^2, \quad (3.4)$$

where $d_i^X = \sum_{j=1}^{N_X} w_{ij}^X$ in (2.4). The parameter $\lambda_i^X$ for the all seeded pixels $S = \{S_k\}_{k=1,...,L}$ is $\lambda$ if $x_i \in S$ and 0 otherwise. The pixel-seed likelihood $q_{ik}^*$ is 1 if $x_i \in S_k$ is the seed with the label $l_k$ and 0 otherwise. In (3.4), the first term $E_{k,P}^X$ is the label-continuity constraint that two neighboring pixels in the small neighborhood system, which is usually chosen to be either a 4 or 8 neighborhoods, should have the same label if their colors are similar. The second term $E_{k,U}^X$ is the unary constraint that each pixel seed in $S_k$ tends to have the user-given label $l_k$. The $\lambda$ is positive coefficient measuring how much we want to fit the initial seeds. Typically, $\lambda = \infty$ imposes the hard constraint the each seed definitely has the initial label. This term is multiplied by the value $d_i^X$, since the balance between two competing constraints $E_{k,P}^X$ and $E_{k,U}^X$ is needed.

**Formulation:** The function $J_X^k$ in (3.4) can be formulated as the matrix form with
respect to the likelihoods \( q_k \) as follows.

\[
J^X_k = \bar{q}_k^T (D_X - W_X) \bar{q}_k + (\bar{q}_k - \bar{q}_k^*)^T D_X \Lambda_X (\bar{q}_k - \bar{q}_k^*),
\]

(3.5)

where \( W_X = [w_{ij}^X]_{N_X \times N_X} \) and \( D_X = \text{diag}([d_1^X, ..., d_{N_X}^X]) \). The diagonal elements of the matrix \( A_X = \text{diag}([\lambda_1^X, ..., \lambda_{N_X}^X]) \) are the parameter \( \lambda \) for the seeds \( S \) and 0 otherwise. Differentiating this matrix formulation \( J^X_k \) with respect to \( \bar{q}_k \) and set to zero, we can get all likelihoods \( \bar{q}_k \) from the initial ones \( \bar{q}_k^* = [q_{ik}^*]_{N_X \times 1} \) for the label \( l_k \) simply by

\[
\frac{\partial J^X_k}{\partial \bar{q}_k} = \bar{q}_k - Z_X \bar{q}_k + A_X (\bar{q}_k - \bar{q}_k^*) = 0,
\]

where \( Z_X = D_X^{-1} W_X \). It can be transformed into

\[
\bar{q}_k = (I_X + A_X - Z_X)^{-1} (A_X \bar{q}_k^*).
\]

(3.6)

Since \((I_X + A_X - Z_X)\) is positive definite, the linear equation in (3.6) can be solved easily and we finally have the likelihoods \( \bar{q}_k \) of all pixels \( X \) by a sparse matrix inversion technique.

Fig. 3.4 shows four example segmentations by solving Eq. (3.6). Although this model produces the smooth likelihood maps in Fig. 3.4(b) and (c), it makes some errors according to the seed quantity and placement in Fig. 3.4(d).

3.3.2 Incorporating Higher-Order Cues

Let us consider the segmentation as the grouping of regions \( Y = \{y_n\}_{n=1,...,N_Y} \) generated by over-segmentations such as the Mean Shift algorithm [4], instead of pixels \( X = \{x_i\}_{i=1,...,N_X} \) in addition. In this case, we also should find the likelihood \( \bar{q}_{nk} = p(y_n | l_k) \) that the label of region \( y_n \) is \( l_k \). Here, we propose to simultaneously
Figure 3.4: Example segmentations by solving Eq. (3.6) in natural images. (a) Input images with the seeds selected for each label. (b) and (c) Likelihoods $\vec{q}_k$ for “red” and “green” labels, respectively. (e) Segmentation results by assigning each pixel the label with the largest likelihood.

estimate all pixel and region likelihoods $\vec{q}_k$ and $\vec{\bar{q}}_k = [\bar{q}_{nk}]_{N_Y \times 1}$ for each label $l_k$, since the pixels and their corresponding regions must have the self-consistent likelihoods. Therefore, the region likelihoods are defined as higher-order cues in order to learn the pixel likelihoods.

Main Idea: We first construct a bilayer graph $G = (\mathcal{V}, \mathcal{E})$ whose nodes $\mathcal{V}$ are the $N_X$ given pixels $X$ and the $N_Y$ over-segmented regions $Y = \{y_n\}_{n=1,...,N_Y}$, generated by an unsupervised segmentation algorithm such as Mean Shift [4], as shown in Fig. 3.5(b). The edges $\mathcal{E} = \{\mathcal{E}_X \cup \mathcal{E}_Y \cup \mathcal{E}_{XY}\}$ are linked by different criteria according to the node types. An undirected edge exists, if the one of the following conditions is
3.3. PROPOSED LIKELIHOOD ESTIMATION

Figure 3.5: An illustration of the proposed graph $G$. In $\mathcal{V}$, a node denotes a pixel $x_i \in \mathcal{X}$ (green circle) or a region $y_n \in \mathcal{Y}$ (violet circle). The regions $\mathcal{Y}$, whose boundaries are drawn in red color overlaid on the original image, are generated by an unsupervised image segmentation algorithm in (a). The undirected edges $\mathcal{E}$ represent the pairwise relationship between the nodes. (b) shows the edges connected to one region with violet lines. The edge weights are defined by the color differences between two nodes as in (2.4), (2.16), and (3.7) according to the node types.

1) In the pixel-based layer, an intralayer edge $e_{ij}^{X} \in \mathcal{E}_{X}$ between two neighboring pixels $x_i, x_j \in \mathcal{X}$ exists with the weight $w_{ij}^{X}$ in (2.4).

2) In the region-based layer, another intralayer edge $e_{nm}^{Y} \in \mathcal{E}_{Y}$ between two regions $y_n, y_m \in \mathcal{Y}$ which share a common boundary is connected with the following weight $w_{nm}^{Y}$ in (2.16).

3) An interlayer edge $e_{in}^{XY} \in \mathcal{E}_{XY}$ between a pixel $x_i \in \mathcal{X}$ and its corresponding
region $y_n \in \mathcal{Y}$ is added with the following weight $w_{in}^{XY} (= w_{ni}^{XY})$:

$$w_{in}^{XY} = \exp \left( -\theta_y \| \vec{g}_i - \vec{g}_n \| \right). \tag{3.7}$$

By these connections, it is possible to transfer the higher-order effect of the region-based cues into the pixel-based one. Simultaneously, each region can get the supplementary information of pixel-based cues.

We then build two quadratic cost functions for pixel and region likelihoods, which are related to each other in this graph, and simultaneously optimize them in a simple way.

A) We first compute the pixel likelihoods $\vec{q}_k$ by defining the region likelihoods $\vec{\bar{q}}_k$ as higher-order cues in a principled manner. By definition of the relationship between all nodes $\mathcal{V}$ in the graph $G$, the quadratic cost function $J_k^X$ of pixel likelihoods $\vec{q}_k$ with respect to a label $l_k$ is as follows.

$$J_k^X = E^X_{k,P} + E^X_{k,U} + E^X_{k,H}$$

$$= \sum_{i,j=1}^{N_X} w_{ij}^X (q_{ik} - q_{jk})^2 + \sum_{i=1}^{N_X} \lambda_i^X d_i^X (q_{ik} - q_{ik}^+)^2 + \sum_{i=1}^{N_X} \kappa d_i^X (q_{ik} - q_{ik}^+)^2, \tag{3.8}$$

where $q_{ik}^+$ corresponds to the likelihood of the region containing the pixel $x_i$ among $\vec{\bar{q}}_k$:

$$q_{ik}^+ = \sum_{n=1}^{N_Y} z_{in}^{XY} \bar{q}_{nk}, \tag{3.9}$$

where $z_{in}^{XY} = w_{in}^{XY} / \sum_{n=1}^{N_Y} w_{in}^{XY}$ with the weight $w_{in}^{XY}$ in (3.7). The pairwise $E^X_{k,P}$ and unary $E^X_{k,U}$ terms are same in (3.4). The third term $E^X_{k,H}$ in (3.8) is the higher-order region consistency by which a pixel likelihood should be similar to its corresponding region likelihood. Since the regions quite often contain pixels belonging to multiple labels, it partly enforces the label consistency inside the regions with a weight $\kappa$. This term is also multiplied by the value $d_i^X$ for the balance between two competing
3.3. PROPOSED LIKELIHOOD ESTIMATION

constraints $E_{k,P}^X$ and $E_{k,H}^X$, same to the second constraint $E_{k,U}^X$. In contrast to other segmentation algorithms which use the hard label consistency in regions on the assumption that all pixels constituting a particular region belong to the same label, our work uses this soft label consistency constraint, similarly to the Robust $P^n$ model of Kohli et al. [7]. Unlike this parametric model which is based on the number of pixels in the region not taking the dominant label, we estimate the region likelihoods $\vec{\bar{q}}_k$ by nonparametric learning from the given image.

B) To solve the formulation in (3.8), we also estimate the region likelihoods $\vec{\bar{q}}_k$ by referring the pixel likelihoods $\vec{q}_k$ in the graph $G$. In a similar way to $J_k^X$ in (3.8), the cost function $J_k^Y$ of region likelihoods $\vec{\bar{q}}_k$ is designed as follows.

$$J_k^Y = E_{k,P}^Y + E_{k,U}^Y + E_{k,R}^Y$$

$$= \sum_{n,m=1}^{N_X} w_{nm}^X (\vec{q}_{nk} - \vec{q}_{mk})^2 + \sum_{n=1}^{N_Y} \lambda_n^Y d_n^Y (\vec{q}_{nk} - \vec{q}_{nk}^*)^2 + \sum_{n=1}^{N_Y} \varepsilon d_n^Y (\vec{q}_{nk} - \bar{q}_{nk}^+)^2,$$

(3.10)

where $d_n^Y = \sum_{m=1}^{N_X} w_{nm}^X$ in (2.16). The parameter $\lambda_n^Y$ for the seeded regions $\bar{S}_k$, containing the seeded pixels in $S_k$, is $\lambda$ if $y_n \in \bar{S} = \{\bar{S}_k\}_{k=1,\ldots,K}$ and 0 otherwise. The region-seed likelihood $\vec{q}_{nk}^*$ is 1 if $y_n$ is the seed with the label $l_k$ and 0 otherwise.

