Efficient optimization for labeling problems with prior information: applications to natural and medical images

Junjie Bai
University of Iowa

Follow this and additional works at: https://ir.uiowa.edu/etd
Part of the Electrical and Computer Engineering Commons

EFFICIENT OPTIMIZATION FOR LABELING PROBLEMS WITH PRIOR INFORMATION: APPLICATIONS TO NATURAL AND MEDICAL IMAGES

by

Junjie Bai

A thesis submitted in partial fulfillment of the
requirements for the Doctor of Philosophy
degree in Electrical and Computer Engineering
in the Graduate College of
The University of Iowa

May 2016

Thesis Supervisor: Associate Professor Xiaodong Wu

Graduate College
The University of Iowa
Iowa City, Iowa

CERTIFICATE OF APPROVAL

PH.D. THESIS

This is to certify that the Ph.D. thesis of

Junjie Bai

has been approved by the Examining Committee for the thesis requirement for the Doctor of Philosophy degree in Electrical and Computer Engineering at the May 2016 graduation.

Thesis Committee: ______________________
Xiaodong Wu, Thesis Supervisor

____
Mona K. Garvin

____
Yusung Kim

____
Joseph M. Reinhardt

____
Kasturi R. Varadarajan

ACKNOWLEDGEMENTS

I am deeply grateful for every help and support I got from everyone. Without your help and support, I would never be able to do what I have accomplished during my Ph.D. study.

I would like to thank Dr. Xiaodong Wu, my advisor. He is wise, patient, and greatly helpful in guiding me through my whole Ph.D. life. The meeting and discussion with you is always filled with inspiring ideas and practical suggestions. It is very lucky of me to have you as my Ph.D. advisor. I am honored and proud to be your Ph.D. student.

I would also like to thank my committee members: Prof. Garvin, Prof. Kim, Prof. Reinhardt, and Prof. Varadarajan. Thank you for being on my committee. I appreciate all your constructive and valuable feedback to my dissertation. It is great joy to collaborate with you during my Ph.D. study.

I also thank my wife and my family. No matter what happens, you are always my home. You know how much you mean to me.

Last but not least, I would love to thank my Iowa friends. I will not name all of them because I am in fear of accidentally leaving someone out. But thank you for sharing part of your life with me. The help, discussion, and hangout are all very valuable to me. I would like to thank everyone of you for being friends with me and having a great six years together.

ABSTRACT

Labeling problem, due to its versatile modeling ability, is widely used in various image analysis tasks. In practice, certain prior information is often available to be embedded in the model to increase accuracy and robustness. However, it is not always straightforward to formulate the problem so that the prior information is correctly incorporated. It is even more challenging that the proposed model admits efficient algorithms to find globally optimal solution.

In this dissertation, a series of natural and medical image segmentation tasks are modeled as labeling problems. Each proposed model incorporates different useful prior information. These prior information includes ordering constraints between certain labels, soft user input enforcement, multi-scale context between over-segmented regions and original voxel, multi-modality context prior, location context between multiple modalities, star-shape prior, and gradient vector flow shape prior.

With judicious exploitation of each problem's intricate structure, efficient and exact algorithms are designed for all proposed models. The efficient computation allow the proposed models to be applied on large natural and medical image datasets using small memory footprint and reasonable running time. The global optimality guarantee makes the methods robust to local noise and easy to debug.

The proposed models and algorithms are validated on multiple experiments, using both natural and medical images. Promising and competitive results are shown compared to state-of-art.

PUBLIC ABSTRACT

Image segmentation, which extracts the target object from background, is an important task for both natural and medical images. It is natural to model image segmentation as a labeling problem in which every voxel gets a label indicating object or background. Various prior information for different applications are usually available in practice problems. In fact, for many tasks, the incorporation of prior information may be the key to achieve successful segmentations.

However, it is not always straightforward to enforce certain prior constraints in the labeling formulation correctly. What's even more challenging, the designed model may not have efficient algorithm to find the globally optimal solution.

In this dissertation, a series of labeling problems incorporating various useful prior information are proposed. Moreover, the globally optimal solution for all the proposed models can be computed using efficient algorithms. The proposed models and algorithms are validated on multiple applications of natural and medical image segmentation tasks. The experiment shows promising results competitive to the state-of-art.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Labeling problems are widely used to formulate natural and medical image analysis problems. The advantages of using this formulation are mainly three-folds: 1) a global energy formulation is more robust to local noise in images; 2) flexible modeling ability by introducing various terms including unary, pairwise or high-order terms; 3) efficient exact or approximate solutions for a large set of useful models.

My Ph.D. thesis identifies a series of labeling problems with different prior information, propose efficient and *exact* solution for them, and validate the model and algorithm on natural or medical image applications. All the proposed works have a special emphasis of finding *globally optimal* solution efficiently. The guarantee of a globally optimal solution greatly reduces the burden of application development cycle. With the globally optimality guarantee, if the application is not working satisfactorily, then we can safely concentrate our work on designing better image feature computation methods. Without this guarantee, we will be constantly wondering whether the poor result comes from the poor image feature computation or the poor optimization ability of the optimization method we use. Being efficient, the proposed works will be applicable to very large problems, which is required by most of the 3D and 4D image applications.

The introduction is organized as follows. First, an introduction about the general labeling problem formulation and optimization techniques is given in Sec.1.1 and 1.2. In Sec.1.3 and 1.4 two popular techniques for solving labeling problems:

graph-cut [52] and graph-search [15] are reviewed. These two methods motivate the proposed works and are building blocks of a large portion of the proposed methods. A summary of the dissertation is then given in Sec.1.5. Seven labeling problems with different prior information are introduced one by one. Finally, all specific aims are summed up concisely from the point view of labeling problems with different priors in Sec.1.6.

1.1 Labeling Problem

Many problems in computer vision and medical image analysis have been modeled as a labeling problem, including stereo matching [15], texture restoration [41, 40], binary interactive image segmentation [14, 12], and multi-label image segmentation [24], etc. A labeling problem formulation aims to label every pixel p with an integer valued label $x_p \in \mathcal{L}$, where $\mathcal{L} = \{1, 2, \ldots, |\mathcal{L}| - 1\}$ is the set of all available labels. The labeling is computed by *minimizing* an energy function that reflects both image information and some prior informations we expect from the labeling.

A typical labeling problem defines an energy function in the form of Eq.(1.1).

$$E(\mathbf{x}) = \sum_{p} D_p(x_p) + \sum_{(p,q)\in\mathcal{N}_2} V_{pq}(x_p, x_q) + \sum_{\{i_k\}\in\mathcal{N}_K} V_{\{i_k\}}(x_{i_1}, \ldots, x_{i_K}), K \geq 3 \quad (1.1)$$

The first term $D_p(x_p)$ is a unary term (also called 'data term') defining the likelihood of assigning label x_p to pixel p. The unary data term reflects image information. The second term $V_{pq}(x_p, x_q)$ is a pairwise term that reflects the prior penalty of assigning label x_p to pixel p and label x_q for pixel q for pixel pair (p, q) in a neighborhood setting $\mathcal{N}_2$. Similarly, the third term $V_{\{i_k\}}$ is a high-order term $(K \geq 3)$ that reflects the prior penalty of assigning labels $x_{i_1}, x_{i_2}, \ldots, x_{i_K}$ respectively to pixels $i_1, i_2, \ldots, i_K$ for pixel set $\{i_k\}$ in a high-order neighborhood setting (also called 'clique'), which defines a neighborhood involving at least three variables.

Let's look at a concrete example. Suppose we are trying to segment a single-pixel-wide curve in the image. Then the labeling set is binary $\mathcal{L} = \{0, 1\}$, where 0 denotes the background and 1 denotes the object, i.e., the curve. The unary term $D_p(x_p)$ reflects the reverse likelihood of pixel p being labeled as x_p. For example, if we know the curve is black and the background is white, then the higher the intensity is, the less likely pixel p is going to be object. The pairwise term V_{pq} will in general encourages the two adjacent neighboring pixels (according to 4-neighborhood or 8-neighborhood setting) to have the same label, which prevents a cluttered labeling where adjacent pixels frequently having different labels. The high-order term can be defined on a neighborhood $\mathcal{N}_3$ involving any three consecutive pixels on a straight line. We can use this term to encourage the segmentation to be single pixel wide. For example, $V_{i_1,i_2,i_3}(x_{i_1} = 0, x_{i_2} = 1, x_{i_3} = 0)$ will be low since this represents a single pixel wide line going through pixel i_2. However, $V_{i_1,i_2,i_3}(x_{i_1} = 0, x_{i_2} = 1, x_{i_3} = 1)$ will be high since this usually signals a line more than one pixel wide.

1.2 Labeling Problem Categories

Labeling problem in Eq.(1.1) in general is NP-hard. For certain special cases, efficient algorithms exist for globally optimal solution. An import subset of these special cases uses only the unary term and pairwise term in Eq.(1.1).

$$E(\mathbf{x}) = \sum_{p} D_p(x_p) + \sum_{(p,q)\in\mathcal{N}_2} V_{pq}(x_p, x_q) \tag{1.2}$$

Assume all variables in Eq.(1.2) are binary, then given a mild condition (being submodular) on the pairwise term, Eq.(1.2) can be efficiently and *exactly* minimized by graph-cut [50]. The problem is reduced to a single minimum s-t cut computation in an appropriately constructed graph, given that $V_{pq}(x_p, x_q)$ is submodular for all pairs of $(p, q) \in \mathcal{N}_2$. A pairwise function $V(x, y)$ is submodular if and only if $V(1, 1) + V(0, 0) \leq V(1, 0) + V(0, 1)$. Luckily many useful pairwise priors are submodular. For example, in the binary image segmentation problem, we often set $V(1, 1) = V(0, 0) = 0$ to encourage smooth labeling. The rationale is that if we set $V(1, 0)$ and $V(0, 1)$ to have lower penalty than $V(0, 0)$ and $V(1, 1)$, we are effectively encouraging a very cluttered segmentation similar to a checkerboard pattern, which is usually not what we want. The graph-cut technique will be discussed in more details in Sec.1.3.

For the non-submodular labeling problems that cannot be exactly minimized by graph-cut, several techniques, such as quadratic pseudo boolean optimization [49], belief propagation [29], and tree-reweighted message passing [48], can achieve approximate solutions. Quadratic pseudo boolean optimization (QPBO) is an iterative combinatorial optimization method which computes a series of minimum s-t cut [49]. It outputs partial optimal labeling, in which some variables may not be labeled at all. But all the labeled variables are guaranteed to be labeled exactly as some optimal solution. Max-product loopy belief propagation [29] and tree-reweighted message passing [48] are two methods based on solving linear program relaxation, which involves the relaxation of the variables to be continuous, solving the relaxed problem, and then rounding the solution to be an integral one. These relaxation methods can provide a lower-bound to the optimal integral solution. However, the rounding step cannot provide optimality guarantee on the rounded integral solution.

Given the pairwise term $V(x, y)$ is a convex function of $|x - y|$, the multi-labeling version of Eq.(1.2) can be efficiently and *exactly* minimized by graph-search [106]. The problem is also reduced to a single minimum s-t cut computation. Graph-search uses 'columns' to transform the multi-labeling problem equivalently to a binary labeling problem in which every 'column' will corresponds to all possible values of a variable. This 'column' structures enables elegant incorporation of various priors like minimum/maximum surface distance constraints when applied to the multi-surface segmentation problem [52], which proves to be very beneficial for medical image segmentation where such prior informations are usually vital to obtain good segmentation [87, 33, 34, 51, 80]. Graph-search technique will be discussed in more details in Sec.1.4.

For general pairwise term $V(x, y)$, the multi-labeling energy in Eq.(1.2) can be *approximately* solved by α-expansion, and α, β-swap techniques [15]. Both techniques reduce the problem to a series of binary problems and optimizes the energy iteratively. In each binary problem, the algorithm computes a minimum s-t cut to determine an optimal binary move that guarantees not to increase the energy. The iteration continues until no binary moves can decrease the energy any more. For example, in each α-expansion iteration, each variable has to make a binary decision: either retain its original labeling, or change to a predefined label α. The iteration continues until no label $\alpha \in \mathcal{L}$ can decrease the energy by any binary move (called α-expansion) any more.

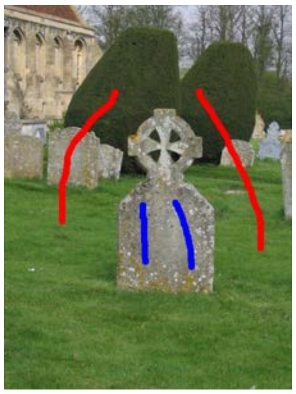
The high-order terms in Eq.(1.1) are less frequently used due to the fact that useful high-order terms often lead to NP-hard problems. But there are applications that require such high-order priors, such as texture restoration [41, 40] and vessel segmentation [45]. For the binary variables, Ishikawa et al. [40] proposes a method to reduce high-order term to possibly exponential number of multiple pairwise terms. The reduced pairwise terms are usually non-submodular. This requires techniques such as QPBO to optimize the resulting pairwise term.

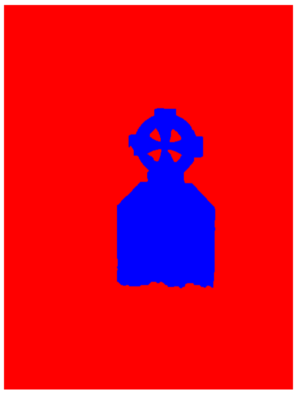
For the high-order terms with multiple possible variable values, the multi-labeling problem is first transformed to an equivalent binary high-order labeling problem using the "column" technique in graph-search [52, 106]. The resulting binary high-order labeling problem is then reduced to a binary labeling problem with only pairwise terms. Finally techniques optimizing binary pairwise labeling problems, such as QPBO, are used to get the approximate solution.

1.3 Graph-Cut

Graph-cut is widely used for binary interactive segmentation[12, 14, 72]. The segmentation problem is reduced to a binary labeling problem with a unary term and submodular pairwise term as in Eq.(1.2). Each pixel in the image corresponds to a

(a) input image


(b) seed pixels


(c) graph-cut result

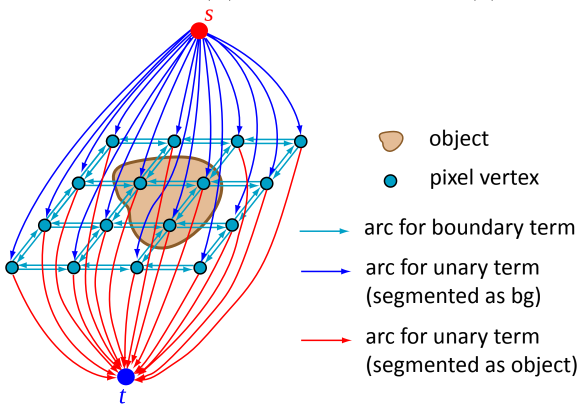

s
object
pixel vertex
arc for boundary term
arc for unary term (segmented as bg)
arc for unary term (segmented as object)
t
(d) graph construction

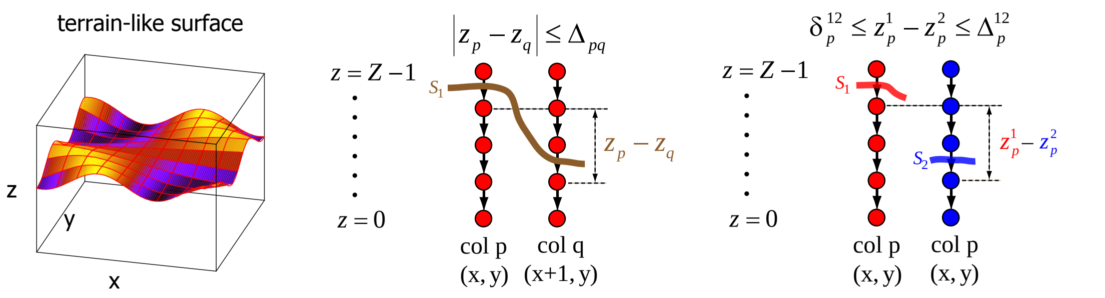
Figure 1.1: Graph-cut interactive binary image segmentation example. (a)(b)(c) shows a typical graph-cut based interactive segmentation process. (d) shows the graph construction of graph-cut.

variable. If the variable is labeled as 0, then the pixel is segmented as background. If the variable is labeled as 1, then the pixel is segmented as object.

The user usually needs to specify some object seeds and background seeds, as shown in Fig.1.1b. The seed pixels are used to build two models for object and background respectively. The model is usually based on color, but more complex models incorporating texture and other information is also possible. The two models are used to compute the likelihood of each pixel being object or background. These likelihood corresponds to the unary term $D_p(x_p)$ in Eq.(1.2).

The pairwise term in Eq.(1.2) is usually set according to gradient information from image. $V_{pq}(0,0)$ and $V_{pq}(1,1)$ are both set to be zero. While $V_{pq}(0,1)$ and $V_{pq}(1,0)$ are both set to be reversely proportional to the gradient between pixel p and q, i.e., $V_{pq}(0,1) = V_{pq}(1,0) \propto \exp(-|\nabla g(I_p, I_q)|/\sigma^2)$ where $\nabla g(I_p, I_q)$ is the gradient value between intensities I_p and I_q. σ is a tuning parameter controlling how concentrated or spread the pairwise terms are throughout the image.

A graph is then constructed as shown in Fig.1.1d. Every pixel corresponds to a vertex in the graph. Two additional vertex: source vertex s and sink vertex t are also added. An arc exists from s to every pixel vertex carrying a certain weight (dark blue arcs). And arc also exists from every pixel vertex to t carrying another weight (red arcs). The weights on these arcs encode the unary terms in Eq.(1.2). For every pair of neighboring pixels p, q according to a 4-neighborhood setting, two arcs connecting the two pixel vertices are also added to encode the pairwise term (light blue arcs).



(a) terrain-like surface (b) smoothness constraint (c) surface distance constraint

Figure 1.2: Graph-search finds multiple interacting terrain-like surfaces with various geometric constraints.

An *s*-*t* cut in a graph is defined as a subset of arcs such that the removal of them leads to a 2-partition of the graph vertices: source set and sink set. All vertices in the source set are reachable from source vertex s after the removal of the arcs in the cut. All other vertices are in the sink set. A minimum *s*-*t* cut aims to find a cut in the graph such that the sum of arc weights from source set vertices to sink set vertices is minimum. Recall that the graph-cut construction uses arc weight to encode the unary and pairwise terms, thus, a minimum *s*-*t* cut corresponds to a minimized energy in Eq.(1.2). All pixels corresponding to vertices in the source set are labeled as object in the segmentation and all other pixels are labeled as background. Fig.1.1c shows an example graph-cut segmentation.

1.4 Graph-Search

Graph-search (also known as LOGISMOS [110, 64]) is widely used for medical image segmentation [52, 34, 87]. This technique can segment objects with complex topologies provided a pre-segmentation of the target object. However, it is easier to describe it when it's used to segment a *terrain-like* surface. Assume a 3D image has size $X \times Y \times Z$, then a *terrain-like* surface is a mapping $X \times Y \to Z$. In another word, for every (x, y) location, a 'column height' z defines the location of the surface at current (x, y)-column (Fig.1.2a). If we treat the 'column height' z_p at column p as a variable, then the possible values for the variable z_p are $\{0, 1, \ldots, Z-1\}$. And each terrain-like surface is equivalent to a multi-labeling for every variable z_p at every column location $p = (x, y) \in X \times Y$.

For each labeling l of a column p, we have an associated unary cost $D_p(z_p = l)$. Graph-search aims to find such a surface (i.e., multi-labeling) to minimize the sum of the unary terms. Without any pairwise term, this problem is trivial to solve in linear time. However, with the pairwise terms expressed as the smoothness constraint, min/max surface distance constraints, this problem requires the computation of a single minimum s-t cut in an appropriately constructed graph [106].

The column structure enables graph-search to incorporate a series of useful prior informations:

1. smoothness constraint: the column height difference at two adjacent columns can be restricted to be within a pre-defined threshold. In another word, for adjacent columns p, q, we require $|z_p - z_q| \leq \Delta_{pq}, (p, q) \in \mathcal{N}$, where Δ_{pq} is the smoothness constraint value. The column height difference between column p and column q is required to be no larger than Δ_{pq}. See Fig.1.2bfor a smoothness constraint example.
2. minimum surface distance constraint: when segmenting two interacting terrain-like surfaces, we may wish to segment them with a minimum distance between

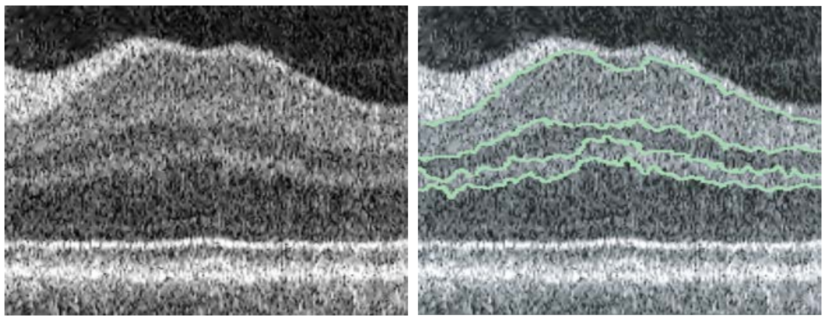

(a) OCT image (b) graph-search result

Figure 1.3: Graph-search segments multiple interacting surfaces in OCT images. The smoothness constraints and min/max surface distance constraints are vital for the successful segmentation.

the two surfaces. In another word, for each column p, two column heights z_p^1 and z_p^2 correspond to two surfaces S_1 and S_2, and we require $z_p^1 - z_p^2 \geq \delta_p^{12}$, where δ_p^{12} is the minimum surface distance between surfaces S_1 and S_2 at column p. See Fig.1.2c for a minimum surface distance constraint example.

3. maximum surface distance constraint: similar to minimum surface distance constraint, we can enforce two surfaces with a maximum surface distance between them, i.e., $z_p^1 - z_p^2 \leq \Delta_p^{12}$. See Fig.1.2c for a maximum surface distance constriant example.

These geometrical constraints prove to be quite helpful for various medical image segmentation problems, such as retinal layer segmentation [34], prostate and bladder segmentation [87], and brain gray and white matter segmentation [64], etc. Fig.1.3 shows an example of retinal layer segmentation in OCT images. Without the smoothness constraint and min/max surface distance constraints, this problem is hard to be segmented satisfactorily.

1.5 Summary Of Projects

Prior information, when combined in the labeling problem formulation, can help achieve more accurate and robust results. In my Ph.D. thesis, seven labeling problems with different prior informations are proposed with *efficient* and *exact* solutions.

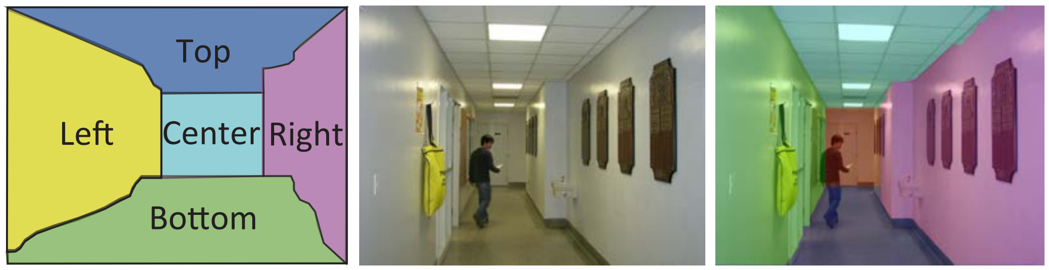
In Sec.1.5.1, a geometric scene labeling problem is proposed with geometric ordering constraints. This ordering prior prevents unreasonable geometric relationships in the final labeling, e.g., “top” pixel cannot appear under a “bottom” pixel. In Sec.1.5.2, the assumption of classical graph-cut that all user-specified seed pixels are accurate is removed. The prior that user-specified seed pixels may contain error requires a novel soft way to encourage the user-specified seed pixels to be respected in the final segmentation. Sec.1.5.3 proposes a multi-scale segmentation technique with interacting surface priors. One or more interacting surface(s) and the target object are simultaneously segmented to achieve more robust segmentations. One over-segmentation of the original image is also proposed to be interacting with the pixel-wise target object formulation to facilitate image information propagation in a larger neighborhood. Sec.1.5.4 proposes a volume agreement encouragement prior between the tumor volume segmented in PET scan and multiple tumor volumes segmented in CT scan from different breathing phases. This prior will lead to more robust tumor volume segmentations in different breathing phases. Sec.1.5.5 incorporates locational information along with traditional intensity information in the multi-modality context prior term to achieve more accurate multi-modality co-segmentation. In Sec.1.5.6, a novel star-shaped prior is proposed to be incorporated in the graph-search framework. Working directly in the voxel grid space, the proposed method eliminates the typical "unwrapping" procedure required to handle such complex shaped objects. In Sec.1.5.7, a novel gradient vector flow shape prior is incorporated into a graph-based multi-object segmentation framework. The flexible and powerful shape prior takes advantage of the shape information from a pre-segmentation while admitting efficient and exact solution to a simultaneous multi-object segmentation framework.

1.5.1 Labeling with ordering constraint

The geometric scene labeling problem in the natural images aims to segment the image into multiple coherent regions according to their geometric relationships. A useful special case aims to label every pixel as one of five labels: "top", "bottom", "left", "right", and "center". The labeling must obey certain *ordering constraints*: a "top" pixel may never appear under "bottom", a "left" pixel may never appear in the right of "right", etc. See Fig.1.4a for an illustration of the model. Fig.1.4b and 1.4c show an example application of the proposed 5-parts labeling model.

The order-preserving moves algorithm is proposed to solve the problem [57, 56]. It modifies the popular α-expansion algorithm to make sure the ordering constraints are respected in each move. In each move, the algorithm solves a minimum s-t cut problem. However, the problem does not guarantee globally optimal solution and may get stuck in a local minimum that's arbitrarily far from the optimal solution.

Tiered structure labeling method [30] is a dynamic programming based al-



(a) 5-parts model (b) input (c) scene labeling

Figure 1.4: Five-parts labeling model and its application on geometric scene labeling. The five-parts labeling model aims to label each pixel as one of {"top", "bottom", "left", "right", "center"}, with geometric constraints among the labeling.

gorithm which may solve the 5-labeling problem as a special case. This problem guarantees the optimal solution. However, it requires $O(N^{1.5})$ memory space to solve the problem, where N is the number of pixels in the image. The large memory consumption may cause difficulty processing large images and may make the algorithm run slow in practice due to cache issues.

A novel dynamic programming based algorithm is proposed to exactly solve the problem with only $O(N)$ memory. The theoretical running time is $O(N^{1.5})$, the same as tiered structure labeling [30]. But the practical running time is faster possibly due to the higher cache hit rate brought by our method's smaller memory footprint. Moreover, our method can be parallelized much easier.

1.5.2 Robust interactive segmentation

Although graph-cut is widely used for binary segmentation. It assumes the object and background seed pixels specified by user are perfectly accurate. This assumption is hardly true for challenging images, and the seed pixels given on touch

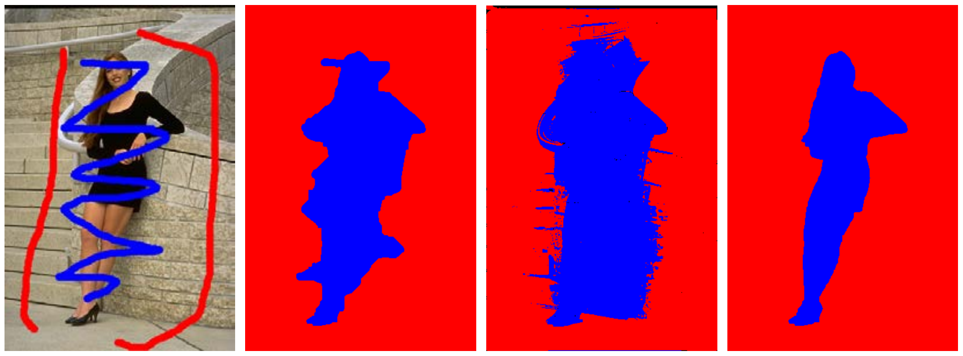


Figure 1.5: The proposed method tolerates errors in user-specified scribbles and is better at segmenting fine structures in an image.

screen. Fig.1.5a shows a careless user input, which is likely to happen on a touch screen device. Fig.1.5b is the segmentation of traditional graph-cut, which assumes all user input are perfectly accurate.

Various techniques have been proposed to use user seed as soft constraint. An et al. [4] formalizes the image editing problem (which is a “soft” segmentation instead of binary segmentation) as a quadratic optimization problem based on pixel affinity matrix among all pairs of pixels. Li et al. [54] observed the above optimization problem is essentially a smooth function with a sparse set of constraints. Based on this observation, the segmentation is approximately achieved by the composition of a series of radial basis functions in the pixel appearance space. However, both methods only output a continuous probability map of the image instead of a binary segmentation.

Liu et al. proposes a method [55] which allows user to interactively override previous erroneous seed pixels. However, the new seed pixels are again assumed to be perfectly accurate. Sener et al. [75] develops a heuristic preprocessing step to identify erroneous part of the user input and exclude them from being used as seed pixels in the following graph-cut segmentation. Subr et al. [89] uses a dense conditional random field (CRF) to infer the segmentation from inaccurate user input. The dense CRF is approximated by an embedding in a low-dimensional Euclidean space. Fig.1.5c shows an example segmentation of Subr's method [89].

A novel ratio energy is proposed which simultaneously minimizes a graph-cut energy on the numerator and maximizes a seed utility term on the denominator. The seed utility term encourages the seed pixels to be labeled as the way user specifies them in general, but allows it to have opposite label if strong evidence is present.The ratio energy can be efficiently and *exactly* minimized by the Newton's algorithm for ratio problem (i.e., Dinkelbach's algorithm). The proposed algorithm solves a series of minimum *s*-*t* cut problem in a series of slightly modified graphs. The algorithm runs fast in practice and gives good segmentation on user input with errors. See Fig.1.5d for an example of the proposed robust interactive segmentation method.

1.5.3 Surface-object segmentation with multi-scale technique

Graph-search and graph-cut are combined to segment lung tumor in the poor-quality Mega-Voltage Cone-Beam CT (MVCBCT) in [84]. Graph-cut is used to segment the tumor, while the graph-search is used to segment the adjacent lung boundary. Interactions between graph-cut and graph-search is added to ensure the tumor segmentation never "leak" beyond the lung boundary. This interaction proves

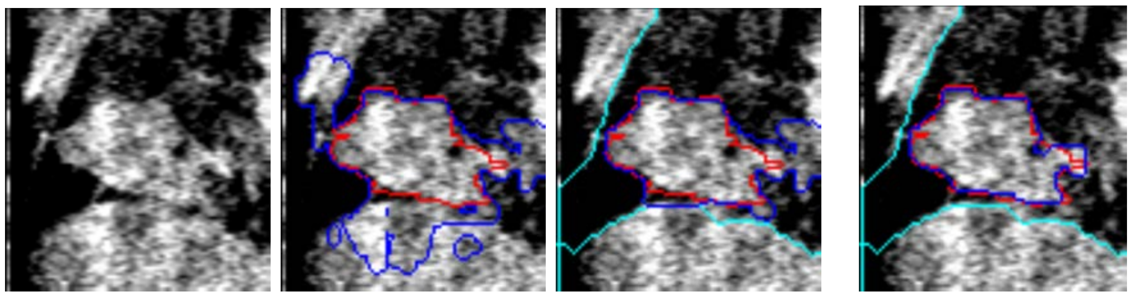


Figure 1.6: Surface-object segmentation with multi-scale technique of lung tumor in MVCBCT. Red contours in all images are drawn by human expert. Blue contours are segmentation using corresponding methods.

to be useful for segmenting lung tumor robustly. Fig.1.6c segments two interacting lung boundaries simultaneously with tumor, which avoids the heavy leakage of tumor segmentation without using such surface information (Fig.1.6b, red contour is by human expert).

Most of current segmentation only uses small neighborhood in images, such as 4-neighborhood for 2D images and 6-neighborhood for 3D images. In [21], Cour et al. shows that the larger the neighborhood is in the normalized cut [79], the more accurate the segmentation is. To prove this point, they build affinity matrices at different scales and enforce cross-scale constraints that encourage the segmentation at coarse scale to agree with the segmentation at finer scale. The segmentation is improved even though they only used a naive down-sampling on a regular grid to implement the multi-scale idea.

Kim et al. [44] uses a *data-driven* method to define multiple over-segmentation of the image. All segments of the over-segmentation are interacting with each other, which encourages large-scale information propagation. The pixel-wise segmentation is encouraged to agree with the labeling of the over-segmentation regions. The problem is reduced to an inversion of a large matrix, which is solved by computing a random walk with restart. For 3D images, the matrix size is prohibitively large to be inverted. And the completely-connected graph among all regions in the segmentation also makes the running time increases quickly with the number of regions in the over-segmentation.

We propose a novel multi-scale method incorporating information from an over-segmentation of the original image. Although the formulation is similar to [44], the problem is reduced to a binary labeling problem in the proposed method, instead of a continuous generative model estimation problem. The resulting binary labeling problem can be efficiently and exactly optimized by computing a single minimum s-t cut in an appropriately constructed graph. Moreover, the completely-connected graph consisting of all segments in the over-segmentation is replaced with a graph using neighborhood of fixed radius. The proposed multi-scale technique exploits the larger neighborhood among the segments in the over-segmentation, and the parent-children relationship between one segment and all pixels within it.