The estimated likelihood $\vec{q}_{nk}^+$ of the region $y_n$ from the pixel likelihoods $\vec{q}_k$ is defined as the weighted average of its containing pixel likelihoods:

$$\vec{q}_{nk}^+ = \sum_{i=1}^{N_X} z_{ni}^{XY} \vec{q}_{ik},$$

(3.11)

where $z_{ni}^{XY} = w_{ni}^{XY} / \sum_{i=1}^{N_X} w_{ni}^{XY}$ with the weight $w_{ni}^{XY}$ in (3.7). In (3.10), the first term $E_{k,P}^Y$ is the label-continuity constraint that both regions in the full neighborhood system should have the same label if their representative colors are similar. Instead of the parametric color models such as Gaussian Mixture Model, as the colors of all regions are compared with those of all seeded regions through higher-order con-
nections, it gives the same effects as the nonparametric prior model. The second term \( E_{k,U}^Y \) is the unary constraint that each region seed tends to have the user-given label of inner seed pixels, similarly to \( E_{k,U}^X \) in (3.8). Finally, the third term \( E_{k,R}^Y \) is another estimated unary constraint whereby a region likelihood should be similar to the weighted average of inner pixel likelihoods. This term has the effect of refining the region likelihoods \( \vec{q}_k \) from more informative pixel likelihoods \( \vec{q}_k^* \), since it can not guarantee that the reliable regions are always extracted in highly textured environments.

**Formulation:** Since two cost functions \( J_X^k \) in (3.8) and \( J_Y^k \) in (3.10) are supplementary to each other, we should minimize them simultaneously. These functions can be formulated as the matrix forms with respect to the likelihoods \( \vec{q}_k \) and \( \vec{\bar{q}}_k \) from the initial seeds \( \vec{q}_k^* = [\pi_{ik}^*]_{N_X \times 1} \) and \( \vec{\bar{q}}_k^* = [\bar{q}_{nk}^*]_{N_Y \times 1} \) respectively, as follows.

\[
J_X^k = \vec{q}_k^T (D_X - W_X) \vec{q}_k + (\vec{q}_k - \vec{\bar{q}}_k)^T D_X A_X (\vec{q}_k - \vec{\bar{q}}_k)
\]

\[
+ \kappa (\vec{q}_k - Z_{XY} \vec{\bar{q}}_k)^T D_X (\vec{q}_k - Z_{XY} \vec{\bar{q}}_k)
\]

\[
J_Y^k = \vec{\bar{q}}_k^T (D_Y - W_Y) \vec{\bar{q}}_k + (\vec{\bar{q}}_k - \vec{\bar{\bar{q}}}_k)^T D_Y A_Y (\vec{\bar{q}}_k - \vec{\bar{\bar{q}}}_k)
\]

\[
+ \epsilon (\vec{\bar{q}}_k - Z_{YX} \vec{\bar{\bar{q}}}_k)^T D_Y (\vec{\bar{q}}_k - Z_{YX} \vec{\bar{\bar{q}}}_k)
\]

where \( W_X = [w_{ij}^X]_{N_X \times N_X} \) and \( W_Y = [w_{nm}^Y]_{N_Y \times N_Y} \). The elements of the matrices \( Z_{XY} = [z_{in}^{XY}]_{N_X \times N_Y} \) in (3.9) and \( Z_{YX} = [z_{mi}^{YX}]_{N_Y \times N_X} \) in (3.11) mean the label dependencies between the pixels and their corresponding regions in the graph \( G \). The diagonal elements of the matrices \( D_X = \text{diag}([d_1^X, ..., d_{N_X}^X]) \) and \( D_Y = \text{diag}([d_1^Y, ..., d_{N_Y}^Y]) \) are the degree values of the weight matrices \( W_X \) and \( W_Y \), respectively. The diagonal elements of the matrix \( A_X = \text{diag}([\lambda_1^X, ..., \lambda_{N_X}^X]) \) (or \( A_Y = \text{diag}([\lambda_1^Y, ..., \lambda_{N_Y}^Y]) \)
3.3. PROPOSED LIKELIHOOD ESTIMATION

are the parameter \( \lambda \) for the seeds \( S \) (or \( \tilde{S} \)) and 0 otherwise.

Differentiating these matrix formulations \( J^X_k \) and \( J^Y_k \) with respect to \( \tilde{q}_k \) and \( \tilde{q}_k \), respectively, and set to zero, we can get all likelihoods \( \tilde{q}_k = [\tilde{q}_k; \tilde{q}_k] \) from the initial values \( \tilde{q}_k^* = [\tilde{q}_k; \tilde{q}_k^*] \) of the label \( l_k \) simply by

\[
\frac{\partial J^X_k}{\partial \tilde{q}_k} = \tilde{q}_k - Z_X \tilde{q}_k + \Lambda_X (\tilde{q}_k - \tilde{q}_k^*) + \kappa (\tilde{q}_k - Z_{XY} \tilde{q}_k^*) = 0
\]

\[
\frac{\partial J^Y_k}{\partial \tilde{q}_k} = \tilde{q}_k - Z_Y \tilde{q}_k + \Lambda_Y (\tilde{q}_k - \tilde{q}_k^*) + \epsilon (\tilde{q}_k - Z_{YX} \tilde{q}_k^*) = 0,
\]

where \( Z_X = D^{-1}W_X \) and \( Z_Y = D^{-1}W_Y \). It can be jointly transformed into

\[
\tilde{q}_k = (I - \Omega)Z \tilde{q}_k + \Omega \tilde{q}_k^*;
\]  

(3.12)

or simply

\[
(I - (I - \Omega)Z)\tilde{q}_k = \Omega \tilde{q}_k^*;
\]  

(3.13)

where the matrix \( Z = \begin{bmatrix} \tilde{\kappa} Z_X & (1 - \tilde{\kappa}) Z_{XY} \\ (1 - \tilde{\epsilon}) Z_{YX} & \tilde{\epsilon} Z_Y \end{bmatrix} \) is a row-normalized one with \( \tilde{\kappa} = \frac{1}{1 + \kappa} \) and \( \tilde{\epsilon} = \frac{1}{1 + \epsilon} \). To solve this matrix formulation in (3.13), we use three weight parameters \( \lambda, \kappa \) and \( \epsilon \) as the diagonal matrix form:

\[
\Omega = \begin{bmatrix} \frac{\Lambda_X}{(1 + \lambda)(1 + \kappa)} & \frac{\Lambda_Y}{(1 + \lambda)(1 + \epsilon)} \\ \frac{\Lambda_X}{(1 + \lambda)(1 + \kappa)} & \frac{\Lambda_Y}{(1 + \lambda)(1 + \epsilon)} \end{bmatrix}.
\]

Since \( I - (I - \Omega)Z \) is positive definite, the linear equation in (3.13) can be solved easily and we finally have the likelihoods \( \tilde{q}_k \) of all pixels and all regions by a sparse matrix inversion technique as follows.

\[
\tilde{q}_k = (I - (I - \Omega)Z)^{-1}(\Omega \tilde{q}_k^*).
\]  

(3.14)

This form is similar to the solutions of the semi-supervised learning techniques [11] in the data mining.
Figure 3.6: Example segmentations by solving Eq. (3.14) with one over-segmentation of each image in Fig. 3.4(a). (a) Over-segmentations. (b) and (c) Likelihoods $\{\vec{q}_k\}$ for “red” and “green” labels, respectively. (e) Segmentation results by assigning each pixel the label with the largest likelihood.

Fig. 3.6 shows four example segmentations by solving Eq. (3.14) with one over-segmentation in Fig. 3.6(a). By making the likelihood of each region and the likelihoods of its inner pixels similar, the label consistency inside the regions can be emphasized. However, if the unreliable regions exists, they may cause some errors such as in Fig. 3.6(d).

3.3.3 Using Multiple Over-Segmentations

Let us consider the use of regions obtained from multiple over-segmentations. Since the shapes of some regions may be inconsistent with the real object boundaries
3.3. PROPOSED LIKELIHOOD ESTIMATION

Figure 3.7: Multilayer graph using multiple over-segmentations. A set of regions $\mathcal{Y}$ consists of the sum of the regions obtained from multiple over-segmentations in (a). (b) and (c) show the edges connected to one same region and one same pixel with violet lines, respectively.

In only one over-segmentation due to noise and parameter values, the uncertainty about the region shapes can be reduced by using the multiple over-segmentations with different parameter sets together. We simply extend our algorithm to exploit multiple segmentations as shown in Fig. 3.7. The formulation to estimate the pixel and region likelihoods is the same as two cost functions in (3.8) and (3.10) with just minor changes of the criteria for generating the edges in the graph. Note that unlike single region layer case where each pixel is connected to only one region in Fig. 3.5, in multiple over-segmentations all edges between a pixel and multiple regions containing it are added as depicted in Fig. 3.7. Therefore the pixel likelihood depends on the weighted average of the multiple region likelihoods, not only one
Figure 3.8: Example segmentations by solving Eq. (3.14) with three over-segmentations in Fig. 3.4(a). (a) Multiple over-segmentations, where red represents larger probability that the edge is a part of object boundary. (b) and (c) Likelihoods $\{\vec{q}_k\}$ for “red” and “green” labels, respectively. (e) Segmentation results by assigning each pixel the label with the largest likelihood.