The proposed method is applied to the lung tumor segmentation in MVCBCT. Fig.1.7 describes the workflow incorporating both the multi-scale technique and the surface-object interaction prior. Experiment shows that the proposed multi-scale technique boosts the segmentation accuracy on top of the surface-object segmentation technique. Fig.1.6d shows an example of the multi-scale surface-object segmentation. Compared to Fig.1.6c, which does not use the multi-scale technique, the segmentation

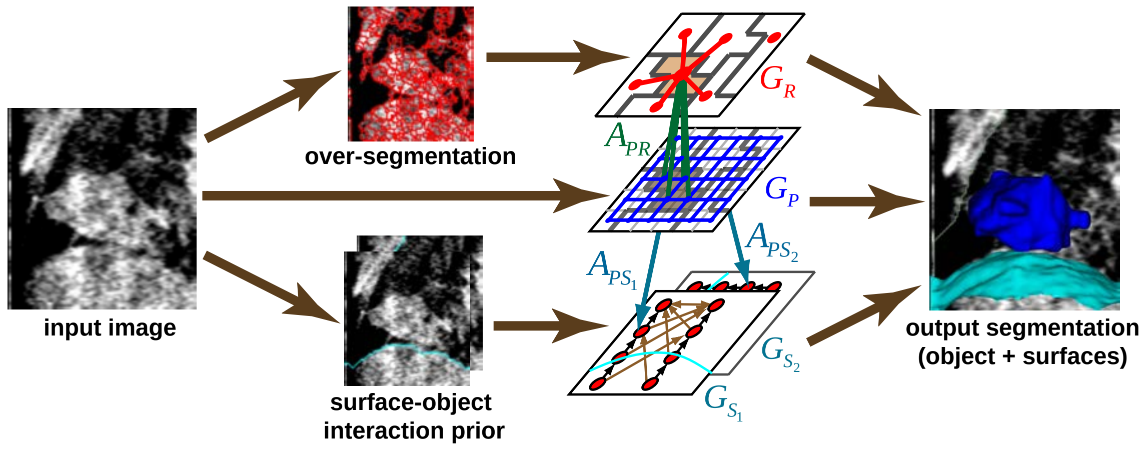


Figure 1.7: Workflow of the multi-scale segmentation with surface-object interaction prior. Surface-object interaction prior is encoded in the resulting graph. Region-wise information from an over-segmentation of the original image is also incorporated. The target object and the interacting surfaces are segmented simultaneously.

is further improved.

1.5.4 4DCT-PET co-segmentation

A PET-CT co-segmentation technique is proposed by Song et al. in [83] to segment tumor volumes in PET-CT scan pairs. Two graph-cut based graphs are constructed for PET and CT respectively to segment two tumor volumes in PET and CT, respectively. Interaction priors between PET and CT are enforced to encourage two segmented volumes agree with each other.

We extend the co-segmentation method to segment 4DCT-PET pairs, which typically involves 10 different CT scans showing different breathing phases of the same patient, and one PET scan. Interaction priors are introduced between PET scan and each CT scan in different phases to encourage the segmentation to agree with each other. Fig.1.8 shows one example.

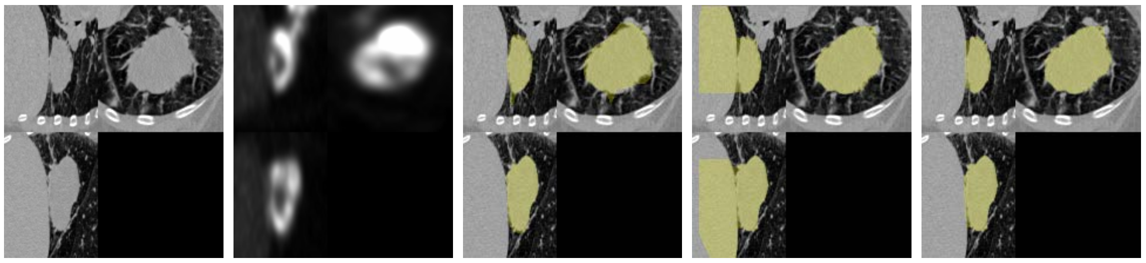


Figure 1.8: Illustrative results of one phase in 4DCT-PET co-segmentation. Note the guidance of PET prevents the segmentation from leaking into the adjacent normal tissues.

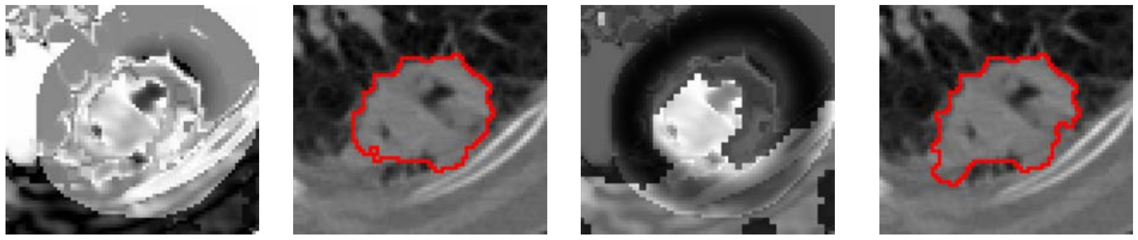
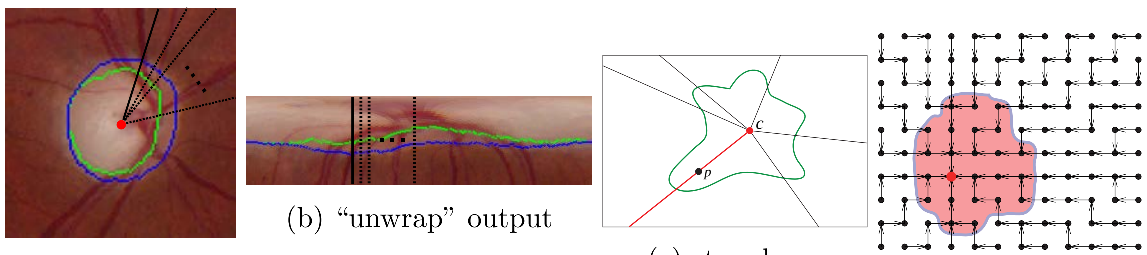


Figure 1.9: Incorporating location context into the traditional intensity context leads to more accurage segmentation. The left bottom corner of the tumor is incorrectly segmented as background given no locational information (b). With the help of location context, the co-segmentation correctly finds the whole tumor volume (d).

1.5.5 Incorporating location context into co-segmentation

Co-segmentation allows different segmentations of tumor volume in CT and PET images. The difference is modulated by a context term. One key desirable property is to have the metabolical PET information anchor the tumor segmentation in both PET and CT roughly around the highly active region indicating tumor. At the same time, flexibilty should still be be allowed around tumor boundary because of different characteristics of CT and PET images, and potential registration error.

However, most current co-segmentation [83, 11] only uses intensity difference



(a) radial sampling (b) "unwrap" output (c) star-shape (d) digital ray

Figure 1.10: To segment circular or tubular structure, classical graph-search usually requires "unwrapping" from the Cartesian coordinate system (a) to a polar coordinate system (b). (c) shows a star-shaped object example. (d) shows a consistent digital ray system, which is used to enforce a star-shape prior and avoid resampling in the "unwrapping" procedure.

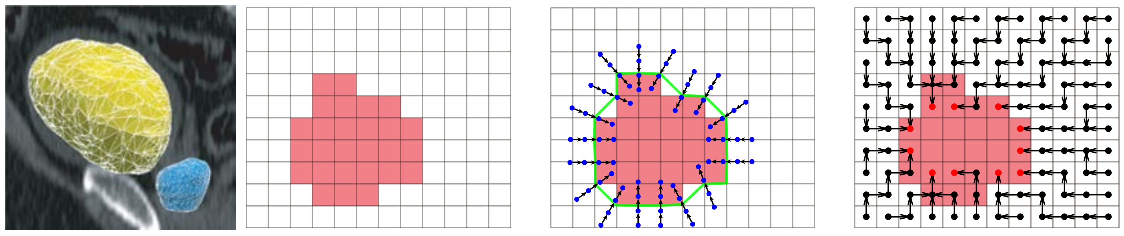
between CT and PET region cost to compute the context term, omitting the valuable locational prior. We propose a truncated quadratic function to model the location prior based on the distance to the highly metabolically active PET regions. Combined with the traditional intensity based context, the proposed inter-modality context term leads to more accurate co-segmentation, as shown in Fig.1.9.

1.5.6 Segmentation with star-shape prior

While the column structure enforced in the graph-search enables enforcement of the useful smoothness and surface distance constraints, it also restricts the topology of target surface to be terrain-like. For objects with other shapes, such as circular surface, an "unwrapping" procedure usually needs to be taken. The "unwrapping" procedure starts with a circle center, then resamples the image along the radial direction. This resampling is repeated for different angles (for example, 360 different angles), resulting in a terrain-like surface (see Fig.1.10a and 1.10b). The graph-search is conducted in the resampled space and then transformed back to the original image space.

However, the sampling density is not uniform for the “unwrapping” procedure. The segmentation accuracy at under-sampled outer regions may be limited by the low sampling density. While the overly-sampled inner regions may have intersecting graph-search columns and suffer from column-intersection/mesh-folding problems [109, 67, 64, 42].

A novel segmentation method enforcing a star-shaped prior is introduced to eliminate the “unwrapping” procedure. A *star-shaped* object is an object for which there exists a star center point c such that for every other point p in the object, the line segment cp lies completely within the object. See Fig.1.10c for an example of star-shaped object. A *discrete* resampling tree rooted at the star center can be efficiently constructed with nice geometrical properties by using consistent digital ray method [19]. Every digital line segment connecting the image boundary pixel and the star center pixel is equivalent to a column in the graph-search (Fig.1.10d). However, unlike traditional graph-search, different columns can be merged as they approach the star center. Fortunately, we will show that this difference will not pose difficulty for us to enforce the smoothness constraint and min/max surface distance constraints as in traditional graph-search. Thus, given the star center pixel location, we can conduct graph-search on the consistent digital ray tree in the original image space, with all the powerful constraints provided by graph-search.



(a) pre-seg mesh (b) pre-segmentation (c) resample from mesh (d) GVF shape prior

Figure 1.11: When "unwrapping" is not possible, classical graph-search builds graph based on triangulated mesh of an initial pre-segmentation. (a) shows a triangulated mesh on a pre-segmentation of prostate and bladder in CT. (c) demonstrates the resampling along columns at mesh vertices. Red grids are pre-segmentation result (b); green line is triangulated mesh; blue columns are columns for graph-search. Note the blue columns may intersect at high curvature regions, which is undesirable. Gradient vector flow shape prior shown in (d) can be used to enforce shape prior directly in original voxel grid space.

1.5.7 Segmentation with GVF shape prior

Although star-shaped prior is flexible to handle much shape variation, it's not easy, or sometimes impossible, to identify the star center in 3D images. In some situations, it's more convenient to provide a pre-segmentation as a shape prior for graph-search based segmentation [109, 87, 86]. A *pre-segmentation* is a rough segmentation of the target object which provides rough shape and location information, but not accurate boundary location information.

A traditional graph-search based method will build triangulated mesh on the pre-segmentation surface, grow columns on the vertices of the triangles on the mesh, and resample the image along these columns. The graph-search segmentation is conducted in the resampled column space, and the segmentation result has to be converted back to the original image space. Similar uneven resampling problems exist for this mesh-based graph-search method as the “unwrapping”-based method. Various techniques have been proposed to handle this problem [81, 86, 64, 109].

A novel gradient vector flow (GVF) shape prior is proposed to encode the pre-segmentation shape information directly in the voxel grid space. This eliminates the need to build mesh and resample the image. Working directly in the voxel space also enable straightforward incorporation of a multi-object simultaneous segmentation framework which works not only for nesting surfaces, but also for excluding surfaces. In Sec.8, two applications showing nesting and excluding surfaces are used to demonstrate the proposed GVF shape prior segmentation method.

1.6 Contribution

The contribution of this proposal is the formulation of a series of applications as labeling problem with different prior informations. All proposed formulations can be *efficiently* and *exactly* minimized. This enables the problems to be effectively solved in practice.

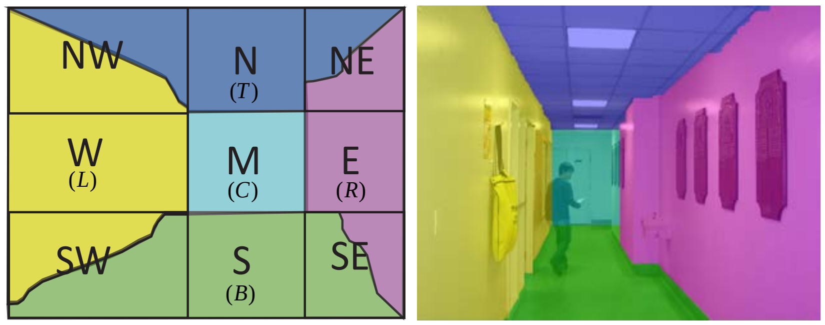
Now we summarize the proposed works from the point view of labeling problems with different prior informations, and give pointers to more detailed description of the proposed works. The problem of labeling with ordering constraints is a multi-labeling problem with geometric location ordering constraints of the labeled regions (Chapter.2). The robust interactive segmentation is a binary labeling problem which removes the hard seed constraints from graph-cut. This makes the interactive segmentation more robust to human interaction error (Chapter.3). The surface-object segmentation with multi-scale technique combines a multi-labeling problem (graph-search for surface segmentation), a binary labeling problem (pixel-wise graph-cut for object), and a second binary labeling problem (region/segment-wise graph-cut for object over-segmentation). This combination boosts the segmentation accuracy by segmenting an auxiliary interaction surface and allowing larger interacting neighborhood in the object segmentation (Chapter.4). The 4DCT-PET co-segmentation combines multiple binary labeling problem (one binary labeling for tumor volume in each CT scan in different phase and PET scan) to achieve accurate segmentation in multiple CT phases (Chapter.5). Star-shape prior and GVF shape prior simplifies a multi-labeling problem (graph-search) in the resampled image space to a binary labeling problem in the original image space with the help of consistent digital ray given star center pixel (star-shape prior, Chapter.7), and the gradient vector flow of pre-segmentation (GVF shape prior, Chapter.8).

CHAPTER 2
LABELING WITH ORDERING CONSTRAINTS BY FAST DYNAMIC PROGRAMMING

Many computer vision applications can be formulated as labeling problems. However, multilabeling problems are usually very challenging to solve, especially when some ordering constraints are enforced. We solve a five-parts labeling problem proposed in [57, 56]. In this model, one wants to find an optimal labeling for an image with five possible parts: “left”, “right”, “top”, “bottom” and “center”. The geometric ordering constraints can be read naturally from the names. No previous method can solve the problem with globally optimal solutions in a linear space complexity. We propose an efficient dynamic programming based algorithm which guarantees the global optimal labeling for the five-parts model. The time complexity is $O(N^{1.5})$ and the space complexity is $O(N)$, with N being the number of pixels in the image. In practice, it runs faster than previous methods. Moreover, it works for both 4-neighborhood and 8-neighborhood settings, and can be easily parallelized for GPU.

2.1 Introduction

Given an image, the multilabeling problem seeks to assign a label to each pixel from a set of fixed labels, which, in general, is NP-hard [16]. Multilabel oprimization is a very active area of research in computer vision since a wide variety of vision problems can be formulated as multilabeling problems. Ishikawa developed an exact optimization method for Markov Random Fields with convex priors [39], which was



(a) (b)

Figure 2.1: The five-parts labeling model. (a) The partition of the image into nine regions. The letter in the parenthesis indicates the label of each region. (b) An example labeling. Although the person in the hall way occludes significant area of the image, our method can correctly label the end of the hallway.

among the first computational frameworks for efficiently solving the multilabeling problems. Wu and Chen's algorithm for convex multilabeling works in a more general setting, restricting the label transition between two neighboring pixels in a certain range of the linearly ordered labels [105]. For more general cost functions, Boykov*et al.*'s α-expansion based graph cut approach [16] is widely used in the vision community due to its accuracy and efficiency. However, this method provides no optimality guarantees in some cases, including the cases studied in this work.

Recently, the introduction of label ordering constraints into the multilabel optimization, which allows substantially generalized cost functions, attracts noticeable attention [30, 57, 56, 88]. In [57], Liu et al. proposed a five-parts labeling model with the ordering constraints, in which the image is to be labeled into five parts, namely, "left", "right", "top", "bottom" and "center", as shown in Fig. 2.1. The geometric ordering constraints can be read from the names: (1) a pixel labeled as "left" cannot be to the *right* of a pixel with any other label; (2) a pixel labeled as "right" cannot be the *left* of a pixel with any other label; (3) a pixel labeled as “top” cannot be *below* a pixel with any other label; (4) a pixel labeled as “bottom” cannot be *above* a pixel with any other label; and (5) if a pixel p labeled as “center” has a neighbor with a different label, then the neighbor pixel has to be labeled as “left”, “right”, “top”, or “bottom” if it is to the left, right, above, or below p, respectively. The last constraint indicates that the “center ” region is a rectangle. Note that not all parts have to be present.

To solve this five-parts labeling problem, Liu et al. [57] proposed the *ordering preserving moves* for the graph cut optimization, which was demonstrated more effective than the α-expansion method [16]. However, their method does not guarantee to find the globally optimal solution.

Our main technical contribution is a dynamic programming algorithm for computing a globally optimal five-parts labeling of an $N = n \times n$ image in $O(N^{1.5})$ time. This algorithm runs quite fast in practice, taking just seconds to compute an optimal labeling for a rather large images. In addition, our algorithm takes only a linear $O(N)$ memory space.