Fig. 3.8: Example segmentations by solving Eq. (3.14) with three over-segmentations in Fig. 3.8(a). By using multiple over-segmentations, various edge information can be imposed into our algorithm. Therefore, we can refine the likelihood maps (e.g. Fig. 3.8(b) and (c)), and improve the segmentation results based on these likelihoods (e.g. Fig. 3.8(d)).
3.4 Experimental Results

Our algorithm is based on the multiple regions. Therefore the overall qualities of the regions exert influence on our segmentation results. We also need three parameters \( \lambda, \kappa \) and \( \epsilon \) in two cost functions (3.8) and (3.10). \( \lambda \) is set to a high value \( (= 10^5) \), since each seed should be assigned to the initial label. We use the 8-neighborhood system in the pixel layer. In this section, we first analyze the effect of initial regions and other parameters \( (\kappa \text{ and } \epsilon) \), and then, we compare the performance of our algorithm with the state-of-the-art methods \([3, 5, 7]\) on several natural images.
3.4.1 Parameter Settings

We now explain how multiple regions $\mathcal{Y}$ were generated. In this work, we initially extract the regions by the Mean Shift image segmentation algorithm [4]. The Mean Shift algorithm uses two bandwidth parameters ($h_s, h_r$) for the spatial and range domains, respectively. Our algorithm is based on multiple over-segmentations as shown in Fig. 3.7. Following [7], we choose to generate multiple regions by varying the Mean Shift parameters. The over-segmentations with different parameters ($h_s, h_r$) are shown in Fig. 3.9(b)-(d). By increasing the range parameter $h_r$ we can get a set of regions which vary from over-segmented to under-segmented. Fig. 3.9(e)-(g) show that the resulting segmentations vary according to which regions are used. In smaller regions, more detailed boundaries are extracted but more errors may be generated, since their features are less informative. In this work, our experiments use all three over-segmentations with $(h_s, h_r) = \{(10, 7), (10, 10), (10, 15)\}$ for producing the final flexible segmentations such as in Fig. 3.9(h).

Fig. 3.10 and Fig. 3.11 show the segmentation results by varying the other...
3.4. EXPERIMENTAL RESULTS

Figure 3.11: Examples of the segmentations with respect to the variation of $\epsilon$ in (3.10). From the images (a) with scribbles with two red and green labels, (b)-(d) give the resulting segmentations according to $\epsilon$. ($\kappa = 0.002$)

parameters $\kappa$ and $\epsilon$ in (3.8) and (3.10), respectively. In a larger $\kappa$, the region label consistency is emphasized more and the boundaries of final segmentations are more consistent with the initial regions as shown in Fig. 3.10. In a smaller $\kappa$, more smooth boundaries are extracted around the seed placement. When the faint object boundaries exist as shown in the bottom image of Fig. 3.10(a), it is important to find appropriate $\kappa$ to reduce dependence on the boundaries of the inaccurate regions and to alleviate the over-smoothing effect. $\epsilon$ controls the weight of the region unary term estimated from the pixel likelihoods in (3.10). Compared with $\kappa$, the resulting segmentations are less sensitive to $\epsilon$ in Fig. 3.11. Thus it is more important to decide the parameter $\kappa$ in this framework. In this work, $\kappa$ and $\epsilon$ were chosen empirically, and we set $\kappa = 0.002$ and $\epsilon = 0.2$ for all the test images.

3.4.2 Segmentation Results

We first demonstrate the quality of our proposed algorithm on the Microsoft GrabCut database. As the measure for the accuracy between two segmentations, a nor-
CHAPTER 3. ROBUST HIGHER-ORDER INTERACTIVE SEGMENTATION

<table>
<thead>
<tr>
<th>Segmentation model</th>
<th>Accuracy ($a_o$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GC [5]</td>
<td>0.9559</td>
</tr>
<tr>
<td>GD [2]</td>
<td>0.9517</td>
</tr>
<tr>
<td>RW [3]</td>
<td>0.9552</td>
</tr>
<tr>
<td>GMM-GC</td>
<td>0.9612</td>
</tr>
<tr>
<td>gRWR [6]</td>
<td>0.9581</td>
</tr>
<tr>
<td>Robust $P^n$ model [7]</td>
<td>0.9602</td>
</tr>
<tr>
<td>Our method (None)</td>
<td>0.9510</td>
</tr>
<tr>
<td>Our method (one over-segmentation)</td>
<td>0.9700</td>
</tr>
<tr>
<td>Our method (three over-segmentations)</td>
<td>0.9709</td>
</tr>
</tbody>
</table>

Table 3.1: Segmentation accuracy $a_o$ in the Microsoft GrabCut database.

Normalized overlap $a_o = \frac{|R \cap R'|}{|R \cup R'|}$ was used [38], where $R$ and $R'$ indicate the sets of pixels assigned as the foreground in two segmentations. Table 3.1 presents the comparative evaluation of the interactive segmentation algorithms: GC [5], GD [2], RW [3], GC using GMM (GMM-GC), gRWR introduced in Section 2.3.4, Robust $P^n$ Model [7], and our proposed algorithm. Our algorithm quantitatively has better performance than conventional pairwise and higher-order algorithms. Also, our likelihood model produces better performance than the GMM-based likelihood model. Fig. 3.12 illustrates the example segmentations on the Microsoft GrabCut database. This figure shows that our algorithm intuitively produces high-quality segmentation results. For example, our segmentation result in the second row image of Fig. 3.12(e) well indicates that the color information of face seeds gives aid to the detection of human ears and neck through our higher-order connections, without parametric color
3.4. EXPERIMENTAL RESULTS

Figure 3.12: Example segmentations using the tri-maps on the Microsoft GrabCut database. (a) Test image with tri-map. Segmentation results by (b) GMM-GC [5], (c) gRWR [6], (d) Robust $P^n$ model [7], and (e) Our algorithm.

We compare the sensitivity of our algorithm with respect to seed quantity and placement, similarly to the evaluation of Sinop et al. [38]. The sensitivity check has the following procedure as shown in Fig. 3.13. We first chose one test image in Fig. 3.13(a) with a tri-map in Fig. 3.13(b). In Fig. 3.13(d)-(g), the standard segmentations were produced from this initial tri-map by GC, RW, Robust $P^n$ model, and our algorithm. Then, some seeds were randomly taken from 1% to 50%
CHAPTER 3. ROBUST HIGHER-ORDER INTERACTIVE SEGMENTATION

Figure 3.13: Description of the sensitivity check with respect to seed quantity and placement. (a) Test image with an initial tri-map (b). (c) Results of the sensitivity check by varying the perturbed seed quantity: 1%, 10%, 30%, 50% of total seed quantity. (d)-(g) Standard segmentations produced from the initial tri-map, and (h)-(k) Perturbed segmentations produced from the randomly chosen 1% seeds by GC, RW, Robust $P^n$ Model, and our algorithm respectively.

The perturbed segmentations were recomputed from these selected seeds in Fig. 3.13(h)-(k), and compared with the standard segmentations. This procedure was repeated 10 times on the same problem set. We finally checked the sensitivity of the algorithms as mean normalized overlap in Fig. 3.13(c). This experiment quantitatively shows that our algorithm is less dependence on user-inputs than other state-of-the-art algorithms.
3.4. EXPERIMENTAL RESULTS

Figure 3.14: Comparison of our algorithm with the state-of-the-art algorithms for finding multiple patterns or objects with a few scribbles. (a) Input images with scribbles with two red and green labels. Segmentation results by (b) RW [3] with pairwise potentials, (c) Robust $P^n$ model [7] with parametric higher-order potentials, and (d) Our algorithm.

Fig. 3.14 presents the higher-order effect of our algorithm in the natural images, compared with the state-of-the-art algorithms. The full connections between regions cause the propagation of local grouping cues across the whole image. Our algorithm is useful for the detection of multiple patterns or objects with a few scribbles without parametric color models. These properties give less sensitivity to seed quantity and
Figure 3.15: Visual comparison of our algorithm with the state-of-the-art algorithms in natural images. (a) Inputs with scribbles with two red and green labels. Segmentation results by (b) RW [3], (c) Robust $P^n$ model [7], and (d) Our algorithm.

placement as well as the high-quality segmentations. Compared with the conventional higher-order models like Robust $P^n$ model [7], our algorithm still has these
advantages. It is because it is difficult to propagate the local grouping cues across larger image regions without additional information such as parametric prior models in previous higher-order models. Additional visual comparisons of segmentation results are shown in Fig. 3.15.

3.4.3 Complexity Consideration

The $N \times N$ matrix $(I - (I - \Pi)Z)$, where $N = N_X + N_Y$, in (3.14) is an sparse matrix, since the pixel number $N_X$ is very much larger than the region number $N_Y$ and the links between pairs of pixels are very sparse. Thus its inversion typically has an efficient computation. In our MATLAB implementation, the division operator ‘\’ which executes a fast LU decomposition to solve the linear system was used for the inversion of $(I - (I - \Pi)\Omega)$. In the top image (size: $213 \times 320$) of Fig. 3.1(a), the computation time of our algorithm is about 5.25 sec (except the over-segmentation stage) on an Intel Core 2 Quad CPU running at 2.4 GHz and the memory usage is about 220 MByte in MATLAB. For comparison purpose, [3] takes about 1.11 sec and needs about 60 MByte.

3.5 Conclusion

We present a new generative model for interactive segmentation. More importantly, we propose the new higher-order cues defined in the regions. These higher-order cues are efficiently used to estimate the pixel likelihoods in the quadratic formulation, and recursively estimated from the resulting likelihoods of pixels included in each region. Since in this manner all regional cues are propagated into the whole image by using the full connections between regions, it gives the same effect as the nonparametric
prior model. The performance of our algorithm was tested on the challenging data sets. The segmentation results demonstrate that our higher-order cues are helpful in producing high-quality segmentations with detailed boundaries and reducing the sensitivity of the seed quantity and placement.

In our framework, two parameters $\kappa$ and $\epsilon$ were empirically chosen. They are not, however, optimal for every image. Moreover we used only color values as the pixel and region properties. If we use the optimal parameters $\kappa$ and $\epsilon$ and other local grouping cues such as texture and boundary, better segmentation results will be obtained. Therefore our future work will include the automatic parameter selection and the fusion of multiple grouping cues.
Chapter 4

Full Affinity Model for Spectral Segmentation

4.1 Introduction

In recent years, spectral segmentation has become a major trend in unsupervised image segmentation. It typically starts from local information encoded in a graph-based representation of a given image, and partitions it according to the global information embedded in the spectrum of the graph affinity matrix. Most methods \cite{8,9} take $K$ eigenvectors of the affinity matrix, and then invoke another technique such as $K$-means to cluster pixels by their respective $K$ components in the eigenvectors. However, these methods often break uniform regions where the eigenvectors have smooth gradients. To circumvent this difficulty, other methods \cite{28} construct a hierarchy of regions that is consistent with the contour signals in the eigenvectors themselves.

In natural images, many problems make segmentation difficult, such as deter-
mining faint object boundaries and separating highly cluttered backgrounds. As a popular way to solve these problems, several recent approaches [8, 72] have attempted to integrate local grouping cues across long-range connections in image space. Although their segmentation results are usually impressive, the direct use of long-range connections entails great computational costs. Therefore, the use of approximate techniques to propagate local grouping cues into larger image areas has been proposed.

1) **Sampling-based Model:** In [73], the Nyström method to find numerical approximations to large-scale problems allows one to extrapolate the complete solution using a small random subset of pixels. Although this sampling-based model substantially reduces the computational costs, its accuracy is affected by the random sample selection.

2) **Region-oriented Model:** In [25, 65, 74], the segmentation results are obtained by grouping non-overlapping regions, instead of pixels. This region-oriented model is inspired by the hard constraint that the pixels in a particular region should have the same label. It has the advantage of using more informative features extracted from the inner pixels of a region, as well as of transferring local grouping cues to a larger image area with links across regions. However, since it enforces region constraints strongly, the segmentation results depends highly on the initial region boundaries.

3) **Multiscale Model:** In [8, 75, 76], the segmentation methods are based on multiscale model to combine both coarse- and fine-level image details. Although this model gives efficient approximation for the incorporation of long-range connections with low complexity, due to inherent coarsening error, fine-level details along object boundaries are not well recovered.
4.1. INTRODUCTION

Figure 4.1: Learning pairwise affinities. (a) Image with a user-given pixel point +. (b)-(d) Affinity images between + and other pixels in the image, where red represents a greater affinity. These affinities are obtained by the general color+boundary affinity model used in MNCut [8], our proposed model, and the ground-truth affinity function from human annotations, respectively.

4) Multilayer Model: In [7,31], the multilayer graph was designed. In each layer, the nodes consist of pixels or over-segmented regions by MShift with different parameters. By defining the intra- and inter-layer affinities between graph nodes, local grouping cues can be propagated over a large area. However, since using long-range affinities leads to excessive complexity, the range of affinities is generally limited.

Unlike the above mentioned approximation models, we try to design and use a full range of affinity model in the spectral segmentation framework, so that we can obtain high-quality segmentation results efficiently by using the proposed affinity measure. The key contributions of our algorithm are as follows.

1) A new affinity model for spectral segmentation is introduced. Unlike previous models learned from a large labeled dataset [72], or from only local properties of adjacent pixels [8,77], using the relevance scores between all pairs of pixels, estimated by semi-supervised learning (SSL) [11,54,78], as affinities is proposed. A multilayer
Figure 4.2: Introducing $K$-way segmentation. (a) Original image. (b)-(d) $K$-way partitioning by NCut [9], MNCut [8], and our algorithm FNCut with the boundaries drawn in red, respectively.

Figure 4.3: Introducing hierarchical segmentation. (a)-(c) Hierarchical segmentations of an input image in Fig. 4.2(a) by Canny-OWT-UCM, gPb-OWT-UCM [10], and our algorithm fPb-OWT-UCM, respectively. (d) Human boundaries. Darker color means greater edgeness.

to the over-segmented regions by MShift as well as the pixels, is first constructed. Then, the affinities between each node and other nodes are estimated by applying the SSL strategy to this graph through assuming the current node as labeled data and the others as unlabeled data. Fig. 4.1 clearly shows that our model intuitively provides much better affinities in highly textured images,
4.1. **INTRODUCTION**

compared with the general affinity model [8] that heuristically unifies the color and boundary cues.

2) Spectral analysis of our full affinity matrix is done efficiently. In general, performing spectral analysis of a full large matrix requires a prohibitively expensive computation. However, since our full affinity matrix is expressed as the inverse of a sparse matrix, its eigen-decomposition is very efficient using the basics of matrix computation [79].

3) We introduce two types of spectral segmentation algorithms based on our affinity matrix: $K$-way segmentation [9, 80, 81] and hierarchical segmentation [13]. For $K$-way segmentation, our algorithm FNCut clusters all pixels and regions simultaneously into the $K$ visually coherent groups in a single multilayer framework of Normalized Cuts. For hierarchical segmentation, our algorithm fPb-OWT-UCM produces a hierarchy of regions from the contour signals in the eigenvectors using a sequence of two transformations: Oriented Watershed Transform (OWT) [82] and Ultrametric Contour Map (UCM) [83].

4) Our spectral segmentation algorithms: FNCut and fPb-OWT-UCM, produce high-quality segmentation results by considering all intra and interlayer affinities. Fig. 4.2 shows that FNCut produces much better $K$-way segmentations with object details than other spectral segmentation methods such as Normalized Cuts (NCut) [9] and Multiscale NCut (MNCut) [8]. Fig. 4.3 shows a comparison of fPb-OWT-UCM with other existing hierarchical segmentation methods based on the sequence of operations OWT and UCM: Canny-OWT-UCM and gPb-OWT-UCM [13] that use the Canny edge detector (Canny) [84] and the global boundary detector (gPb) [10] for contour detection, respectively. The results by fPb-OWT-
4.2 Previous Affinity Models

In the spectral segmentation algorithms, defining the affinity model gained by integrating local grouping cues such as color and boundary is important. Now, we review several previous affinity models, except the learned ones [72] from a large
4.2. PREVIOUS AFFINITY MODELS

training dataset with the manually segmented images. The affinity $w_{ij}$ between two pixels $i$ and $j$ is modeled according to the grouping cues used as follows:

1) **Color-based Model:** Close-by pixels with similar colors likely belong to the same segment. The color-based affinity model $\varphi_c$ is usually formulated by \[9\] \[1\]

$$w_{ij} = \varphi_c(i, j) = \exp \left( -\theta_x \| \vec{x}_i - \vec{x}_j \|^2 - \theta_g \| \vec{g}_i - \vec{g}_j \|^2 \right),$$ \hfill (4.1)

where $\vec{x}_i$ and $\vec{g}_i$ denote the position and color vectors of pixel $i$, respectively. Fig. 4.4(b) shows that connecting pixels by color is useful when linking disjointed object parts. However, this process results in errors if the background has a similar color distribution with the object parts.

2) **Boundary-based Model:** Edgeness is an important cue to detect a potential object boundary. The boundary-based affinity model $\varphi_b$ is commonly formulated by measuring the edge magnitude between two pixels [77]:

$$w_{ij} = \varphi_b(i, j) = \exp \left( -\theta_x \max_{\nu \in \vec{ij}} \| \kappa_{i\nu} \| ^2 \right),$$ \hfill (4.2)

where $\vec{ij}$ is a straight line joining two pixels $i$ and $j$, and $\kappa_{i\nu}$ is the edge strength of pixel $i$. This boundary-based model is particularly useful when the background clutter has a similar color with the object body, such as in Fig. 4.4(c). However, since it is based on the edgeness along the straight line between two pixels without considering all possible paths in image space, texture edges often disturb the affinity estimation.

3) **Combined Model:** To design a better affinity model for all natural images, the combination of the color and boundary cues is helpful. They can be simply combined with a parameter $\alpha$ for the combined affinity model $\varphi_m$ as follows [8].

$$w_{ij} = \varphi_m(i, j) = \sqrt{\varphi_c(i, j) \times \varphi_b(i, j)} + \alpha \cdot \varphi_b(i, j),$$ \hfill (4.3)
CHAPTER 4. FULL AFFINITY MODEL FOR SPECTRAL SEGMENTATION

Figure 4.5: Our multilayer graph $G = (\mathcal{V}, \mathcal{E})$. In (a), the graph nodes $\mathcal{V}$ consist of pixels $\mathcal{X}$ and regions $\mathcal{Y} = \{Y_1 \cup ... \cup Y_L\}$. The undirected edges $\mathcal{E}$ represent the relationship between pairs of nodes. (b) and (c) show the examples of edges (violet lines) connected to one pixel and to one region, respectively.

where there are two models $\varphi_c$ and $\varphi_b$ in (4.1) and (4.2), respectively. Fig. 4.4(d) shows some affinity images gained through this combined model. However, the model $\varphi_m$ still has some weaknesses in long-range affinity estimation, since it is formulated by naively mixing two color and boundary affinity models.

4.3 Proposed Full Affinity Model

To estimate the full pairwise affinities by integrating local grouping cues extracted from the entire image, we propose to use the SSL method [11, 54, 78], which is a very successful technique to learn the global relevance scores between labeled
4.3. PROPOSED FULL AFFINITY MODEL

and unlabeled data in a sparse graph. We first design a sparsely connected graph with multilayers for efficiently combining local grouping cues. The relevance scores defined by [11] between all pairs of the graph nodes are then used as the full affinities.

4.3.1 Graph Design

Let us construct a multilayer graph \( G = (\mathcal{V}, \mathcal{E}) \), where the nodes \( \mathcal{V} = \{\mathcal{X} \cup \mathcal{Y}\} \) consist of a set of pixels \( \mathcal{X} \) and the sets of regions \( \mathcal{Y} = \{\mathcal{Y}_1 \cup ... \cup \mathcal{Y}_L\} \), and the edges \( \mathcal{E} \) are the undirected links, as shown in Fig. 4.5. The node subset \( \mathcal{Y}_l \) contains the \( N_l \) non-overlapping regions, generated by an unsupervised segmentation algorithm such as MShift [4], at the \( l \)-th region layer. In this work, \( L \) different over-segmentations are used, since it is known that the use of multiple overlapped regions can reduce the errors of those regions that may contain many objects [7]. The edges \( \mathcal{E} \) are linked by different criteria according to the node types. An undirected edge \( e_{ij} \in \mathcal{E} \) exists if one of the following conditions is satisfied:

1) If two pixels \( i, j \in \mathcal{X} \) are adjacent, usually in the 4 or 8 neighborhood system, an intralayer edge \( e_{ij} \) between \( i \) and \( j \) exists with the following weight \( \omega_{ij} \):

\[
\omega_{ij} = \exp \left( -\theta_g \| \vec{g}_i - \vec{g}_j \| \right),
\]

where \( \theta_g \) is a constant that controls the strength of the weight.