The key idea for solving the five-parts labeling problem in our algorithm is to guess the possible “center” rectangles, and then for each rectangle, compute an optimal two-labeling for each of those four conner regions incident to the rectangle (Fig. 2.1a) by finding a shortest path. Note that there are $O(N^2)$ possible rectangles, and thus a straightforward algorithm, with an efficient $O(N)$ shortest path algorithm called for each possible rectangle, would take $O(N^3)$ time, which is too slow for practical use. Fortunately, by judiciously characterizing the intrinsic structure of the problem, we are able to improve the running time by a factor of $O(N^{1.5})$.

We evaluate our algorithm on the geometric class scene labeling problem [37], where the goal is to assign each pixel a rough geometric label, such as “sky”, “ground”, “surface above ground”, etc. Experiment shows our algorithm runs faster and more robustly on average than the order-preserving moves method [56]. The standard deviation of the execution times over hundreds of test images with the same size is almost 0 for our method, while it is comparable to the mean execution time for the order-preserving moves method. By intentionally adding Gaussian noise, we observe little effect on the execution time of our algorithm, while a big deterioration is observed for the order-preserving moves method. The average labeling accuracy of the two methods is highly comparable over all the test datasets, though we do find some image datasets, for which our algorithm obtains clearly superior labeling.

Related Work. Felzenszwalb and Veksler recently proposed a tiered scene model which is more general than the five-parts model [30]. In this model, the image is first divided by two horizontal curves into the top, middle and bottom regions, and the middle region is further subdivided vertically into subregions. They give a dynamic programming based algorithm which runs in $O(N^{1.5}K)$ time for a Potts-like model and in $O(N^{1.5}K^2)$ time for more general models, in which N is the size of the image, and K is the number of possible labels in the middle region. However, the space complexity of the algorithm is $O(N^{1.5})$, which could be problematic in practical use for a large image. For example, for an 500×500 image, an $O(N^{1.5})$ memory algorithm will require hundreds of times more memory than an $O(N)$ memory algorithm. In addition, only the 4-connected neighborhood setting case is presented in [30]. Another closely related work is Strekalovskiy and Cremers' multilabeling framework with generalized ordering constraints based on spatially continuous optimization [88]. This framework includes both the five-parts model and the tiered scene model as special cases, and can deal with even more complex ordering constraints. However, this algorithm does not guarantee global optimality and does not run fast enough in practice. The execution time reported in [88] is 90 seconds for solving the five-parts labeling problem on a 640×480 image using CUDA parallel implementation; while our algorithm takes only about 17 seconds with no parallel implementation.

2.2 Model

Given an image $\mathcal{I}$ with a set $\mathcal{P}$ of $N = n \times n$ pixels and a set $\mathcal{L}$ of labels, the pixel labeling problem seeks a labeling f that assigns a label $f_p \in \mathcal{L}$ to each pixel $p \in \mathcal{P}$, such that the energy function of the following form is minimized.

$$\mathcal{E}(f) = \lambda \sum_{p \in \mathcal{P}} D_p(f_p) + \sum_{(p,q) \in \mathcal{N}} V_{pq}(f_p, f_q) \tag{2.1}$$

In Eq. (2.1), $\mathcal{N}$ is a neighboring system defined on $\mathcal{P}$. A 4-connected neighborhood setting is assumed for the following sections. But our approach can be easily extended to 8-connected neighborhood. $D_p(f_p)$ is the data term, which reversely measures the likelihood of assigning label f_p to the pixel p. $V_{pq}(f_p, f_q)$ is the smoothness term, which is the penalty we pay for assigning labels f_p and f_q to neighboring pixels

Table 2.1: Ordering constraints penalty table.

Vertical Neighbors $p=(x,y), q=(x,y+1)$					
$f_p \backslash f_q$	L	R	C	T	B
L	0	∞	∞	∞	w_{pq}
R	∞	0	∞	∞	w_{pq}
C	∞	∞	0	∞	w_{pq}
T	w_{pq}	w_{pq}	w_{pq}	0	∞
B	∞	∞	∞	∞	0

Horizontal Neighbors $p=(x,y), q=(x+1,y)$					
$f_p \backslash f_q$	L	R	C	T	B
L	0	∞	w_{pq}	w_{pq}	w_{pq}
R	∞	0	∞	∞	∞
C	∞	w_{pq}	0	∞	∞
T	∞	w_{pq}	∞	0	∞
B	∞	w_{pq}	∞	∞	0

p and q, respectively.

Due to the intractability of the general labeling problem, most of previous work assumes V_{pq} is of a particular form (e.g, convex or metric). Here in the five-parts labeling problem, we make no assumptions on either D_p or V_{pq}. Instead we consider the problem of minimizing $\mathcal{E}(f)$ over a restricted class of labelings. Specifically, D_p and V_{pq} can be arbitrary.

The label set $\mathcal{L}$ includes five labels, $'L'$, $'R'$, $'C'$, $'T'$, $'B'$, which represent "left", "right", "center", "top" and "bottom", respectively. This model enforces the ordering constraints by letting $V_{pq}(f_p, f_q) = \infty$ if f_p and f_q are not allowed. Because we're minimizing the energy, such labeling will not be feasible in a minimized energy. For example, we set $V_{(x,y)(x,y+1)}('B','C') = \infty$ to prevent a pixel (x,y) labeled as "bottom" from appearing above a pixel $(x,y+1)$ labeled as "center". If $f_p = f_q$, then $V_{pq}(f_p, f_q) = 0$. Finally, if f_p and f_q satisfy the ordering constraints, $V_{pq}(f_p, f_q) = w_{pq} > 0$.

The complete set of ordering constraints is described in Table.2.1. An example labeling satisfying the 5 label ordering constraint model is presented in Fig. 2.1a.

2.3 Method

This section presents our $O(N^{1.5})$ time algorithm for solving the five-parts labeling problem by dynamic programming.

A key observation for our algorithm is as follows. For a fixed center rectangle M, one can extend the four sides of M to divide the image $\mathcal{I}$ into nine regions, denoted by $NW, N, NE, W, M, E, SW, S, SE$, as shown in Figure 2.1(a). Due to the ordering constraints, the label for each of the regions N, W, M, E, and S is determined, and each of the four corner regions NW, NE, SW, and SE is labeled with at most two different labels, more precisely, regions NW, NE, SW, and SE are labeled with 'L' and 'T', 'T' and 'R', 'L' and 'B', and 'B' and 'R', respectively. Clearly, the energy on each of the regions N, W, M, E, and S is well defined if the center rectangle M is fixed. Thus, the problem is reduced to computing an optimal two-labeling for each of those corner regions. We further observe that, in each of those corner regions, the boundary between the two labeled parts forms a monotone path with respect to both horizontal and vertical directions (Fig. 2.1(a)). Our main idea is to optimally solve the two-labeling problem for each corner region by computing a shortest monotone path, which takes $O(N)$ time. Note that there are $O(N^2)$ possible center rectangles in total. It thus takes $O(N^3)$ time for solving the five-parts labeling problem. Interestingly, we are able to batch the computation of all $O(N)$ shortest paths in $O(N)$ time. Hence, the running time of our algorithm can be reduced to $O(N^2)$. Furthermore, by judiciously exploring the intrinsic structure of the problem, we can further improve the running time to $O(N^{1.5})$.

Sections 2.3.1 and 2.3.2 show that after $O(N)$ time preprocessing, the minimum energy of the five-parts labeling can be computed in $O(1)$ time for each possible center rectangle, and the speedup of the algorithm is presented in Section 2.3.3.

2.3.1 Computing energy for non-corner regions

Given a center rectangle M specified by its two diagonal corner points, (x_1, y_1) and (x_2, y_2) with $x_1 \leq x_2, y_1 \leq y_2$, we show that the energy for each of the regions N, W, M, E and S, can be computed in $O(1)$ time after $O(N)$ time of preprocessing.

The idea is to first pre-compute the integral data cost image [98] $\mathcal{C}^l_{data}(x, y)$ of each label $l \in \mathcal{L}$ for the data term of the energy function, with $\mathcal{C}^l_{data}(x, y) = \sum_{1\leq i\leq x, 1\leq j\leq y} D_{p(i,j)}(f_p = l)$. Note that $\mathcal{C}^l_{data}(\cdot, \cdot)$ can be computed in $O(N)$ time. Then we compute the integral row-smoothness cost $\mathcal{C}^{(T,C)}_{row\_sm}(x, y) = \sum_{1\leq i\leq x} V_{p(i,y),q(i,y-1)}(f_p = C, f_q = T)$ for the label transition from 'T' on Row $y - 1$ to 'C' on Row y. Similarly, we can compute $\mathcal{C}^{(C,B)}_{row\_sm}(x, y)$ for the label transition from 'C' to 'B'. In addition, we define the integral column-smoothness cost $\mathcal{C}^{(L,C)}_{col\_sm}(x, y) = \sum_{1\leq j\leq y} V_{p(x-1,j),q(x,j)}(f_p = L, f_q = C)$ for the label transition from 'L' on Column $x - 1$ to 'C' on Column x. Similarly, $\mathcal{C}^{(C,R)}_{col\_sm}(x, y)$ can be computed. Note that all these tables can be computed in $O(N)$ time. Now we can compute the energy for each of the regions N, W, M, E and S in $O(1)$ time, as follows.

$$\mathcal{E}_N = \mathcal{C}^T_{data}(x_2, y_1 - 1) - \mathcal{C}^T_{data}(x_1 - 1, y_1 - 1); \tag{2.2}$$

$$\mathcal{E}_W = \mathcal{C}^L_{data}(x_1 - 1, y_2) - \mathcal{C}^L_{data}(x_1 - 1, y_1 - 1); \tag{2.3}$$

$$\mathcal{E}_E = \mathcal{C}^R_{data}(n, y_2) - \mathcal{C}^R_{data}(x_2, y_2) - \mathcal{C}^R_{data}(n, y_1 - 1) + \mathcal{C}^R_{data}(x_2, y_1 - 1); \tag{2.4}$$

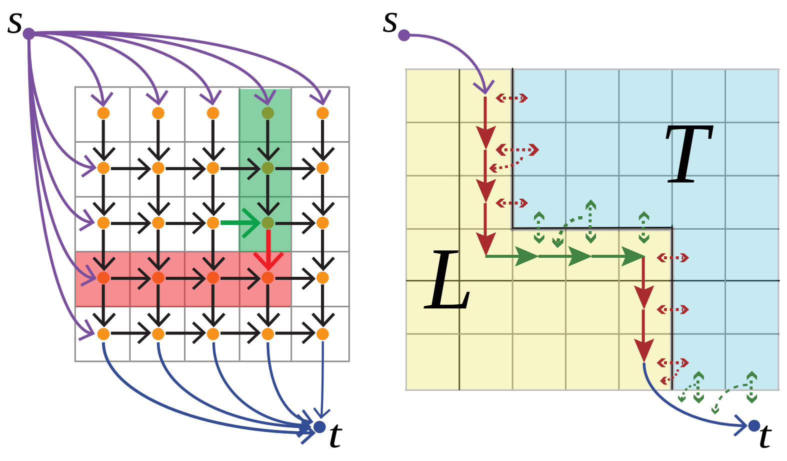
$$\mathcal{E}_S = \mathcal{C}^B_{data}(x_2, n) - \mathcal{C}^B_{data}(x_1 - 1, n) - \mathcal{C}^B_{data}(x_2, y_2) + \mathcal{C}^B_{data}(x_1 - 1, y_2); \tag{2.5}$$

$$\begin{aligned}\mathcal{E}_M = {} & \mathcal{C}^C_{data}(x_2, y_2) - \mathcal{C}^C_{data}(x_1 - 1, y_2) - \mathcal{C}^C_{data}(x_2, y_1 - 1) + \mathcal{C}^C_{data}(x_1 - 1, y_1 - 1) \\ & + \mathcal{C}^{(T,C)}_{row\_sm}(x_2, y_1) - \mathcal{C}^{(T,C)}_{row\_sm}(x_1 - 1, y_1) + \mathcal{C}^{(C,B)}_{row\_sm}(x_2, y_2) - \mathcal{C}^{(C,B)}_{row\_sm}(x_1 - 1, y_2) \\ & + \mathcal{C}^{(L,C)}_{col\_sm}(x_1, y_2) - \mathcal{C}^{(L,C)}_{col\_sm}(x_1, y_1 - 1) + \mathcal{C}^{(C,R)}_{col\_sm}(x_2, y_2) - \mathcal{C}^{(C,R)}_{col\_sm}(x_2, y_1 - 1)\end{aligned} \tag{2.6}$$

2.3.2 Computing min energy for corner regions

For each of the corner regions NW, NE, SW and SE (Fig. 2.1(a)), we essentially need to solve an optimal 2-labeling problem given a fixed center rectangle M. The idea is to compute a shortest monotone path which completely separates the two parts with different labels. Now we illustrate on the corner region NW that after $O(N)$ preprocessing, given a fixed center rectangle M, each of those 2-labeling problem can be solved in $O(1)$ time.

Assume the lower-right corner of the NW region is (x_0, y_0). We construct the following directed acyclic graph (DAG) $G_{(x_0,y_0)}$ to compute a shortest path minimizing the energy function $\mathcal{E}_{NW}(x_0, y_0)$. The node set consists of two dummy nodes, a *source* s and a *sink* t, and N pixel nodes $v_{(x,y)}$ with each corresponding to exactly one pixel $\mathcal{I}(x, y)$ in the image $\mathcal{I}$.



(a) (b)

Figure 2.2: (a) Graph construction for solving the 2-labeling problem on Region NW. The horizontal edge (green) incorporates the data term of part of the current column (green block). The vertical edge (red) incorporates the data term of part of the current row (red block). (b) Distribution of the smoothness penalties to the edges. The dotted double-arrows represent the smoothness penalties between two pixels with different labels. The dotted single-arrows shows how the smoothness penalties are assigned to the edges. The smoothness penalties indicated by the red (green) double-arrows are distributed to the red (green) edges.

Now we define directed edges in $G_{(x_0,y_0)}$. Note that the boundary between the L-part (whose pixels are labeled as "left") and the T-part (whose pixels are labeled as "top") of the NW region is monotone to both horizontal and vertical directions. Thus, for every node $v_{(x,y)}$ with $1 \leq x \leq x_0$ and $1 \leq y < y_0$, one vertical edge to $v_{(x,y+1)}$ is introduced. For every node $v_{(x,y)}$ with $1 \leq x < x_0$ and $1 < y \leq y_0$, one horizontal edge to $v_{(x+1,y)}$ is introduced. We define the *boundary path* in $G_{(x_0,y_0)}$ as a path whose nodes corresponding to the upper envelop of an L-part of the NW region. We further notice that a boundary path could start from any node in the first row or first column, and may end at any node in the last row. Hence, we add one directed edge from s to every node in the first row and the first column, and a directed edge from each node in the last row to the sink t (Fig. 2.2(a)).

We assign edge costs to encode the energy function in $G_{(x_0,y_0)}$. For notation convenience, denote by $rPreSum(L; x, y) = \sum_{1 \leq i \leq x} D_{p(i,y)}(f_p = L)$ the total sum of the data cost of the first x pixels of Row y, which are labeled as "left"; and by $cPreSum(T; x, y) = \sum_{1 \leq j \leq y} D_{p(x,j)}(f_p = T)$ the total sum of the data cost of the first y pixels of Column x, which are labeled as "top".

For any two vertically neighboring pixels $p(x, y)$ and $r(x, y + 1)$, $(1 \leq x \leq n, 1 \leq y < n)$, there is a downward edge from p to r. If both v_p and v_r are on the boundary path (i.e. p and r are labeled as "left"), then pixel $q(x + 1, y)$ is labeled as "top". Hence, a smoothness penalty $V_{pq}(f_p = L, f_q = T)$ needs to be enforced. In addition, all pixels from the leftmost pixel of Row $y + 1$ to pixel r are labeled as “left”. We thus assign a cost $c_e(v_p, v_r)$ to the edge $e(v_p, v_r)$, with

$$c_e(v_p, v_r) = V_{pq}(f_p = L, f_q = T) + rPreSum(L; x, y+1) \tag{2.7}$$

Specifically, the edge from s to each node in the first row $v_r(x, 1)$ can be treated as a special case of vertical edges, with cost:

$$c_e(s, v_{r(x,1)}) = rPreSum(L; x, 1) \tag{2.8}$$

For any two horizontal neighbor pixels $p(x, y)$ and $r(x+1, y)$, there is a rightward edge from p to r. If both v_p and v_r are on the boundary path (i.e. p and r are labeled as “left”), then pixel $q(x+1, y-1)$ is labeled as “top”. Hence, a smoothness penalty $V_{rq}(f_r = L, f_q = T)$ needs to be enforced. In addition, all pixels in Column y starting from the topmost pixel to pixel q are labeled as “top”, and pixel r is labeled as “left”. Thus the cost of edge $e(v_p, v_r)$ is

$$c_e(v_p, v_r) = V_{qr}(f_q = T, f_r = L) + D_r(f_r = L) + cPreSum(T; x+1, y-1) \tag{2.9}$$

Specifically, the edge from s to each node in the first column $v_{r(1,y)}$ can be treated as a special case of horizontal edges, with cost:

$$c_e(s, v_{r(1,y)}) = V_{q(1,y-1),r}(f_q = T, f_r = L) + D_r(f_r = L) + cPreSum(T; 1, y-1) \tag{2.10}$$

Finally, we need to set the costs for the edges connected to the sink t. For a pixel $p(x, y_0)$ in the last row of the NW region, if v_p is the last node on a boundary path, then p is labeled as “left”, and each pixel $q(i, y_0)$ right after p in the same row (i.e. $x < i \leq x_0$) is labeled as “top”. However, the pixel $r(i, y_0+1)$ immediately below $q(i, y_0)$ is labeled as “left”, as r is in Region W. Thus, we assign a cost $c_e(v_p, t)$ to the edge $e(v_p, t)$ to enforce the smoothness penalty for those label changes. In addition, the data cost for those columns after Column x also need to be enforced. Hence, we have

$$\begin{aligned} c_e(v_p, t) = & V_{p,q(x+1,y_0)}(f_p = L, f_q = T) \\ & + \sum_{x<i\leq x_0} V_{q(i,y_0),r(i,y_0+1)}(f_q = T, f_r = L) \\ & + \mathcal{C}^T_{data}(x_0, y_0) - \mathcal{C}^T_{data}(x, y_0) \end{aligned} \tag{2.11}$$

This completes the construction of $G_{(x_0,y_0)}$ for computing $\mathcal{E}_{NW}(x_0, y_0)$. A shortest s-to-t path can be computed in $O(N)$ time using topological ordering of this DAG, which specifies an optimal 2-labeling for the region NW. However, this is far from good enough to achieve our goal to compute $\mathcal{E}_{NW}(x_0, y_0)$ in $O(1)$ time after an $O(N)$ preprocessing.

Observe that for $x_0 \leq x'_0$ and $y_0 \leq y'_0$, the induced graph of $G_{(x_0,y_0)}$ after removing its sink is a subgraph of the induced graph of $G_{(x'_0,y'_0)}$ after removing the sink. Thus we can compute all $\mathcal{E}_{NW}(x_0, y_0)$ for $1 \leq x_0 \leq n$ and $1 \leq y_0 \leq n$, as follows. First, construct the graph $G_{(n,n)}$, and compute a *shortest path tree* from the source s in $O(N)$ time. Then, for each node $v_{(x_0,y_0)}$, we introduce the sink $t(x_0, y_0)$ and its incident edges, as we do for the construction of $G_{(x_0,y_0)}$. Thus, it take additional $O(N^{0.5})$ time to find a shortest path from s to $t(x_0, y_0)$ from the computed shortest path tree, rather than from scratch. In this way, it takes $O(N^{1.5})$ time to compute all $\mathcal{E}_{NW}(x_0, y_0)$ for $1 \leq x_0 \leq n$ and $1 \leq y_0 \leq n$.

Interestingly, we can further improve our algorithm. Consider all $\mathcal{E}_{NW}(x, y_0)$ (for $1 \leq x \leq n$) in the same Row y_0. Define the *ending point* of $\mathcal{E}_{NW}(x, y_0)$ as the last node that is on the shortest path from s to the sink $t(x, y_0)$. We have the following lemma.

Lemma 2.1. *If the ending point of $\mathcal{E}_{NW}(x, y_0)$ is $v_{(x',y_0)}$ ($x' \leq x$), then the ending point of $\mathcal{E}_{NW}(x+1, y_0)$ is either $v_{(x',y_0)}$ or $v_{(x+1,y_0)}$.*

The proof of the lemma is in the Supplementary Material. Based on Lemma 2.1, we can compute all $\mathcal{E}_{NW}(x, y_0)$, $(1 \leq x \leq n)$ for Row y_0 in $O(N^{0.5})$ time from the computed shortest path tree. Hence, all $\mathcal{E}_{NW}(x, y)$ for $1 \leq x \leq n$ and $1 \leq y \leq n$ can be computed in $O(N)$time. Similarly, one can compute the table $\mathcal{E}_{NE}(\cdot, \cdot)$, $\mathcal{E}_{SW}(\cdot, \cdot)$ and $\mathcal{E}_{SE}(\cdot, \cdot)$ in $O(N)$ time.

At this point, given a center rectangle M specified by its two diagonal corner points, (x_1, y_1) and (x_2, y_2), we can compute an optimal five-parts labeling with minimized energy $\mathcal{E}_f(x_1, y_1; x_2, y_2)$ in $O(1)$ after an $O(N)$ preprocessing. That is,

$$\begin{aligned}\mathcal{E}_f(x_1, y_1; x_2, y_2) = \quad & \textstyle\sum_{g \in \{N,W,M,E,S\}} \mathcal{E}_g + \mathcal{E}_{NW}(x_1 - 1, y_1 - 1) + \mathcal{E}_{NE}(x_2 + 1, y_1 - 1) \\ & + \mathcal{E}_{SW}(x_1 - 1, y_2 + 1) + \mathcal{E}_{SE}(x_2 + 1, y_2 + 1) \qquad (2.12)\end{aligned}$$

Since there are $O(N^2)$ possible center rectangles, we are able to optimally solve the five-parts labeling problem in $O(N^2)$ time. During the preprocessing, we only need to compute $O(1)$ tables each with a size of $O(N)$. Thus, the space complexity is $O(N)$.

2.3.3 Speedup from $o(N^2)$ to $o(N^{1.5})$

The key idea of the speedup is: given two rows y_1 and y_2, $y_1 \leq y_2$, we return the best possible solution with its upper leftmost corner resides in Row y_1 and its lower rightmost corner resides in Row y_2, in $O(N^{0.5})$ time. In another word, find $\min_{x_1,x_2} \mathcal{E}_f(x_1, y_1, x_2, y_2)$ in $O(N^{0.5})$ time.

Applying Eqn. (2.12) results in $\mathcal{E}_f(x_1+1, y_1, x_2, y_2) - \mathcal{E}_f(x_1, y_1, x_2, y_2) = H(x_1, y_1; y_2)$. Note $H(x_1, y_1; y_2)$ is independent of x_2. This property of $H(\cdot, \cdot; \cdot)$ is crucial to the speedup (proof can be found in supplementary materials). According to definition of $H(\cdot, \cdot; \cdot)$, we have $\mathcal{E}_f(x_1, y_1, x_2, y_2) = \mathcal{E}_f(1, y_1, x_2, y_2) + \sum_{i=1}^{x_1-1} H(i, y_1; y_2)$. As a result,

$$\arg_{x_2} \min_{x_2 \geq x_1} \mathcal{E}_f(x_1, y_1, x_2, y_2) = \arg_{x_2} \min_{x_2 \geq x_1} \mathcal{E}_f(1, y_1, x_2, y_2) \tag{2.13}$$

for fixed y_1, y_2. In another word, we only need to compute $\min \mathcal{E}_f(1, y_1, \cdot, y_2)$ for $x_1 = 1$, and it could be used to compute $\min_{x_2 \geq x_1} \mathcal{E}_f(x_1, y_1, \cdot, y_2)$ for $x_1 \neq 1$. Define the following running min and running sum:

$$rMin_{(y_1,y_2)}(x) = \min_{i \geq x} \mathcal{E}_f(1, y_1, i, y_2) \tag{2.14}$$

$$hMin_{(y_1,y_2)}(x) = \sum_{i=1}^{x-1} H(i, y_1; y_2) \tag{2.15}$$

For fixed y_1 and y_2, $rMin_{(y_1,y_2)}(\cdot)$ and $hMin_{(y_1,y_2)}(\cdot)$ can be computed within $O(N^{0.5})$ time. Let x_2^* be the optimal x_2 that achieves optimal energy $\mathcal{E}_f(x_1, y_1, x_2^*, y_2)$ for fixed x_1, y_1, y_2, then

$$\mathcal{E}_f(x_1, y_1, x_2^*, y_2) = rMin_{(y_1,y_2)}(x_1) + hMin_{(y_1,y_2)}(x_1) \tag{2.16}$$

Note for fixed x_1, y_1, y_2 this only takes constant time, given that $rMin_{(y_1,y_2)}(\cdot)$ and $hMin_{(y_1,y_2)}(\cdot)$ have been computed.

This accomplish our goal of finding optimal solution for fixed Row y_1 and y_2 in $O(N^{0.5})$ time. Directly repeating this process for all $1 \leq y_1 \leq y_2 \leq n$ results in a $O(N^{1.5})$ algorithm. Note $rMin_{(y_1,y_2)}$ and $hMin_{(y_1,y_2)}$ does not need to be remembered for different y_1, y_2, so memory consumption for them is just $O(N^{0.5})$.

Theorem 2.2. *Given an image of $N = n \times n$ pixels, the five-parts labeling problem can be solved in $O(N^{1.5})$ time and $O(N)$ space.*

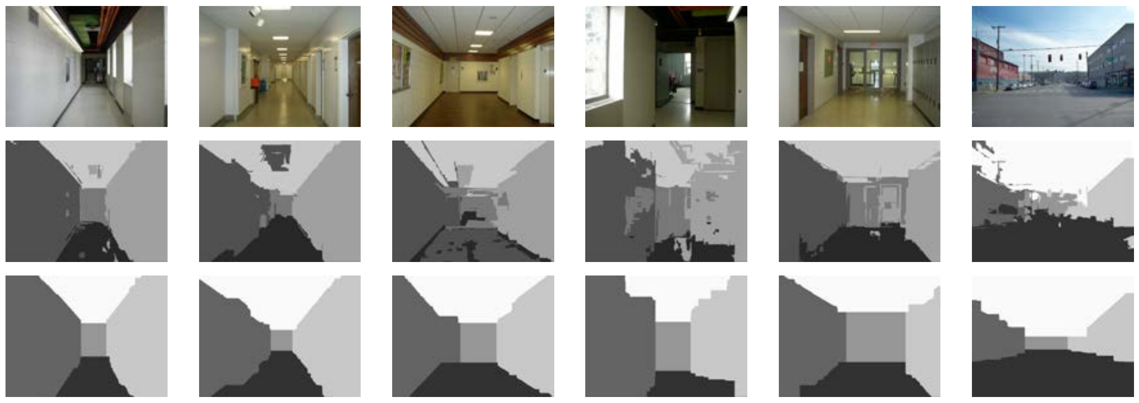


Figure 2.3: Some labeling results. Top row: original images; second row: SVM classifier results using data term only; last row: our results.

2.4 Experiment–Geometric Class Labeling

We used 300 indoor images and 42 outdoor images, which are the same as the test images used in [57]. All indoor images are 640*480. But outdoor images have various sizes.

2.4.1 Cost images

We used the same data term costs as in [57]. First, the images are partitioned into "superpixels", which are homogeneous regions within each region and heterogeneous between different regions, using the algorithm by Felzenszwalb et al. [28]. Similar to Hoeim et al.'s method [37], an SVM classifier is then trained with a wide variety of selected features, such like location, color, texture, geometry, and edges. Finally, a probability for each "superpixel" to be assigned a label $l \in \mathcal{L} = \{L, R, T, B, C\}$ is computed. All pixels within this "superpixel" are assigned a cost according to the probability of the "superpixel" it belongs to. This completes the data term generation.

The smoothness term is generated simply using Sobel operator along the horizontal and vertical directions.

2.4.2 Results

Example results are shown in Fig. 2.3.

Define the accuracy rate as the ratio of the number of correctly labeled pixels over the total number of pixels. The performance on the accuracy of our algorithm and the order-preserving moves method is shown in Table.2.2.

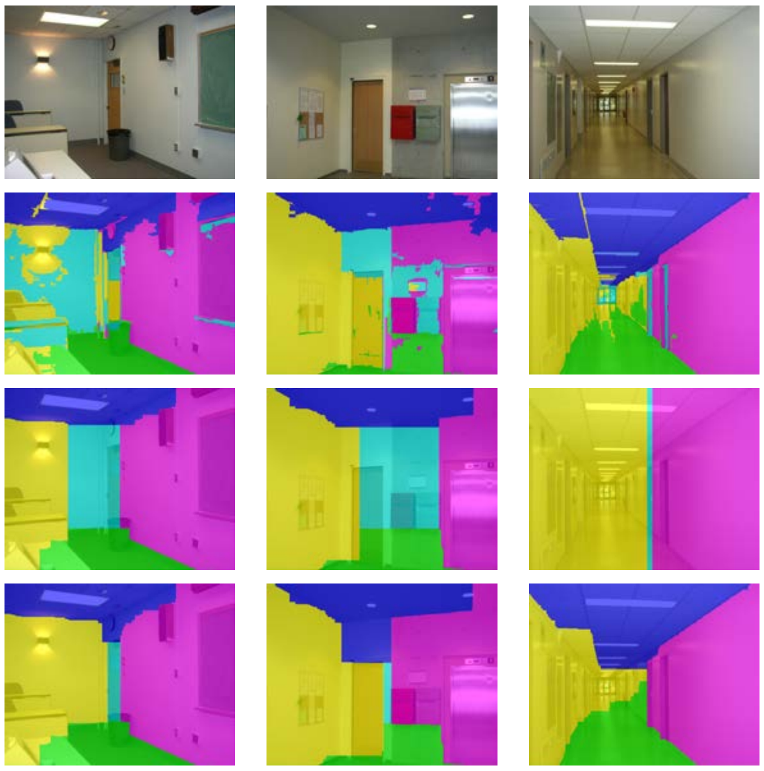


Figure 2.4: Example images on which our algorithm output quite different labeling results from the order-preserving moves. First row: original image; second row: SVM classifier results, i.e. results using only data term; third row: results by OPM; fourth row: results by our algorithm.

Table 2.2: Average accuracy rate (%).

Image sets	OPM	Our alg.
Indoor images	84.9 $\pm$ 14.9	85.1 $\pm$ 14.5
Outdoor images	85.7 $\pm$ 7.0	85.7 $\pm$ 6.9

OPM: the order-preserving moves method.

Our algorithm does not show significant improvement in the accuracy rate. The difference of minimized energy is 0.10% for indoor images and 0.16% for outdoor images on average for both methods, although our method always obtained an energy no worse than the order-preserving moves method. The marginal difference indicates that the order-preserving moves works pretty well in practice.

Although for most of the test cases, there are little difference between the order-preserving moves and our algorithm, we do observe significant difference on some cases, as shown in Fig.2.4. Our algorithm captures the door in the image in the first column (last row), and the rectangular space between the door and the box in the second column. The cost image for the image in the third column is poor, from which it is very difficult to distinguish the "left" region from the "center" region. However, our algorithm still can produce a reasonable labeling, while the order-preserving moves is trapped into a local minima with a long execution time of 244.18 seconds.

Table.2.3 shows the average execution time of the order-preserving moves and our algorithm. Our algorithm outperforms the order-preserving moves significantly while guaranteeing the global optimality. Note that our execution time is much better than that (90s) reported in [88] by Strekalovskiy and Cremers despite their use of CUDA for parallel implementation.

The execution time reported in [30], for tiered scene labeling, is 9.4 seconds on images approximately 300×250; while the execution time of our algorithm is 2.2 seconds on 320×240 images. Note the $O(N^{1.5})$ memory consumption of [30] might

Table 2.3: Average execution time (s).

Image sets	OPM	Our alg.
Indoor images	26.6±22.6	17.1±0.1
27 outdoor images with size of 640 × 480	20.6±12.5	16.4± 0.3
Overall outdoor images*	20.2	14.1

*: The sizes of overall outdoor images vary. Thus, no standard deviations are reported.

Table 2.4: Average execution time comparison (s).

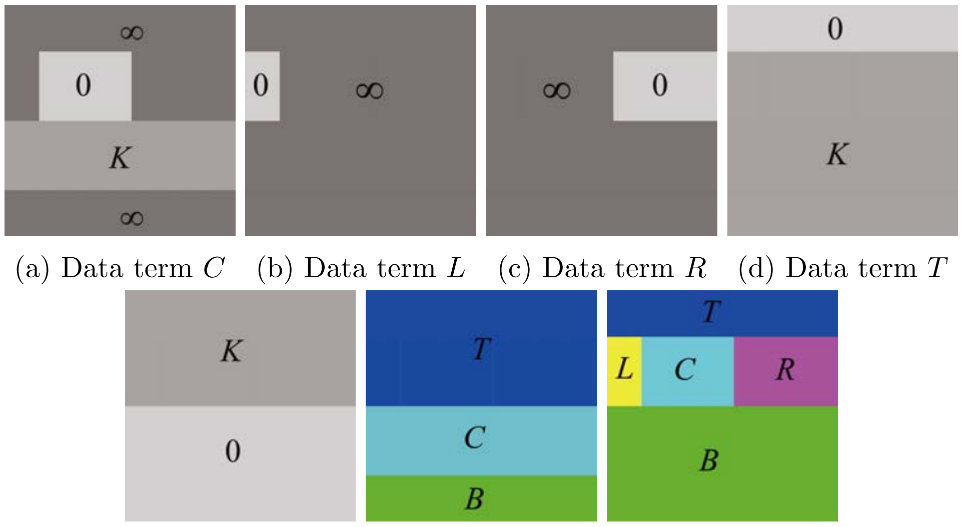
Methods	No noise	$\sigma = 0.17$	$\sigma = 0.29$	$\sigma = 0.58$
OPM	20.2	43.8	40.4	81.4
Our alg.	14.1	14.7	14.5	15.0

make it problematic to process large images, which is overcome by our algorithm. In addition, our algorithm can be easily parallelized for GPU, which may bring more significant speedups. This will be discussed in Sec. 2.5

Moreover, the running time of our method only depends on the image size. The standard deviation of the execution times over hundreds of test images with the same size is almost 0 for our method, while it is comparable to the mean execution time for the order-preserving moves method, as in Table. 2.3. By intentionally adding Gaussian noise to cost images, we observe little effect on the execution time of our algorithm, while a big deterioration is observed for the order-preserving moves method, as shown in Table 2.4 and Table 2.5. The mean value of the Gaussian noise is 0 and σ is normalized with respect to the maximum intensity value of the cost image.