2) If a region \( i \) and its adjacent region \( j \) share a common boundary at the same region-based layer (\( i, j \in \mathcal{Y}_l \)), another intralayer edge \( e_{ij} \) is linked with the following weight \( \omega_{ij} \):

\[
\omega_{ij} = \exp \left( -\theta_g \| \vec{g}_i - \vec{g}_j \| \right),
\]

where \( \vec{g}_i \) denotes the mean color vector of the inner pixels of region \( i \).
3) If a pixel $i \in \mathcal{X}$ is included in its corresponding region $j \in \mathcal{Y}_l$, an interlayer edge $e_{ij}$ is added with the following weight $\omega_{ij}$:

$$\omega_{ij} = \tau,$$  \hspace{1cm} (4.6)

where $\tau$ is a parameter that controls a positive correlation between pixel- and region-based layers.

The intralayer weights in (4.4) and (4.5) encode the color cues in Lab color space, similarly in (4.1). Since the correlation between the pixels in the same region is more emphasized through the interlayer connections between the region and its inner pixels in (4.6), the region itself provides implicit boundary information as the boundary cue, instead of image edges such as in (4.2). Therefore the trade-off between our color and boundary cues is captured by a parameter $\tau$.

### 4.3.2 Learning Full Affinities

Let $\mathbf{P} = [\pi_{ij}]_{N \times N}$, where $N = |\mathcal{V}|$ is the number of nodes, denote the affinity matrix. To learn the full affinities $\mathbf{P}$, we borrow ideas from SSL. Each node $m$ and all other nodes $\mathcal{V} - \{m\}$ are first assigned as labeled and unlabeled nodes, respectively, and then the relevance scores between labeled and unlabeled nodes are determined by applying the SSL strategy to our graph $G$. We propose to use these scores as the full affinities $\vec{\pi}_m = [\pi_{im}]_{N \times 1}$ between that node $m$ and all nodes $\mathcal{V}$. In this work, two popular SSL functions $\psi_{no}$ and $\psi_{un}$ are discussed as found in the work of Zhou et al. [11]. They can make the symmetric affinity matrix $\mathbf{P}$ whose elements are sufficiently smooth with respect to the intrinsic structure of the graph.

1) $\psi_{no}$: First, the affinity vector $\vec{\pi}_m$ of all nodes $\mathcal{V}$ from a node $m$ can be formulated:

$$\vec{\pi}_m = [\psi_{no}(\cdot, m)]_{N \times 1} = c (I - (1 - c)\mathbf{P})^{-1} b_m,$$  \hspace{1cm} (4.7)
where \( I \) is the identity matrix of size \( N \), and \( P \) is the bi-normalized adjacency matrix \( \Omega = [\omega_{ij}]_{N \times N} \) in (4.4)-(4.6): \( P = D^{-\frac{1}{2}}\Omega D^{-\frac{1}{2}} \), where the degree matrix 
\( D = \text{diag}(\{d_1, \ldots, d_N\}) \) is diagonal with \( d_i = \sum_{j=1}^{N} \omega_{ij} \). The vector \( \vec{b}_m = [b_{im}]_{N \times 1} \) indicating labeled node \( m \) is \( N \)-dimensional with \( b_{im} = 1 \) if labeled node \( i = m \) and \( 0 \) otherwise.

The vector \( \vec{\pi}_m \) in (4.7) can be interpreted as the solution to minimize the following cost function \( E_{no}(m) \):

\[
E_{no}(m) = \sum_{i,j=1}^{N} \omega_{ij} \left| \frac{\pi_{im}}{\sqrt{d_i}} - \frac{\pi_{jm}}{\sqrt{d_j}} \right|^2 + \mu \sum_{i=1}^{N} |\pi_{im} - b_{im}|^2,
\]

where \( \mu = \frac{c_1}{1-c} \), \( 0 < c < 1 \). The first term in \( E_{no}(m) \) is the smoothness constraint in which good affinities should not change too much between neighboring nodes. However, this term is defined as local variation by the difference in normalized affinities between two end-nodes of each edge. Thus, the smooth affinity variation between close-by nodes cannot be guaranteed, as shown in Fig. 4.6(e)-(h). The second term in \( E_{no}(m) \) is the fitting constraint in which good affinities should not change too much from the initial label assignment. A positive parameter \( \mu \) (or \( c \)) specifies the relative amount of smoothness and fitting constraints.

2) \( \psi_{un} \): Second, the affinity vector \( \vec{\pi}_m \) of all nodes \( V \) from a node \( m \) is defined as

\[
\vec{\pi}_m = [\psi_{un}(\cdot, m)]_{N \times 1} = c(D - (1 - c)\Omega)^{-1}\vec{b}_m.
\]

The vector \( \vec{\pi}_m \) in (4.9) is the same as the solution to minimize the following cost function \( E_{un}(m) \), similar to \( E_{no}(m) \) in (4.8):

\[
E_{un}(m) = \sum_{i,j=1}^{N} \omega_{ij} |\pi_{im} - \pi_{jm}|^2 + \mu \sum_{i=1}^{N} d_i \left| \frac{\pi_{im} - b_{im}}{d_i} \right|^2.
\]

The function \( E_{un}(m) \) also consists of smoothness and fitting constraints. However, compared with \( E_{no}(m) \) in (4.8), the smoothness constraint in \( E_{un}(m) \) is more helpful.
in estimating the pairwise affinities with local smooth variation, as shown in Fig. 4.6(i)-(l). Moreover, according to the increase in the number of over-segmentations, the intrinsic image structure with object details is well represented in the affinity image by solving the affinity function in (4.9).

Although the function $\psi_{no}$ in (4.7) is known to be better than $\psi_{un}$ in (4.9) for classification problems [11], $\psi_{un}$ is perceptually more suited for the affinity estimation. The quantitative results of these two functions will be compared in the experiments. Therefore, in this work the affinity matrix $\Pi = [\pi_{ij}]_{N \times N}$ is defined by using $\tilde{\pi}_m$ in (4.9), as follows:

$$\Pi = [\tilde{\pi}_1, ..., \tilde{\pi}_N] = [\psi_{un}(i, j)]_{N \times N}$$

$$= c(D - (1 - c)\Omega)^{-1} = cZ^{-1},$$

(4.11)

where $Z = D - (1 - c)\Omega$ is positive definite. Since the matrix $Z$ is sparse but very large, directly inversing it without any approximation technique is generally difficult. Fortunately, however, our full affinity matrix $\Pi$ in (4.11) can be efficiently used in a spectral framework.

4.3.3 Spectral Analysis

Let $D_\Pi = \text{diag}([d_{i1}^\Pi, ..., d_{NN}^\Pi])$, whose diagonal element is $d_{ii}^\Pi = \sum_{j=1}^{N} \pi_{ij}$, be the degree matrix of our affinity matrix $\Pi$. The following generalized eigenvalue problem based on the affinity matrix $\Pi$ is generally solved:

$$(D_\Pi - \Pi)\tilde{v}_k = \lambda_k D_\Pi \tilde{v}_k.$$  

(4.12)

Here, the generalized eigenvectors $\{\tilde{v}_k\}_{k=1,...,K}$, corresponding to the $K$ smallest eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_K$ can be utilized for $K$-way segmentation [9] and
4.3. PROPOSED FULL AFFINITY MODEL

Figure 4.6: Examples of our affinity images using SSL [11]. (a) Image with one pixel + selected. (b)-(d) Three over-segmentations, obtained by varying the MShift parameters. (e)-(h) and (i)-(l) Affinity images \( \tilde{\pi}_+ \) by \( \psi_\text{no} \) in (4.7) and by \( \psi_\text{un} \) in (4.9) with none, one (b), two (b)-(c), and three (b)-(d) over-segmentations, respectively.
hierarchical segmentation [13]. In general, the eigen-decomposition of the large-size full matrix requires a great computation, which is undesirable. However, since the proposed full affinity matrix $\Pi$ is expressed as the inverse of a sparse matrix $Z$ in (4.11), efficiently solving the system in (4.12) is possible. We first transform this system into a standard eigensystem as follows:

$$D_{\Pi}^{-\frac{1}{2}} \Pi D_{\Pi}^{-\frac{1}{2}} \vec{q}_k = (1 - \lambda_k) \vec{q}_k,$$

(4.13)

where $\vec{q}_k = D_{\Pi}^{-\frac{1}{2}} \vec{v}_k$ is an eigenvector of $D_{\Pi}^{-\frac{1}{2}} \Pi D_{\Pi}^{-\frac{1}{2}}$. By substituting $\Pi$ in (4.11) and using a simple spectral theory [79], we then rewrite Eq. (4.13) as the following system based on the sparse matrix $Z$:

$$D_{\Pi}^{\frac{1}{2}} Z D_{\Pi}^{\frac{1}{2}} \vec{q}_k = \varepsilon_k \vec{q}_k,$$

(4.14)

where $\varepsilon_k = \frac{c}{\lambda_k}$ is an eigenvalue of $D_{\Pi}^{\frac{1}{2}} Z D_{\Pi}^{\frac{1}{2}}$, and the diagonal elements of $D_{\Pi}$ are computed by

$$[d_{\Pi}^1, ..., d_{\Pi}^N]^T = \Pi \vec{1}_{N \times 1} = c Z^{-1} \vec{1}_{N \times 1},$$

(4.15)

where the multiplication of $Z$’s inversion by the all-ones vector $\vec{1}_{N \times 1}$ can be efficiently done using the linear system solver implemented by the MATLAB division operator ‘\’. Thus, the $K$ smallest eigenvalues $\{\varepsilon_k\}_{k=1,...,K}$ and their corresponding eigenvectors $\{\vec{q}_k\}_{k=1,...,K}$ of the very sparse matrix $D_{\Pi}^{\frac{1}{2}} Z D_{\Pi}^{\frac{1}{2}}$ in (4.14) are determined, instead of the eigen-decomposition of the very dense and large matrix $D_{\Pi}^{-\frac{1}{2}} \Pi D_{\Pi}^{-\frac{1}{2}}$ in (4.13). Finally, we find the generalized eigenvalue $\{\lambda_k\}_{k=1,...,K}$ and their corresponding eigenvectors $\{\vec{v}_k\}_{k=1,...,K}$ in (4.12) using $\lambda_k = 1 - \frac{c}{\varepsilon_k}$ and $\vec{v}_k = D_{\Pi}^{-\frac{1}{2}} \vec{q}_k$, respectively. The process of spectral analysis on our full affinity matrix $\Pi$ is briefly summarized in Algorithm 1.
Algorithm 1 Proposed Spectral Analysis

1: Given an image, construct a multilayer graph $G$, and calculate its weight matrix $\Omega$ using Eq. (4.4)-(4.6).