Table 2.5: Max execution time comparison (s).

Methods	No noise	$\sigma = 0.17$	$\sigma = 0.29$	$\sigma = 0.58$
OPM	136.4	396.4	267.7	1408.8
Our alg.	33.8	35.2	35.2	41.1



(a) Data term C (b) Data term L (c) Data term R (d) Data term T

(e) Data term B (f) Local minima (g) Optimum

Figure 2.5: Illustrating the lack of global optimality for the order-preserving moves method. All smoothness penalties are 0 in this example. The energy of the global optimum is 0. While the energy of the local minima is a multiple of K. Note K can be arbitrarily large, which will make the local minima arbitrarily far away from the optimum.

2.5 Discussion

2.5.1 Global optimality

Global optimization is important for the labeling problems. Although the order-preserving moves method works well for test image datasets we used, it may get trapped in a local minima very far away from the global optimal solution, and fail to find an acceptable solution.

Consider the given costs for each label shown in Fig. 2.5, and all smoothness penalties are set to be 0. Start from an initial labeling with all pixels labeled as 'C' [56]. A horizontal move results in a labeling of an energy of ∞ , since a vertical strip across the whole image must be labeled as 'C' in this horizontal move. Hence, only a vertical move is possible to return a finite energy by labeling the horizontal strip in which all pixels have a cost of K for label 'C'. Unfortunately, the order-preserving moves method gets stuck here. Any further order-preserving move will results in ∞ energy.

However, the energy of the global optimal solution is 0. Note the value of K could be arbitrarily large, which indicates that even this strong order-preserving moves method gets trapped in a local minima arbitrarily far from the optimal solution.

2.5.2 Parallelization of the algorithm

The most time-consuming part of this algorithm is the optimal center rectangle searching process. In a typical running on a 640×480 image, this process takes about 16 seconds, while the average total execution time for such an images is just about 17 seconds.

However, this process is highly parallelizable. We can view each row pair of y_1 and y_2 as a unit for parallelization. As indicated in Sec. 2.5.1, the computations between different y_1, y_2 row pairs are totally independent of each other. This makes our algorithm straightforward to parallelize on multi-core CPUs and high-end GPUs. There are $O(N)$ different y_1, y_2 pairs in total. This number of threads should be able to saturate the current high-end CPUs, which only have hundreds of cores.

2.5.3 Extension to 8-connectivity

Our method can be easily extended to the 8-neighborhood setting. Note that the algorithm in [30] is at least nontrivial to do the extension.

It introduces additional smoothness terms using the 8-connectivity. The smoothness penalty between the center region and the other four non-corner regions is easy to handle. We next show how to handle the smoothness penalty for the corner regions.

We again illustrate our idea on the NW region. In Fig. 2.6, the red and green dotted double-arrows show the smoothness penalties on the diagonally neighboring pixels we need to enforce into our shortest path model. We basically want to distribute those penalties to the edges on the boundary path. Note that the smoothness penalties indicated by the green double-arrows correspond one-by-one to the edges on the boundary path as follows. For the downward edge from $v_{(x,y)}$ to $v_{(x,y+1)}$, we add an additional cost of $V_{p(x,y+1),q(x+1,y)}(f_p = L, f_q = T)$ to the edge. While for the rightward edge from $v_{(x,y)}$ to $v_{(x+1,y)}$, an additional cost of $V_{p(x+1,y),q(x+2,y-1)}(f_p = L, f_q = T)$ is added to the edge.

Similarly, smoothness penalties indicated by the red double-arrows can be accounted by adding following costs: for the downward edge from $v_{(x,y)}$ to $v_{(x,y+1)}$, add $V_{p(x,y),q(x+1,y+1)}(f_p = L, f_q = T)$, and for the rightward edge from $v_{(x,y)}$ to $v_{(x+1,y)}$, add $V_{p(x+1,y),q(x,y-1)}(f_p = L, f_q = T)$. The only problem is that the sum of the two brown edges incident at $(\bar{x}, \bar{y})$ in Fig. 2.6 overestimates by an amount of $V_{p(\bar{x},\bar{y}-1),q(\bar{x}+1,\bar{y})}(f_p = L, f_q = T) + V_{p(\bar{x}+1,\bar{y}),q(\bar{x},\bar{y}-1)}(f_p = L, f_q = T)$. To solve this problem, we introduce a diagonal edge $e(v_{(\bar{x},\bar{y}-1)}, v_{(\bar{x}+1,\bar{y})})$, whose cost equals

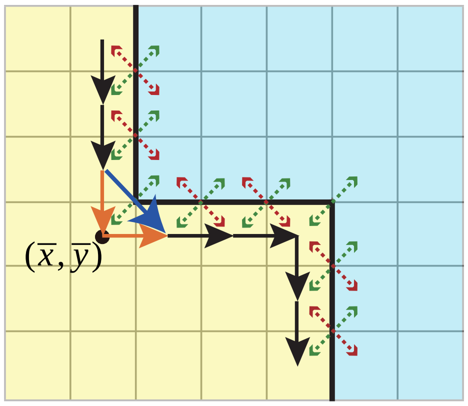


Figure 2.6: The green and red double-arrows indicate the additional smoothness penalties we need to enforce when extended to the 8-connectivity.

$c_e(v_{(\bar{x},\bar{y}-1)}, v_{(\bar{x},\bar{y})}) + c_e(v_{(\bar{x},\bar{y})}, v_{(\bar{x}+1,\bar{y})}) - V_{p(\bar{x},\bar{y}-1),q(\bar{x}+1,\bar{y})}(f_p = L, f_q = T) - V_{p(\bar{x}+1,\bar{y}),q(\bar{x},\bar{y}-1)}(f_p = L, f_q = T)$. If we assume all smoothness penalties are nonnegative, then this edge is always preferable than the “detour” of the two brown edges.

2.6 Conclusion

We present an algorithm optimally solving the five-parts labeling problem, which, to the best of our knowledge, is the first algorithm that guarantees globally optimal solution to that labeling problem with linear space complexity. The theoretical running time is $O(N^{1.5})$, with N being the number of pixels in the image. In practice, it runs much faster than the method reported in [56]. Moreover, it can easily be parallelized for GPU, and it is extensible to the 8-neighborhood setting without affecting the theoretical running time.

CHAPTER 3 ROBUST INTERACTIVE SEGMENTATION

Scribbles in scribble-based interactive segmentation such as graph-cut are usually assumed to be perfectly accurate, i.e., foreground scribble pixels will never be segmented as background in the final segmentation. However, it can be hard to draw perfectly accurate scribbles, especially on fine structures of the image or on mobile touch-screen devices. We propose a novel ratio energy function that tolerates errors in the user input while encouraging maximum use of the user input information. More specifically, the ratio energy aims to minimize the graph-cut energy while maximizing the user input respected in the segmentation. The ratio energy function can be exactly optimized using an efficient iterated graph cut algorithm. The robustness of the proposed method is validated on the GrabCut dataset using both synthetic scribbles and manual scribbles. The experimental results show that the proposed algorithm is robust to the errors in the user input and preserves the "anchoring" capability of the user input.

3.1 Introduction

Image segmentation/object selection is widely used in image processing. While fully-automatic segmentation methods can provide satisfactory result in some cases, human interaction is needed to produce high quality segmentation in more challenging images. Among various interactive approaches, two of the most popular ones are the boundary-based segmentation[62] and the scribble-based segmentation[75, 89, 4, 54, 55, 53]. The boundary-based interactive segmentation such as intelligent scissors [62] requires the user to trace the whole boundary of the object, which is usually time-consuming and tedious for users. Scribble-based interactive segmentation, on the other hand, is based on a number of foreground and optionally background scribbles. The algorithm will automatically label the pixels as either foreground or background based on the information such as location, color, texture, etc. provided by the scribbles.

Classical scribble-based interactive segmentation takes the scribbles as hard constraints, i.e., all foreground and background scribbles are guaranteed to be foreground and background, respectively, in the segmentation results. This requires the scribbles to be highly accurate, otherwise the segmentation gets compromised. This requirement can be hardly met on the mobile touch-screen devices, which has increasingly found wide applications. Even on a big screen with a mouse, it is hard to draw perfectly accurate scribbles on challenging images with fine structures, such as a thin bush stem or legs of a table. The scribble-based approaches have been widely used

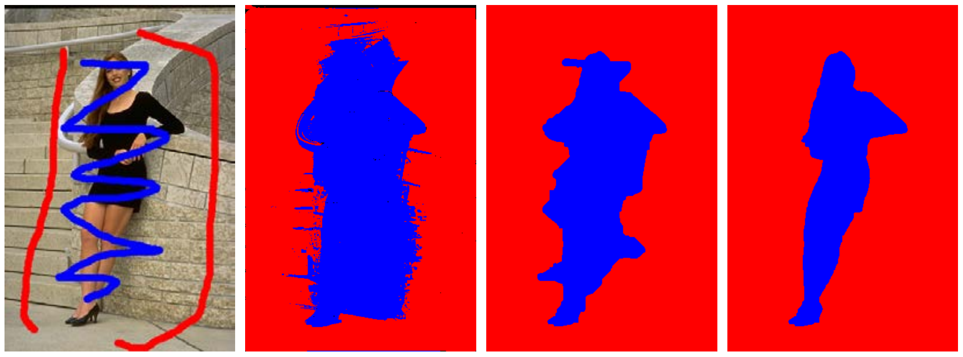

(a) scribbles (b) Subr's [89] (c) graph-cut (d) proposed

Figure 3.1: The proposed method tolerates errors in user-specified scribbles and is better at segmenting fine structures in an image.

in image editing [4, 54] and image segmentation [13]. In the interactive image editing, users first specify sparse scribbles and the corresponding edits to be performed on each scribble, such as tone, color and/or material changes. These edits are then propagated to all the other pixels in the image with a modulated “editing strength”, which can be seen as a certain soft segmentation.

An et. al. [4] formalizes the image editing problem as a quadratic optimization problem based on the pixel affinity matrix in the pixel appearance space. The affinity matrix is defined on all pairs of pixels. A low-rank stochastic approximation is applied to obtain an approximate solution. This method does not build an explicit appearance from the whole set of scribbles, instead it builds an implicit model by propagating information between all pairs of pixels. This method relies on that information propagation among all pairs of pixels to tolerate user input errors. Li et. al.[54] observed that the optimization formulation in An et. al.’s work [4] essentially is a smooth function with a sparse set of constraints. Based on this observation, they approximately decomposed the given editing strength on the scribble pixels into a series of radial-based editing functions. The editing strength on all the other pixels are then interpolated using these radial-based editing functions for their appearance representation. This method runs extremely fast. However, the quality of the results highly depends on the representation capability of the series of radial-based editing functions for the user’s intentions. In addition, both An et. al.’s and Li et. al.’s methods only produce a continuous “editing strength” map, which reflects how similar each pixel is to the foreground seed, instead of binary segmentation.

Various methods have been proposed to alleviate the problem of user input errors in the binary image segmentation. Liu et. al.'s method [55] allows the user to override the erroneous scribbles by specifying new scribbles, which overlap partially with the inaccurate old scribbles. The new scribbles are then enforced as new hard constraints while the old scribbles are regarded as soft constraints for the segmentation. Clearly, this method still highly replies on the accuracy of the new scribbles. Sener et. al. [75] developed an error-tolerant interactive segmentation method using dynamic and iterated graph-cuts. Essentially, the method removes the inaccurate scribble pixels from being used as seeds with some heuristics in the preprocessing step. Subr et. al. [89] make use of a dense conditional random field (CRF) to infer the segmentation from possibly inaccurate scribbles. The dense CRF model contains a simple unary term and a fully connected CRF among all pairs of pixels in the image. To solve the dense CRF, they embedded pixels in a low-dimensional Euclidean space with a metric that approximates the desired high-dimensional affinity function.

We introduce a novel ratio-form energy function which consists of a graph-cut energy term to utilize both region and boundary information from the input image, and a user-scribble utility term to encourage the user scribbles to be respected. Essentially, optimizing the ratio energy function is equivalent to minimizing the graph-cut energy while at the same time respecting the user input as much as possible. The user scribbles are enforced as a soft constraint instead of a hard constraint, which allows the proposed method to tolerate user input errors. In contrast to the methods which deal with user-input errors using heuristics such as Sener et. al.'s work [75], a global optimization framework is utilized to handle the user input errors. Comparing to the fully connected CRF method [89], our method enjoys the sparsity of the constructed graph from the neighborhood setting, as in the graph cut method [13]. Our experiments demonstrated that the energy function in the proposed method can be optimized efficiently and can produce spatially coherent segmentations.

3.2 Methods

We formalize the segmentation problem as the optimization of a ratio energy in which the numerator is the graph-cut energy and the denominator is a utility function which increases as more user input is respected.

3.2.1 Energy formulation

The energy to minimize is

$$E(\mathbf{x}) = \frac{E_{gc}(\mathbf{x})}{M + U(\mathbf{x})}, \tag{3.1}$$

in which $\mathbf{x} \in \mathcal{L}^{\mathcal{P}}$ is the labeling of all the pixels $\mathcal{P}$ from a binary set of available labelings $\mathcal{L} = \{`ob', `bg'\}$.

$E_{gc}(\mathbf{x})$ is the graph-cut energy [13], which consists of a region term $D_p(x_p)$ and a boundary term $V_{pq}(x_p, x_q)$ (Eq.(3.2)). The region term measures how likely each pixel p belongs to object ('ob') or background ('bg'). Unlike classical graph-cut [13] which assigns infinite region term weight to seed pixels to ensure them as hard constraint, we assign region term weights to seed pixels just like any other non-seed pixels. Thus no hard constraint is enforced in Eq.(3.2). The boundary term

$V_{pq}(x_p, x_q)$ penalizes the discontinuity between the object and background, that is, $V_{pq}(x_p, x_q)$ is the penalty of assigning labels x_p and x_q to two neighboring pixels p and q according to the neighborhood setting $\mathcal{N}$. We use 8-neighborhood setting in this work. More precisely,

$$E_{gc}(\mathbf{x}) = \sum_{p \in \mathcal{P}} D_p(x_p) + \eta \sum_{(p,q) \in \mathcal{N}} V_{pq}(x_p, x_q), \tag{3.2}$$

where, η is a balancing constant between the region term and the boundary term. $V_{pq} = 0$ if $x_p = x_q$, and $V_{pq} > 0$ if $x_p \neq x_q$.

$U(\mathbf{x})$ is a nonnegative utility function which increases as more user input information is respected in the segmentation result. Assume that $\mathcal{S}_F$ and $\mathcal{S}_B$ are the sets of pixels included in the foreground and background scribbles, respectively. Denote by $d_b(x_p)$ the distance between pixel p and the nearest scribble boundary. Two user input utility functions that we use in this work are defined, as follows.

$$U_1(x_p) = \begin{cases} 1 & \text{if } p \in \mathcal{S}_F \text{ and } x_p = \text{`}ob\text{'} \\ 1 & \text{if } p \in \mathcal{S}_B \text{ and } x_p = \text{`}bg\text{'} \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

$$U_2(x_p) = \begin{cases} d_b(x_p)^2 & \text{if } p \in \mathcal{S}_F \text{ and } x_p = \text{`}ob\text{'} \\ d_b(x_p)^2 & \text{if } p \in \mathcal{S}_B \text{ and } x_p = \text{`}bg\text{'} \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

The first utility function $U_1(x_p)$ simply counts the number of the user scribble pixels that are respected in the final segmentation. This utility function is used when large amount of user input error is expected (such as in the image editing application), since in this case we usually do not know which portion of seed is more important than the other portions. The rational behind the design of the utility function $U_2(x_p)$ is that a user is more likely to draw a scribble whose centerline is correct while the boundary of the scribble has more bias to be wrong (such as in the image segmentation application). This utility function is used when we expect the user to make small mistakes that mostly happen around the scribble boundary. For example, a careful user may rarely make any mistake except at drawing scribbles on very thin structures such as table legs. More sophisticated $U(\mathbf{x})$ can be designed as long as it increases while more user input information is respected, and is nonnegative for all configurations due to optimization consideration.

M is a constant that controls how “flexible” the method is with respect to user-specified scribbles. The larger M is, the more scribble pixels are likely to be allowed foreground-background swap in the segmentation. To see this, imagine the extreme case in which $M \gg U(\mathbf{x})$, then essentially the denominator of Eq.(3.1) is constant M and we are just optimizing the numerator, which is the graph-cut energy.