2: Find the $K$ smallest eigenvalues $\{\varepsilon_k\}_{k=1,...,K}$ and their corresponding eigenvectors $\{\vec{q}_k\}_{k=1,...,K}$ in (4.14) using our affinity matrix $\Pi$ in (4.11) and its degree matrix $D\Pi$ in (4.15).

3: Estimate the generalized eigenvalues $\{\lambda_k\}_{k=1,...,K}$ and eigenvectors $\{\vec{v}_k\}_{k=1,...,K}$ in (4.12) using $\lambda_k = 1 - \frac{1}{\varepsilon_k}$ and $\vec{v}_k = D\Pi^{\frac{1}{2}} \vec{q}_k$, respectively.

Fig. 4.7 presents the average running time comparison with respect to the number of pixels and some examples of the generalized eigenvectors, produced by solving Eq. (4.12). Spectral analysis of our full matrix $\Pi$, which additionally computes the degree matrix $D\Pi$ in (4.15), takes a little more time than spectral analysis obtained by defining the sparse weight matrix $\Omega$ as affinity matrix, as shown in Fig. 4.7(b). However, the eigenvectors based on our full affinity matrix $\Pi$ in Fig. 4.7(e)-(f) vary in large uniform areas less than those based on the very sparse matrix $\Omega$ in Fig. 4.7(c)-(d). They also contain more detailed object boundaries such as the tree’s branches in the bottom images of Fig. 4.7(e)-(f).

4.4 Spectral Segmentation

We present two types of spectral segmentation algorithms based on our full affinity matrix:

1) K-way Segmentation: We borrow the basic idea of Normalized Cuts [9], which associates with each pixel a length $K$ descriptor formed from entries of the $K$ eigenvectors and uses a clustering algorithm such as $K$-means to create a hard partition
Figure 4.7: Comparison of spectral analysis with respect to the sparse matrix $\Omega$ and our full affinity matrix $\Pi$. (a) Test images. (b) The average running times to solve the generalized eigenvalue problem in (4.12) by MATLAB 7.13 on a quad-core 3.3GHz desktop as a function of the number of pixels ($K = 5$). (c)-(d) and (e)-(f) The 2-3th smallest eigenvectors in (4.12) based on $\Omega$ and $\Pi$, respectively.

2) Hierarchical Segmentation: We transform the contour signals, produced by combining contour information in different eigenvectors, into a hierarchy regions using OWT and UCM, similarly in [13].
4.4. SPECTRAL SEGMENTATION

Figure 4.8: Overview of our $K$-way segmentation algorithm FNCut. (a) Test image. (b)-(e) The four smallest eigenvectors $\{\vec{v}_1, ..., \vec{v}_4\}$ in (4.12). The first eigenvector in (b) is very close to a constant vector. (f)-(h) The $K$-way segmentation results ($K = 2, 3, $ and 4).

4.4.1 $K$-way Segmentation

Let us consider the image segmentation as a labeling problem in which one label $k \in \{1, ..., K\}$ is assigned to each pixel $i$. Let $\tilde{z}_k = [z_{ik}]_{N \times 1}$ denote a partitioning vector with $z_{ik} = 1$ if $i$ belongs to the $k$-th segment and 0 otherwise. Here, the segmentation criterion of our algorithm FNCut is same to that of NCut [9]:

$$\min C(Z) = \frac{1}{K} \sum_{k=1}^{K} \frac{\tilde{z}_k^T \Pi \tilde{z}_k}{\tilde{z}_k^T D \Pi \tilde{z}_k}$$

subject to $\Pi$ in (4.11) and $ZZ^T = I$, where the partitioning matrix $Z = [\tilde{z}_1, ..., \tilde{z}_K]$.

If $\tilde{z}_k$ is relaxed to take on real values, we can minimize Eq. (4.16) by solving the generalized eigenvalue system in (4.12). Namely, the optimal solution of $C(Z)$ in (4.16) is the subspace spanned by the $K$ smallest generalized eigenvectors $V_k = [\vec{v}_1, ..., \vec{v}_K]$ in (4.12). As this subspace is discretized by $K$-means clustering, our $K$-way segmentation is obtained.
Figure 4.9: Example of multilayer segmentation by FNCut. (a) Test image. (b)-(d) The three over-segmentations by varying the parameters of the MShift algorithm. (e)-(h) Our $K$-way segmentations at each layer (a)-(d), respectively ($K = 20$).

Fig. 4.8 shows an example which illustrates our $K$-way segmentation. The four smallest eigenvectors in (4.12) are shown in Fig. 4.8(b)-(e). The 2-4 way partitions in Fig. 4.8(f)-(h) are obtained using the 2-4 different eigenvectors, respectively. Fig. 4.9 presents one example of our multilayer segmentation. The three over-segmentations in Fig. 4.9(b)-(d) contain some incorrect regions, since some parts of an airplane have a similar color distribution to the sky in Fig. 4.9(a). Nevertheless, our multilayer approach produces an impressive final segmentation in Fig. 4.9(e). Further as depicted in Fig. 4.9(e)-(h), the segmentation results at different layers are very similar to one another. This finding is attributed to the label-continuity between two nodes at different layers, which is enforced by our segmentation criterion in (4.16) based on the full affinity matrix between all pixels and all regions.
4.4. SPECTRAL SEGMENTATION

Figure 4.10: Overview of our hierarchical segmentation algorithm fPb-OWT-UCM. (a) Test image. (b) Maximal response of contour signals by fPb in (4.17). (c) Region tree in the UCM. (d)-(h) Segmentations produced by thresholding the UCM in (c) at level 0.01, 0.05, 0.1, 0.3, and 0.5, respectively.

4.4.2 Hierarchical Segmentation

The generalized eigenvectors \( \{\vec{v}_1, ..., \vec{v}_K\} \) in (4.12), corresponding to the \( K \) smallest eigenvalues \( 0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_K \) can be used for contour detection, since the eigenvectors themselves carry contour information [10]. Treating each eigenvector \( \vec{v}_k \) as an image, \( \vec{v}_k \) is convolved with Gaussian directional derivative filters at multiple orientations \( \{\theta\} \) for obtaining the oriented signals \( \{\nabla_\theta \vec{v}_k\} \). By combining the contour signals in the different eigenvectors, the boundary detector fPb based on spectral analysis of our full affinity matrix \( \Pi \) is defined as:

\[
\text{fPb}(\theta) = \sum_{k=2}^{K} \frac{1}{\sqrt{\lambda_k}} \cdot \nabla_\theta \vec{v}_k, \tag{4.17}
\]

where the weighting by the eigenvalue \( \lambda_k \) is motivated by the physical interpretation of the generalized eigenvalue problem as a mass-spring system [85].

From the contour signals obtained by fPb in (4.17), our algorithm fPb-OWT-
UCM can produce a hierarchical segmentation result as follows. Similarly in [13], we first generate the non-overlapped regions based on these contour signals by using the OWT [82], and then construct a hierarchy of regions in the UCM [83]. This procedure can be seen as generic machinery for going from the contours to a hierarchical region tree. Fig. 4.10 shows an example which illustrates our hierarchical segmentation. The UCM result in Fig. 4.10(c) is expressed as a region tree, which preserves the quality of contours, extracted by fPb in Fig. 4.10(b). Therefore, the segmentation result at any scale can be easily retrieved by thresholding the UCM at that scale, such as in Fig. 4.10(d)-(h).

4.5 Experimental Results

4.5.1 Parameter Setting

To construct our multilayer graph, we initially generate the regions by the unsupervised image segmentation methods such as graph-based image segmentation (GBIS) [12] and MShift [4], which attempt to partition image pixels into components such that the resulting segmentation is neither too coarse nor too fine. Fig. 4.11 shows our spectral segmentation results using two different over-segmentation methods: GBIS and MShift. The over-segmentations by MShift in Fig. 4.11(g)-(i) better detect thin elongated parts such as the ski & ski pole, and less contain superfluous regions around the object boundaries than those by GBIS in Fig. 4.11(b)-(d). Therefore, we used MShift as the over-segmentation method in our algorithms.

The MShift method needs two bandwidth parameters \((h_s, h_r)\) for the spatial and range domains, respectively. In the experiments, three region-based layers with the different sets of two MShift parameters were obtained. Now we consider the selection
4.5. EXPERIMENTAL RESULTS

Figure 4.11: Comparison of over-segmentations by GBIS [12] and MShift [4] for spectral segmentation. (a) Test images. (b)-(d) and (g)-(i) The three different over-segmentations by GBIS and MShift, respectively. (e) and (j) $K$-way segmentations using the regions in (b)-(d) and (g)-(i), respectively (Top: $K = 10$, Bottom: $K = 20$). (f) and (k) Hierarchical segmentations using the regions in (b)-(d) and (g)-(i), respectively.

Figure 4.12 shows our segmentation results by changing this selection. If multiple region-based layers of quite different scales in Fig. 4.12(c)-(e) are used, our segmentation results in Fig. 4.12(i)-(l) mainly depend on the quality of the largest-scale region-based layer in Fig. 4.12(e). Namely, some objects whose
Figure 4.12: Examples of our segmentation results using different region-based layers \( Y \) produced by MShift in the multilayer graph \( G \). (a) Test images. (b) Ground-truth boundaries. (c)-(e) and (f)-(h) The three over-segmentations in accordance with the different sets of the MShift parameters \( (h_s, h_r) = \{(5, 7), (10, 15), (15, 25)\} \) and \( (h_s, h_r) = \{(5, 7), (7, 5), (7, 7)\} \), respectively. (i)-(k) and (m)-(o) 20/30/40-way segmentations using the region-based layers (c)-(e) and (f)-(h), respectively. (l) and (p) Hierarchical segmentations using the region-based layers (c)-(e) and (f)-(h), respectively.

Boundaries exist in Fig. 4.12(e) can be well-detected such as “dock” (top) and “car” (bottom) in Fig. 4.12(i)-(l). By contrast, if the boundaries of the object parts are missing in Fig. 4.12(e), finding their exact boundaries is very difficult such as “boat” (top) and “tree” (bottom) in Fig. 4.12(i)-(l). To make up for this large-scale limitation, similar small-scale region-based layers are used, such as in Fig. 4.12(f)-(h). Since they give multiple object boundary candidates, the object details appear on the results in Fig. 4.12(m)-(p) by increasing the number of segments \( K \) for \( K- \)
4.5. EXPERIMENTAL RESULTS

Figure 4.13: Examples of our affinity images with respect to the variation of parameters $\tau$ (top row, $c = 10^{-5}$) and $c$ (bottom row, $\tau = 10^{-3}$) in (4.6) and (4.11), respectively.

way segmentation or by decreasing a threshold for hierarchical segmentation. In our experiments, we set $(h_s, h_r) = \{(5, 7), (7, 5), (7, 7)\}$ empirically.

Note that our affinity model has two parameters: the interlayer edge weight $\tau$ in (4.6) and the balanced weight $c$ in (4.11). Fig. 4.13 shows the variation in affinity images with respect to these two parameters. With a larger $\tau$, the region consistency is more emphasized, and the affinities are more discretized on the region boundaries as shown in the top row of Fig. 4.13. With a smaller $\tau$, the affinities are more over-smoothed around the user-given point. With a larger $c$, the fitting constraint is more emphasized, and the only affinities within a smaller distance are presented as shown in the bottom row of Fig. 4.13. With a smaller $c$, long-range connections around the user-given point are more emphasized. The parameters $\tau, c \in \{10^{-1}, 10^{-3}, 10^{-5}\}$ were experimentally chosen and set $\tau = 10^{-3}$ and $c = 10^{-5}$ for all test images.
4.5.2 Measurements

For quantitative comparison, the following four measures are used:

1) **F-measure**: The traditional F-measure (F) is the harmonic mean of precision and recall as follows.

\[
F = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}},
\]

(4.18)

where Precision is the number of correct results divided by the number of all returned results and Recall is the number of correct results divided by the number of results that should have been returned. The F score reaches its best value at 1 and worst score at 0.

2) **Segmentation Covering**: The overlap between two regions \( R \) and \( R' \) is defined as:

\[
\mathcal{O}(R, R') = \frac{|R \cap R'|}{|R \cup R'|}.
\]

(4.19)

This measure has been used for the evaluation of the pixel-wise classification task in recognition [86]. We define the covering (Covering) of a segmentation \( S \) by a ground-truth segmentation \( T \) as:

\[
\text{Covering}(T \rightarrow S) = \frac{1}{|\mathcal{X}|} \sum_{R \in S} |R| \cdot \max_{R' \in T} \mathcal{O}(R, R'),
\]

(4.20)

where \( \mathcal{X} \) is a set of pixels in the image. Similarly, the covering of a machine segmentation \( S \) by a family of ground-truth segmentations \( \{T_i\} \) is defined by first covering \( S \) separately with each human segmentation \( T_i \), and then averaging over the different humans.

3) **Probabilistic Rand Index**: The rand index (RI) [87] between test and ground-truth segmentation is calculated by the sum of the number of pairs of pixels that
have the same label and those that have different labels in both segmentations, divided by the total number of pairs of pixels. The probabilistic rand index (PRI), a variant of RI, has been proposed for dealing with the case of multiple ground-truth segmentations [88]. The PRI amounts to averaging the RI among different ground-truth segmentations.

4) Variation of Information: The Variation of Information (VI) score was introduced for the purpose of clustering comparison [89]. The VI measures the distance between a segmentation $S$ and a ground-truth segmentation $T$ in terms of their average conditional entropy given by:

$$
\text{VI}(S; T) = H(S) + H(T) - 2I(S; T),
$$

(4.21)

where $H(\cdot)$ and $I(\cdot)$ represent respectively the entropies and mutual information between two segmentations $S$ and $T$. In the presence of several ground-truth segmentations $\{T_i\}$, we average the VI among different ground-truth segmentations.

The segmentation is viewed better if VI is smaller or the other three are larger.

4.5.3 Results

Our spectral segmentation algorithms: FNCut for $K$-way segmentation and fPb-OWT-UCM for hierarchical segmentation, are quantitatively evaluated on two Berkeley image datasets: BSDS300 $^1$ and BSDS500 $^2$. The BSDS300 (or BSDS500) dataset consists of 200 (or 300) training images and 100 (or 200) test images. The above-mentioned measures are performed on the test images. More visual segmentation results are then given on the MSRC object recognition database $^3$.

$^1$http://www.cs.berkeley.edu/projects/vision/bsds
$^2$http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources.html
<table>
<thead>
<tr>
<th></th>
<th>BSDS300</th>
<th>BSDS500</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ODS</td>
<td>OIS</td>
</tr>
<tr>
<td>Human</td>
<td>0.79</td>
<td>0.79</td>
</tr>
<tr>
<td>[4] MShift</td>
<td>0.63</td>
<td>0.66</td>
</tr>
<tr>
<td>[8] MNCut</td>
<td>0.62</td>
<td>0.66</td>
</tr>
<tr>
<td>Canny-OWT-UCM</td>
<td>0.58</td>
<td>0.63</td>
</tr>
<tr>
<td>[12] GBIS</td>
<td>0.58</td>
<td>0.62</td>
</tr>
<tr>
<td>[90] SWA</td>
<td>0.56</td>
<td>0.59</td>
</tr>
<tr>
<td>[81] mPb-OWT-UCM</td>
<td><strong>0.68</strong></td>
<td><strong>0.72</strong></td>
</tr>
<tr>
<td>[13] gPb-OWT-UCM</td>
<td><strong>0.71</strong></td>
<td><strong>0.74</strong></td>
</tr>
<tr>
<td>FNCut</td>
<td>0.64</td>
<td>0.68</td>
</tr>
<tr>
<td>fPb-OWT-UCM</td>
<td>0.67</td>
<td>0.70</td>
</tr>
<tr>
<td>cPb-OWT-UCM</td>
<td><strong>0.70</strong></td>
<td><strong>0.72</strong></td>
</tr>
</tbody>
</table>

Table 4.1: Boundary benchmarks on the BSDS300 and BSDS500 datasets. Results for ten different segmentation algorithms are given. The F scores when choosing an optimal scale for the entire dataset (ODS) or per image (OIS), as well as the average precision (AP). The best three results are highlighted in colors: red, green, and blue in descending order.

In the experiments, a single segmentation that involves a choice of scale as output is produced. The optimal scale is chosen according to the following two conditions: optimal dataset scale (ODS) and optimal image scale (OIS). The former is to use a fixed parameter for all images in the dataset. The latter is to select the optimal parameter by an oracle on a per-image basis. The scale parameter is the number of segments $K \in \{2, 3, ..., 40\}$ for $K$-way segmentation or a threshold in range of $[0,1]$ for hierarchical segmentation.
4.5. EXPERIMENTAL RESULTS

Figure 4.14: Boundary benchmarks on the BSDS300 (Left) and BSDS500 (Right) datasets. In the box, several segmentation approaches are ranked according to their maximum F score in (4.18) with respect to human ground-truth boundaries. Iso-F curves are shown in green. Average agreement between human subjects is indicated by the green dot.

Boundary Quality

In our experiments, we report three different boundary quantities for each algorithm: the best F on the dataset for a fixed scale (ODS), the aggregate F on the dataset for the best scale in each image (OIS), and the average precision (AP) on the full recall range. Table 4.1 shows these quantities for the BSDS300 and BSDS500 datasets. To provide a basis of comparison, the seven existing image segmentation algorithms are reported: one K-way spectral method (MNCut [8]), three region merging methods (MShift [4], GBIS [12], SWA [90]), and three contour-based methods (Canny-OWT-UCM, mPb-OWT-UCM [83], gPb-OWT-UCM [13]) which make the UCM from the
CHAPTER 4. FULL AFFINITY MODEL FOR SPECTRAL SEGMENTATION

Figure 4.15: Example segmentations by our algorithms: FNCut, fPb-OWT-UCM, and cPb-OWT-UCM. (a) Test images. (b), (d) and (f) Segmentation results by FNCut, fPb-OWT-UCM, and cPb-OWT-UCM at the optimal dataset scale (ODS) in terms of F in (4.18), respectively. (c) and (e) UCMs produced by fPb-OWT-UCM and cPb-OWT-UCM, respectively. (g) Human boundaries in multiple ground-truth segmentations.

contour signals produced by Canny, multiscale boundary detector (mPb) [91], and gPb [10], respectively. Fig. 4.14 displays the full precision-recall curves on two BSDS datasets.
4.5. EXPERIMENTAL RESULTS

Figure 4.16: Statistics of our $K$-way segmentation method FNCut on Covering, PRI, and VI over (a) BSDS300 dataset and (b) BSDS500 dataset with $K = \{5, 10, \ldots, 40\}$, compared with the general spectral methods NCut and MNCut.

In Table 4.1, our algorithms FNCut and fPb-OWT-UCM better detect the object boundaries than other non-learning methods: MNCut, MShift, GBIS, SWA, and Canny-OWT-UCM. Particularly, fPb-OWT-UCM has better performance than FNCut, since the $K$-way segmentation algorithms such as FNCut often make large uniform regions in which the eigenvectors vary smoothly broken up. However, the segmentation results by fPb-OWT-UCM have poorer boundary quality than those by
Figure 4.17: Visual comparison of our algorithm FNCut with the general K-way spectral methods NCut [9] and MNCut [8] on the MSRC database. (a) Test images. (b)-(d) Segmentation results by NCut, MNCut, and FNCut, respectively.

mPb-OWT-UCM and gPb-OWT-UCM which are learned using the training images and their corresponding ground-truth segmentations. To improve our performance,
we additionally propose a learning-based method cPb-OWT-UCM whose boundary
detector cPb is designed as the combination of two detectors mPb and fPb using the
balancing weights derived from the training set, similarly to gPb which is written as
a weighted sum of mPb and its spectral component. Although cPb-OWT-UCM has
a little poorer performance than gPb-OWT-UCM, it has better performance than
mPb-OWT-UCM by adding our globalization machinery. Fig. 4.15 gives a visual
comparison of our algorithms: FNCut, fPb-OWT-UCM, and cPb-OWT-UCM.