Note the seed pixels are regarded equivalent to those non-seed pixels in numerator. In this case, seed pixels have no special roles at all in the energy function, and cannot "anchor" segmentation anymore. In our experiment, we set M to be a multiple of the maximum possible utility function value.

$$M = \alpha \sum_{p \in \mathcal{P}} [U(x_p = `ob') + U(x_p = `bg')] \tag{3.5}$$

Note that by minimizing the energy function in Eq.(3.1), we attempt to minimize the graph-cut energy $E_{gc}(\mathbf{x})$ while maximizing the respect to the user input scribbles. The optimization process is to find out the "best" set of scribble pixels such that the swap of their foreground-background labels enables the maximum reduction of the energy $E_{gc}(\mathbf{x})$ (i.e., those erroneously identified as foreground or background scribbles), thus achieving our goal of error-tolerance.

3.2.2 Optimization of ratio energy

We use Newton's method for ratio optimization [32] to minimize the ratio energy function $R(\mathbf{x}) = P(\mathbf{x})/Q(\mathbf{x})$, in which functions $P, Q : \mathcal{X} \to \mathbb{R}, \mathcal{X} = 2^{\mathcal{V}}$ and $Q(\mathbf{x}) \geq 0, \forall \mathbf{x} \in \mathcal{X}$. The main idea is to iteratively minimize a related linear function instead of the ratio function until convergence. The optimal solution of the ratio energy is given by the optimal solution of the linear function after convergence. The related linear function, which is called λ-*function*, is defined as

$$E_R^{\lambda}(\mathbf{x}) = P(\mathbf{x}) - \lambda Q(\mathbf{x}) \tag{3.6}$$

More formally, the Newton's method for ratio optimization is defined in Alg.3.1. Theorem.3.1 claims the correctness of the algorithm. In our experiment, Alg.3.1 always converged in a few iterations (less than 5 iterations).

Algorithm 3.1 Newton's algorithm for ratio optimization

```
Input: Min-ratio problem $\min_{\mathbf{x}\in\mathcal{X}} R(\mathbf{x})$
Output: Opt sln $\mathbf{x}^*$ for $\lambda^* = \min_{\mathbf{x}\in\mathcal{X}} R(\mathbf{x})$
Select some $\mathbf{x}_0 \in \mathcal{X}$. $\lambda_1 \leftarrow R(\mathbf{x}_0)$. $k \leftarrow 1$.
while do
    Compute $\mathbf{x}_k = \text{argmin}_{\mathbf{x}\in\mathcal{X}} E(\mathbf{x}; \lambda_k)$
    $z(\lambda_k) \leftarrow E(\mathbf{x}_k; \lambda_k)$
    if $z(\lambda_k) = 0$ then
        return $\mathbf{x}_k$ as optimal solution, $\lambda^* \leftarrow \lambda_k$
    end
    $\lambda_{k+1} \leftarrow R(\mathbf{x}_k)$
    $k \leftarrow k + 1$
end
```

Theorem 3.1. *[32] Algorithm 3.1 outputs an optimal solution to minimizing* $R(\mathbf{x}) = P(\mathbf{x})/Q(\mathbf{x})$*. In addition, the generated sequence* $\{\lambda^k\}$ *is strictly decreasing, i.e.,* $\lambda^{k+1} < \lambda^k$*.*

For Alg.3.1 to work properly, the λ-function in Eq.(3.6) needs to be efficiently optimized. Note that in Eq.(3.1), $P(\mathbf{x}) = E_{gc}(\mathbf{x})$ is the graph-cut energy consisting of a unary term and a pairwise term, and $Q(\mathbf{x}) = M + U(\mathbf{x})$ is a unary function of $\mathbf{x}$. Thus, the λ-function, as shown in Eq.(3.7), consists of only unary terms and a pairwise term, which can be optimized by the graph-cut method. Since M is a constant, the removal of M in the linear form of the optimization problem does not affect the final solution.

$$E^{\lambda}(\mathbf{x}) = E_{gc}(\mathbf{x}) - \lambda U(\mathbf{x}) \tag{3.7}$$

As a result, we will use graph-cut to optimize λ-function in each iteration. Instead of computing a new max-flow from scratch, we used the dynamic graph-cut implementation [47] to reuse results from previous iteration as an initialization. Note we do not need to modify the weights of all arcs in the graph for each iteration. Since $U(\mathbf{x})$ is nonzero only for those user scribble pixels, thus, only the weights of those arcs associated with scribble pixels need to be updated in each iteration, and there are no weight changes for all the other arcs from one iteration to another. Thus, the overhead of updating arc weights are pretty light during the whole optimization process, and the algorithm runs efficiently in practice. Fig.3.2 shows the graph construction.

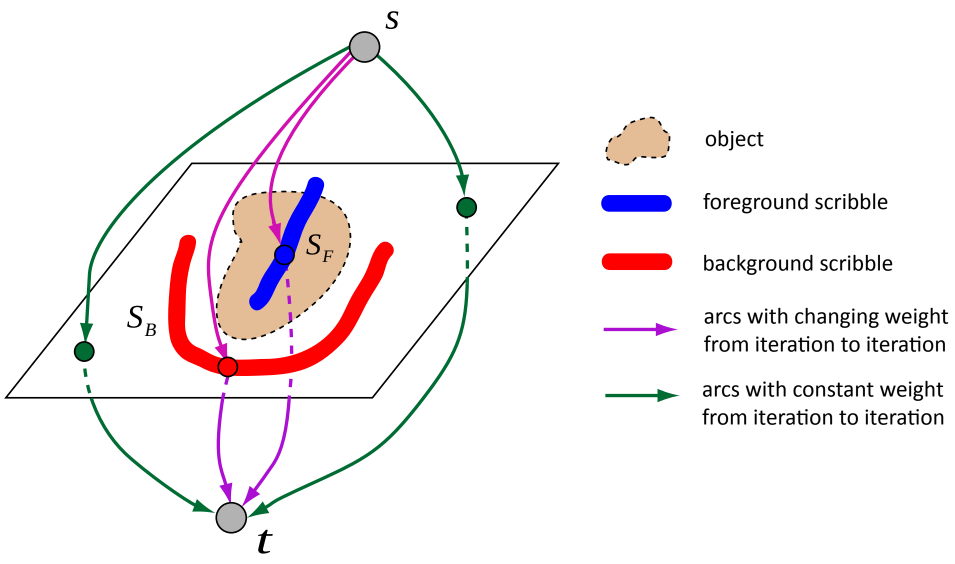

s
t
S_F
S_B
object
foreground scribble
background scribble
arcs with changing weight from iteration to iteration
arcs with constant weight from iteration to iteration

Figure 3.2: Only arcs connected to scribble pixels need to update their weights in each iteration. All other arcs have constant weights during all iterations.

3.3 Experiments

3.3.1 Experiment settings

To validate our algorithm, we use GrabCut dataset [73], which contains 50 images including a variety of objects such as person, car, goat, etc. Ground truths for all 50 images are available.

In this preliminary work, we used some simple cost function designs for each term in the energy function to achieve our goal of proof-of-the-concept. More comprehensive cost designs will likely improve the performance of our proposed method. The region term is generated by building two Gaussian Mixture Models (GMM) for foreground and background, respectively, in the Lab color space. These two GMM models are then applied on all pixels to generate the region term. The boundary term is obtained by computing the gradients on a smoothed image with the bilateral image filtering. For the weighting coefficient η between the region term and the boundary term in the graph-cut energy in Eq.(3.2), we set it to be 1 for all of our experiments. For those experiments in which the number of the erroneous scribble pixels is expected to be small (Sec.3.3.3), we use the utility function Eq.(3.4). Otherwise, we use the utility function Eq.(3.3). The coefficient α in Eq.(3.5) is set to be 1 for the experiments in Sec.3.3.3, to strongly enforce the user scribbles as anchor points. It's set to be 10 for those experiments in Sec.3.3.2 to tolerate large synthetic scribble errors, and is set to 100 for the experiments in Sec.3.3.4 to allow even larger and more spatially coherent scribble errors. We will discuss the issue of choosing appropriate α parameter in Sec.3.5.

Two metrics are used to measure the segmentation accuracy: the labeling accuracy and the Dice similarity coefficient. The metric labeling accuracy is defined as percentage of pixels correctly labeled in the final segmentation. More precisely, assume that TP, TN, FP, and FN represent the number of pixels that are true positive, true negative, false positive, and false negative, respectively, in the segmentation. Then the labeling accuracy $= \frac{TP+TN}{TP+TN+FP+FN}$. The second metric is the Dice Similarity Coefficient (DSC), which measures the overlap between two segmented object volume. Suppose we have two segmentations O_1 and O_2 for the object, the DSC between the two segmentations is defined as $\frac{2|O_1 \cap O_2|}{|O_1|+|O_2|}$, or equivalently $\frac{2TP}{2TP+FP+FN}$.

Compared to the labeling accuracy, the computation of DSC does not rely on TN, which means that DSC is not sensitive to large areas of background in the image. For an image with large background, a segmentation even assigning every pixel to be background will have high labeling accuracy. However, DSC is able to tell that there is zero overlapping between the segmented object and the ground truth in this case.

To demonstrate our algorithm's ability to handle user input errors (i.e., the user scribble pixels with wrong foreground/background labels), we compare our segmentation result to the classical graph-cut method [13] which regards the seeds (i.e., the user input scribbles) as a hard constraint. We also compare the proposed method to Subr et. al.'s approach [89], which also tolerates the user input errors using an optimization framework. The author's publicly available code is used in our comparison. There are multiple tunable parameters in their implementation, an oracle is used to test all possible combinations of the parameters to find the one that results in best accuracy. The resulting set of parameters is used in the comparison.

3.3.2 Experiment using synthetic scribbles

To quantitatively measure how robust the proposed algorithm is provided different degrees of user input errors, we follow the procedure used in Subr et. al.'s work [89]. First 50 foreground pixels and 50 background pixels are randomly selected based on ground truth. They are assigned as foreground or background scribbles, respectively. Then an error-zone for each image is defined as background pixels that are less than a distance D from the foreground, in which D is defined as 5% of the image diagonal (Fig.3.3b). We randomly select 0 to 50 pixels in the error zone and assign them as foreground scribbles to simulate different degrees of user input errors.

Note in our experiment, the randomly selected points are dilated by a radius of 5 pixels before they are used as scribbles, in contrast to the isolated pixels used in [89]. Because isolated point cannot effectively anchor segmentation in the graph-cut method. Moreover, our manual scribble in the next experiment has a width of 10 pixels. Dilating the randomly selected pixels by a radius of 5 will result in a small circle that's similar to a point scribble drawn manually (Fig.3.3c). We randomly select 0, 5, 10, 20, 30, 40, 50 erroneous sample pixels from error zone to simulate the error percentage of 0%, 10%, 20%, 40%, 60%, 80%, 100% in the user input. Fig.3.3 shows one set of synthetic scribbles which contains 20 erroneous samples from the error zone.

The performance of each method is shown in Fig.3.4. We can observe that

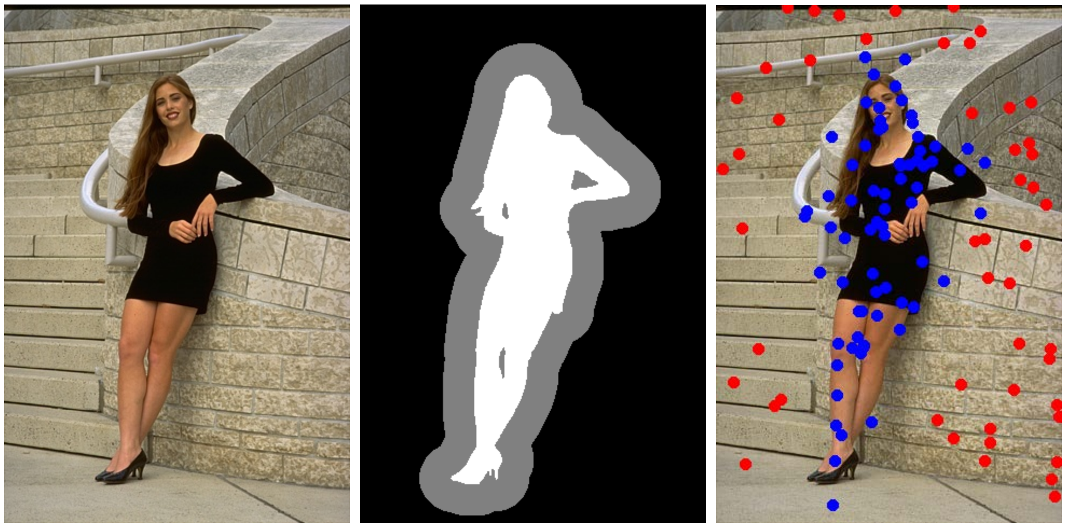

(a) input (b) error zone (c) synthetic scribbles

Figure 3.3: Error zone mask and a set of synthetic scribbles. The foreground scribbles consists of 50 foreground pixels and 20 error zone pixels. The background scribbles consists of 50 background pixels.

all methods perform quite well when no error is in user input. However, as more and more user input errors are added in the scribbles, the performances of graph-cut and Subr's method [89] get compromised quickly, while our method stays performing pretty well.

The reason that Subr's method's performance is worse than the graph-cut based approaches could be due to the use of a fully-connected CRF model, instead of a sparse MRF model as used in the graph-cut based approaches. Thus, it generates a less coherent segmentation since it can easily propagate (wrong) information to remote pixels (see the third row in Fig.3.6). In fact, as more and more errors are added to the user input, Subr's method can quickly propagate those erroneous information to remote pixels instead of stopping the error in a small local region.

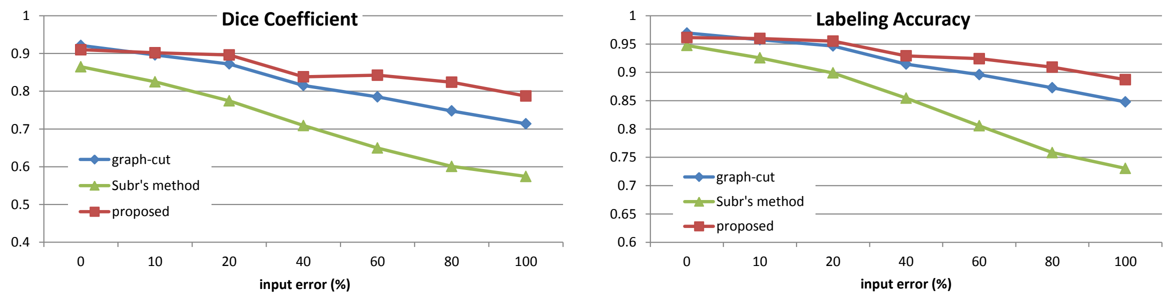


Figure 3.4: DSCs and the labeling accuracies of Subr's error-tolerant segmentation method, the graph-cut method, and the proposed robust segmentation method, given different percentage of errors in *randomly-generated synthetic* user input.

3.3.3 Experiment aiming for high accuracy

The scribbles drawn by users aiming high segmentation accuracy may not make as much mistake as shown in Fig.3.1a. In this case, the user input error usually comes from drawing scribbles on fine structures such as the bush stem, vase handle, sheep legs shown in the second row of Fig.3.5. To validate our algorithm for this type of user input, a user was asked to draw scribbles on all 50 images in GrabCut dataset manually in a *natural* way, i.e., the user neither intentionally makes mistake, nor makes excessive efforts to accurately draw the scribbles. As a result, no scribble errors are found in 20 images out of 50. The remaining 30 images have scribble errors in different degrees. On average, each of those 30 image scribbles contains 1.5% errors, i.e., 1.5% of the foreground scribble pixels are actually background with respect to the ground true. The maximum scribble error is 8.4%.

The proposed robust segmentation (RS) method, the graph-cut (GC) method, and Subr's error-tolerant method [89] are used to segment those 20 images with *error-free* manual scribbles (reported as “RS/GC/Subr's(correct)” in Table.3.1), and those

30 images with *erroneous* manual scribbles ((reported as "RS/GC/Subr's(error)" in Table.3.1)). To gain better understanding about how the errors affect the performance of graph-cut, we use an oracle to correct the errors by removing erroneous foreground scribble pixels that are actually background. Then, the graph-cut method runs on those corrected scribbles and reported as "GC(err-corrected)" in Table.3.1. Note that this method is not really a fair reference. It can be excessively "accurate" because when the oracle removes the scribble pixels outside the object, it is actually drawing the perfect boundary locally using the ground truth.

Table.3.1 shows the performance of each method. When no error happens, both the proposed method and the graph-cut method achieve high accuracy. But when the user scribbles contain errors, the proposed method performs better than the graph-cut method due to its tolerance of user input errors. The difference in accuracy metrics does not seem very high, which is understandable because users only make small errors around object boundary. However, the boundary accuracy of the segmentation indeed is improved by using the proposed robust segmentation method. Fig.3.5 shows the improved boundary by using the proposed method.