Region Quality

Table 4.2 and Table 4.3 present region benchmarks on the BSDS300 and BSDS500
datasets using three measures: Covering, PRI, and VI, respectively. For each mea-
sure, two scores, which correspond to selecting regions from the segmentation result
at a universal fixed scale (ODS) or a fixed scale per image (OIS), are reported.

*K-way Segmentation:* In general, the spectral methods such as NCut [9] and
MNCut [8] follow a K-way segmentation scheme. In Table 4.2 and Table 4.3, the pro-
posed K-way segmentation algorithm FNCut has better region quality than MNCut.
In detail, Fig. 4.16 shows a comparison of FNCut with NCut and MNCut on the
BSDS datasets by varying a fixed segment number K from 5 to 40 (in steps of
five). These experiments prove that FNCut outperforms MNCut which indirectly
considers long-range connections in a multi-scale framework, as well as NCut in all
cases. Fig. 4.17 shows a more visual comparison of segmentation results on the
MSRC database. Compared with NCut and MNCut, FNCut produces perceptually
high-quality segmentations which detect large textured regions as well as elongated
object parts by considering the well-defined full pairwise affinities.
### Table 4.2: Region benchmarks on the BSDS300 dataset.

sPb is the boundary detector based on spectral analysis of sparse affinity matrix such as the weight matrix $\Omega$.

Hierarchical Segmentation: Unlike the state-of-the-art learning-based methods mPb-OWT-UCM [83] and gPb-OWT-UCM [13], our algorithm fPb-OWT-UCM works without training. In spite of that, the regions produced by fPb-OWT-UCM have better quality than those by mPb-OWT-UCM and a little poorer quality than those by gPb-OWT-UCM in Table 4.2 and Table 4.3. By additionally using local edge information in mPb, another proposed hierarchical segmentation algorithm...
4.5. EXPERIMENTAL RESULTS

<table>
<thead>
<tr>
<th></th>
<th>BSDS500</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Covering</td>
</tr>
<tr>
<td>Human</td>
<td>0.72 0.72</td>
</tr>
<tr>
<td>[4] MShift</td>
<td>0.54 0.58</td>
</tr>
<tr>
<td>[12] GHS</td>
<td>0.52 0.57</td>
</tr>
<tr>
<td>Canny-OWT-UCM</td>
<td>0.49 0.55</td>
</tr>
<tr>
<td>[8] MNCut</td>
<td>0.45 0.53</td>
</tr>
<tr>
<td>[83] mPb-OWT-UCM</td>
<td>0.566 0.629</td>
</tr>
<tr>
<td>[13] gPb-OWT-UCM</td>
<td><strong>0.588</strong> 0.647</td>
</tr>
<tr>
<td>FNCut</td>
<td>0.529 0.602</td>
</tr>
<tr>
<td>fPb-OWT-UCM</td>
<td><strong>0.582</strong> 0.633</td>
</tr>
<tr>
<td>cPb-OWT-UCM</td>
<td><strong>0.594</strong> 0.653</td>
</tr>
<tr>
<td>NCut with Ω</td>
<td>0.518 0.598</td>
</tr>
<tr>
<td>FNCut by ψ_{mi}</td>
<td>0.523 0.605</td>
</tr>
<tr>
<td>sPb-OWT-UCM with Ω</td>
<td>0.576 0.627</td>
</tr>
<tr>
<td>fPb-OWT-UCM by ψ_{mi}</td>
<td>0.577 0.629</td>
</tr>
</tbody>
</table>

Table 4.3: Region benchmarks on the BSDS500 dataset.

cPb-OWT-UCM achieves the best performance in region benchmarks. Fig. 4.18 compares our algorithms: fPb-OWT-UCM and cPb-OWT-UCM, with gPb-OWT-UCM on the BSDS500 dataset. The region hierarchy produced by gPb-OWT-UCM conceptually well reflects the likelihood that each contour is a true boundary when the contents of a test image contains in training images, but the regions usually have oversmoothed boundaries. On the other hand, fPb-OWT-UCM provides the regions which preserve object details by directly incorporating the full-range connections, but it may fail to separate the object parts with similar color distributions. By combining advantages of these two algorithms, cPb-OWT-UCM well construct a hierarchy of object parts with detailed boundaries.
Figure 4.18: Visual comparison of our algorithms fPb-OWT-UCM and cPb-OWT-UCM with the state-of-the-art hierarchical segmentation method gPb-OWT-UCM [13] on the BSDS500 dataset. (a) Test images. (b), (d), and (f) UCMs produced by gPb-OWT-UCM, fPb-OWT-UCM, and cPb-OWT-UCM, respectively. (c), (e), and (g) Segmentation results by gPb-OWT-UCM, fPb-OWT-UCM, and cPb-OWT-UCM at the optimal dataset scale (ODS) in terms of the Covering measure in (4.20), respectively. (h) Human boundaries.

Affinity Quality

We first consider that the sparse weight matrix $\Omega$ in our multilayer graph, instead of the full matrix $\Pi$, is directly used as the affinity matrix in the spectral segmentation
4.6 Conclusion

This chapter deals with two types of spectral segmentation algorithms: $K$-way segmentation and hierarchical segmentation. Our work is novel in that it sheds understanding on the full pairwise affinities gained in integrating local grouping cues by taking the SSL strategy. Since these well-defined affinities are directly embedded in our spectral segmentation algorithms FNCut and fPb-OWT-UCM without any approximation, our algorithms produce high-quality segmentation results with object details in natural images. Further, despite the use of full affinities, their spectral analysis is computationally very efficient. Finally, in case of hierarchical segmentation, by simply combining local and spectral contour signals, the advanced algorithm cPb-OWT-UCM can achieve the best performance in region benchmarks.

To produce an ideal segmentation result, an optimal scale should be chosen. Therefore our future work will deal with automatic estimation of the optimal scale. To improve the performance of our algorithms, a more effective graph or SSL function also needs to be designed for affinity estimation.
Chapter 5

Conclusion

In this dissertation, we proposed two interactive segmentation algorithms and one spectral segmentation algorithm using semi-supervised learning, respectively.

In Chapter 2 and 3, we present two novel likelihood models in the generative image segmentation framework.

In Chapter 2, we provide a new interpretation of Random Walks with Restart (RWR) for likelihood estimation. Although the previous Random Walker image segmentation algorithm (RW) [3] is also based on the Random Walks concept, our work is conceptually different from RW, and produces significant improvement in performance as shown in the experiments. The key differences between RW and our work are "First arrival probability vs. Average probability". In [3], the score between a pixel and each label is defined by the first arrival probability that a particle starting at a pixel reaches to a seed. On the other hand, in our work it is defined as the average probability that a particle starting at one of the seeds stays at a pixel. Our approach has several advantages for image segmentation. First, owing to the generative segmentation model based on RWR, it can obtain segmentations with
just single label. Second, it is less dependent on the number of seeds, since the likelihood is computed by the average of the relevance scores of all the seeds and the higher-order constraint can be additionally imposed. Third, we can integrate region and boundary information globally in the RWR-based framework. Finally, it gives qualitatively and quantitatively better segmentations on highly-textured natural images. Generally, large neighborhood system is needed for obtaining object details, because it captures image structure well. However, since it gives high computations, many efficient techniques, like multi-scale approach, have been proposed. Our RWR-based algorithm is an alternative solution, since it considers all possible paths between two nodes in small neighborhood system.

In Chapter 3, we design another likelihood model to strengthen the higher-order constraint in the segmentation algorithm, since the higher-order effect of RWR-based algorithm is actually limited for segmentation of natural images. In this work, we propose the new higher-order cues defined in the regions. These higher-order cues are efficiently used to estimate the pixel likelihoods in the quadratic formulation, and recursively estimated from the resulting likelihoods of pixels included in each region. Since in this manner all regional cues are propagated into the whole image by using the full connections between regions, it gives the same effect as the nonparametric priori model. The performance of our algorithm was tested on the challenging data sets. The segmentation results demonstrate that our higher-order cues are helpful in producing high-quality segmentations with detailed boundaries and reducing the sensitivity of the seed quantity and placement.

In Chapter 4, we present two types of spectral segmentation algorithms: $K$-way segmentation and hierarchical segmentation. Our work is novel in that it sheds un-
derstanding on the full pairwise affinities gained in integrating local grouping cues by taking the semi-supervised learning strategy. Since these well-defined affinities are directly embedded in our spectral segmentation algorithms FNCut and fPb-OWT-UCM without any approximation, our algorithms produce high-quality segmentation results with object details in natural images. Further, despite the use of full affinities, their spectral analysis is computationally very efficient. Finally, in case of hierarchical segmentation, by simply combining local and spectral contour signals, the advanced algorithm cPb-OWT-UCM can achieve the best performance in region benchmarks.

There are two common parts of these three proposed algorithms as follows.

First, they use the same region and boundary cues for image segmentation. By using the colors of the pixels and the over-segmented regions as the region cues, we add the following soft constraints into the segmentation framework: the smoothness constraint that two pixels or regions should have similar likelihoods or affinities if their colors are similar and the region consistency constraint that the pixels in a region should have similar likelihoods or affinities. Also, by using the boundaries of the over-segmented regions as the boundary cues, the discontinuity constraint that the boundary between two adjacent regions is emphasized when their local properties are quite different is imposed.

Second, region and boundary cues are integrated using the graph-based semi-supervised learning techniques: the RWR-based techniques which conceptually define the particle movements in the graph (in Chapter 2) and the cost minimization techniques which meaningfully design the loss function and regularizer (in Chapter 3 and Chapter 4). Although we consider only the relationship between the neighboring pixels
for graph construction, the integration of local region and boundary cues in the graph indirectly gives the higher-order or long-range effects. Therefore, using semi-supervised learning can help improve the segmentation performance.

To produce an ideal segmentation result, the optimal parameters should be chosen. Since they were empirically chosen in our algorithms, they are not optimal for every image. Therefore, our future work will deal with automatic estimation of the optimal parameters. Moreover we used only color values as the region cues. If we use other cues such as texture, better segmentation results will be obtained. Also, in order to improve the performance of our algorithms, a more effective graph or a different semi-supervised learning function will be tested in the segmentation frameworks.
Bibliography


BIBLIOGRAPHY