In the first column of Fig.3.5, the stem segmented by the proposed method is much thinner than graph-cut, which is closer to the ground truth. The proposed method also generates more accurate vase handle in the second column, more accurate sheep and person legs in the third and forth column. Subr's method performs even worse than the graph-cut method. It usually does not generate a spatially coherent segmentation and propagates errors in the segmentation to remote regions from the

Table 3.1: Manual scribble experiment result.

method (scribble type)	DSC (%)	label accuracy (%)
Subr's [89] (correct)	92.88	96.77
GC (correct)	97.64	98.91
RS (correct)	**97.65**	**98.92**
Subr's (error)	82.68	92.23
GC (error)	95.19	98.44
RS (error)	**95.26**	**98.46**
GC (err-corrected)	95.72	98.61
Subr's (all)	86.76	94.05
GC (all)	96.17	98.63
RS (all)	**96.22**	**98.64**
GC(all, err-corrected)	96.49	98.73

erroneous scribbles, as shown in the third row in Fig.3.5.

For the 30 images with erroneous scribbles, the proposed robust segmentation result corrected 60.1% of the erroneous scribble pixels on average.

3.3.4 Illustrative results

Scribbles with large errors frequently happens on mobile touch-screen devices due to the use of less accurate pointing devices (such as fingers). In these experiments, the user scribbles contain much more errors. Illustrative results are shown in Fig.3.6. Subr's method can easily propagate foreground scribble errors to background due to its fully connected CRF formulation, as shown in the bear and the lady images. Another issues is that it does not generate spatially coherent segmentation for textured object, as shown in the grave tombstone and the sheep images. The graph-cut method uses the user scribbles as hard seeds, and thus is not able to correct errors

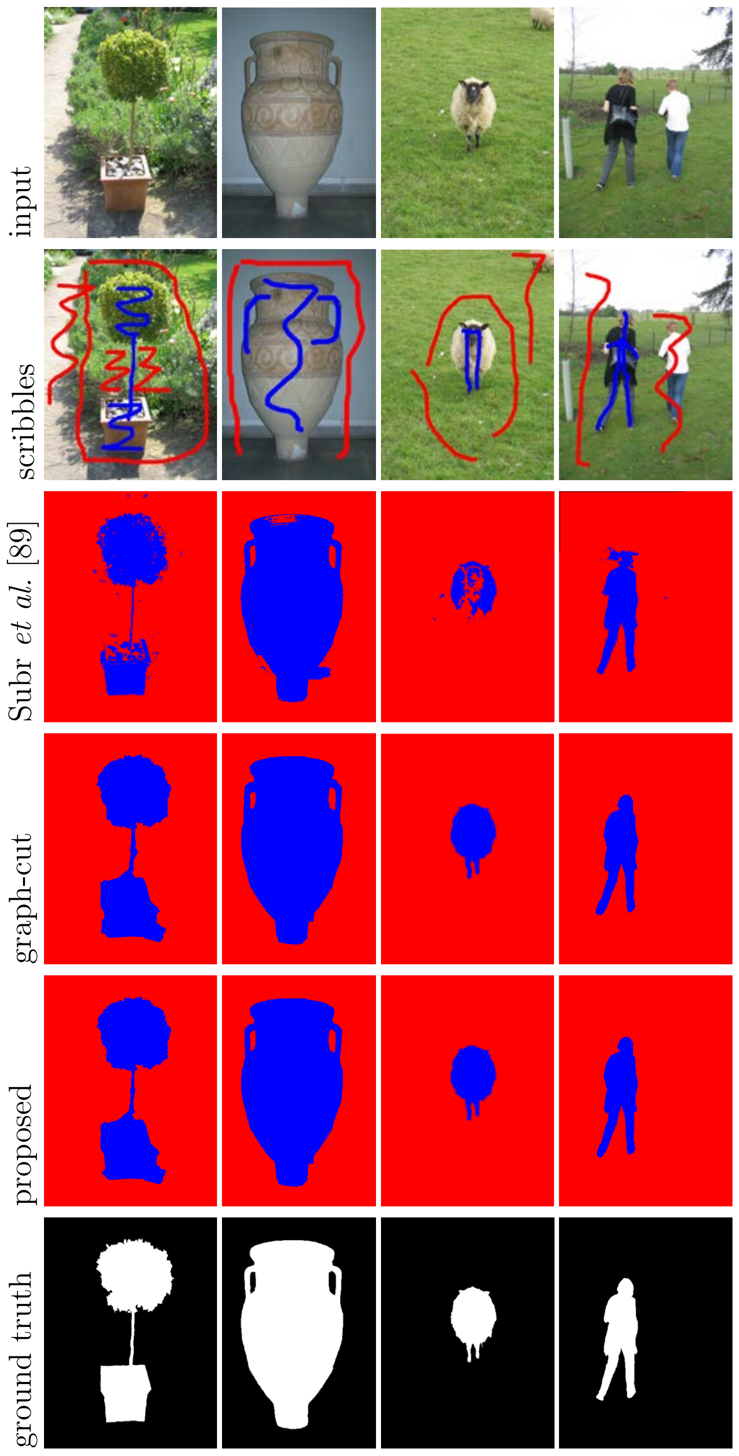


Figure 3.5: Improved boundary accuracy by using the proposed robust segmentation method. Note the proposed method improved boundary segmentation accuracy of the bush stem, vase handle, sheep legs and person's legs.

in user input. In contrast, the proposed method performs very well in this type of situation.

One interesting case is the sheep (last column in Fig.3.6). In order to separate the two legs shown in the image, the user adds one background scribble between them. Graph-cut simply follows the boundary of the added background scribble to separate the two legs. Our proposed method, however, is able to reject part of the erroneous background scribble pixels and segments the two legs with higher accuracy.

3.4 Proof Of Running Time

Although experimentally the algorithm always converge within several iterations, a theoretical running time proof provides a dependable worst case guarantee even for ill-posed cases. The Newton's method, which is also called Dinkelbach's algorithm [74, 25], can be proved to converge superlinearly for continuous optimization problems [74].

A more elaborated derivation of the superlinear convergence property is presented in [35]. In more details, first the denominator function value is strictly decreasing in every iteration (Lemma.3.2). Then an error function measured in terms of ratio objective function value is shown to reduce by a fraction of its previous iteration (Thm.3.3). What's even better, this fraction bound converges to zero as more iterations are run. This means the convergence rate increases as we get closer to the optimal solution, in contrast to many algorithms' slower convergence rate as they get close to the minimum. This fractional error reduction behavior leads to a

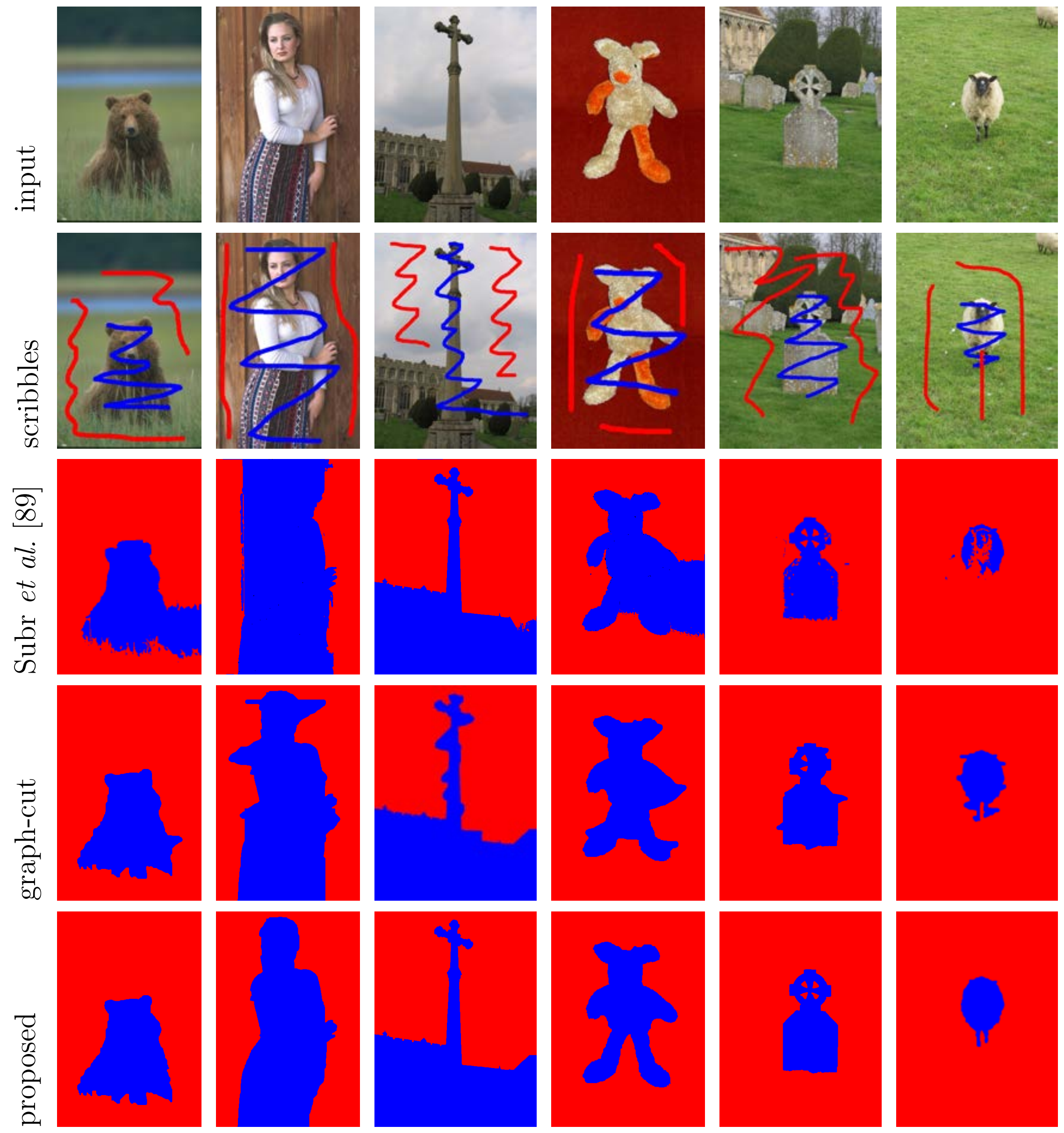


Figure 3.6: Illustrative results on user scribbles with large amount of errors.

pseudo-polynomial bound on the number of iterations assuming the numerator and denominator is bounded by polynomials in number of variables n (Thm.3.4). Finally we show that the proposed robust segmentation energy function satisfy this condition and the pseudo-polynomial bound on the number of iterations is applicable (Thm.3.5).

In this section, we are going to show that the Newton's algorithm for robust segmentation (Alg.3.1) converges in pseudo-log number of iterations $O(\log n + \log C_{max})$ (Thm.3.5), where n is the number of pixels, and C_{max} is the maximum absolute value of energy term coefficients of numerator and denominator. The proofs of monotonic denominator decreasing (Lemma.3.2), error fractional reduction (Thm.3.3) and pseudo-polynomial number of iterations when both numerator and denominator are bounded by polynomials in n (Thm.3.4) are elaborated versions shown in [74] and [35]. Although the lemma and theorem claims are the same, the proofs given here contain more intuitive explanations including where all the inequalities come from. Finally, it is novel to identify which polynomials the numerator and denominator are bounded by, and therefore conclude the bound of number of iterations for robust segmentation's Dinkelbach's algorithm (Thm.3.5).

3.4.1 Simplified notation and lemma

To simplify notation, we rewrite $R(\mathbf{x}) = P(\mathbf{x})/Q(\mathbf{x})$ to be $R(\mathbf{x}) = f(\mathbf{x})/g(\mathbf{x})$, where $g(\mathbf{x}) > 0$. In each Newton's method iteration, we have $\lambda_{k+1} = R(x_k) = f(x_k)/g(x_k)$.

Lemma 3.2. *$g(\mathbf{x}_{k+1}) < g(\mathbf{x}_k), \forall 1 \leq k < K-1$, where K is the number of iterations Alg.3.1 takes to converge. In another word, $\{g(\mathbf{x}_k)\}$ is a strictly decreasing sequence except the last iteration. For the last iteration, $g(\mathbf{x}_K) \leq g(\mathbf{x}_{K-1})$.*

Proof. When $1 \leq k < K-1$, we have

$$\begin{aligned}
0 &= f(\mathbf{x}_{k+1}) - \lambda_{k+2} g(\mathbf{x}_{k+1}) \\
&> f(\mathbf{x}_{k+1}) - \lambda_{k+1} g(\mathbf{x}_{k+1}) \\
&= f(\mathbf{x}_{k+1}) - \lambda_k g(\mathbf{x}_{k+1}) + \lambda_k g(\mathbf{x}_{k+1}) - \lambda_{k+1} g(\mathbf{x}_{k+1}) \\
&\geq f(\mathbf{x}_k) - \lambda_k g(\mathbf{x}_k) + \lambda_k g(\mathbf{x}_{k+1}) - \lambda_{k+1} g(\mathbf{x}_{k+1}) \\
&= \lambda_{k+1} g(\mathbf{x}_k) - \lambda_k g(\mathbf{x}_k) + \lambda_k g(\mathbf{x}_{k+1}) - \lambda_{k+1} g(\mathbf{x}_{k+1}) \\
&= (\lambda_{k+1} - \lambda_k)(g(\mathbf{x}_k) - g(\mathbf{x}_{k+1}))
\end{aligned} \tag{3.8}$$

The second inequality comes from the fact that $\mathbf{x}_k$ minimizes $E(\mathbf{x}; \lambda_k) = f(\mathbf{x}) - \lambda_k g(\mathbf{x})$. Note $\lambda_{k+1} < \lambda_k$ according to Thm.3.1. Thus, $g(\mathbf{x}_k) - g(\mathbf{x}_{k+1}) > 0$, i.e., $g(\mathbf{x}_{k+1}) < g(\mathbf{x}_k), \forall 1 \leq k < K-1$.

When $k = K-1$, Eq.(3.8) still holds except the first strict inequality should be replaced with the equality. Thus, we have $g(\mathbf{x}_K) \leq g(\mathbf{x}_{K-1})$. □

3.4.2 Error reduction in each iteration

With Lemma 3.2, it can be shown that the error at each iteration is reduced to a fraction of the error in previous iteration.

Theorem 3.3.

$$\frac{\lambda_{k+1} - \lambda^*}{\lambda_k - \lambda^*} \leq 1 - \frac{g(\mathbf{x}^*)}{g(\mathbf{x}_k)} < 1$$

where

$$r_k = 1 - \frac{g(\mathbf{x}^*)}{g(\mathbf{x}_k)} \geq 0$$

decreases towards 0 with increasing k.

Proof.

$$\begin{aligned}
\frac{\lambda_{k+1} - \lambda^*}{\lambda_k - \lambda^*} &= \frac{\frac{f(\mathbf{x}_k)}{g(\mathbf{x}_k)} - \lambda^*}{\lambda_k - \frac{f(\mathbf{x}^*)}{g(\mathbf{x}^*)}} \\
&= \frac{g(\mathbf{x}^*)[f(\mathbf{x}_k) - \lambda^* g(\mathbf{x}_k)]}{g(\mathbf{x}_k)[\lambda_k g(\mathbf{x}^*) - f(\mathbf{x}^*)]} \\
&= -\frac{g(\mathbf{x}^*)[f(\mathbf{x}_k) - \lambda^* g(\mathbf{x}_k)]}{g(\mathbf{x}_k)[f(\mathbf{x}^*) - \lambda_k g(\mathbf{x}^*)]} \\
&= -\frac{g(\mathbf{x}^*)[f(\mathbf{x}_k) - \lambda_k g(\mathbf{x}_k) + (\lambda_k - \lambda^*)g(\mathbf{x}_k)]}{g(\mathbf{x}_k)[f(\mathbf{x}^*) - \lambda_k g(\mathbf{x}^*)]} \\
&= -\frac{g(\mathbf{x}^*)[f(\mathbf{x}_k) - \lambda_k g(\mathbf{x}_k)]}{g(\mathbf{x}_k)[f(\mathbf{x}^*) - \lambda_k g(\mathbf{x}^*)]} - \frac{g(\mathbf{x}^*)(\lambda_k - \lambda^*)g(\mathbf{x}_k)}{g(\mathbf{x}_k)[f(\mathbf{x}^*) - \lambda_k g(\mathbf{x}^*)]} \\
&\leq -\frac{g(\mathbf{x}^*)}{g(\mathbf{x}_k)} + \frac{g(\mathbf{x}^*)(\lambda^* - \lambda_k)}{f(\mathbf{x}^*) - \lambda_k g(\mathbf{x}^*)} \qquad (3.9) \\
&= -\frac{g(\mathbf{x}^*)}{g(\mathbf{x}_k)} + \frac{f(\mathbf{x}^*) - \lambda_k g(\mathbf{x}^*)}{f(\mathbf{x}^*) - \lambda_k g(\mathbf{x}^*)} \\
&= 1 - \frac{g(\mathbf{x}^*)}{g(\mathbf{x}_k)}
\end{aligned}$$

The inequality at Eq.(3.9) comes from the fact that $\mathbf{x}_k$ minimizes $E(\mathbf{x};\lambda_k) = f(\mathbf{x}) - \lambda_k g(\mathbf{x})$. In another word, we know

$$f(\mathbf{x}_k) - \lambda_k g(\mathbf{x}_k) \leq f(\mathbf{x}^*) - \lambda_k g(\mathbf{x}^*) \tag{3.10}$$

Since $\lambda_k > \lambda_{k+1} \geq \lambda^*$ and $g(\mathbf{x}) > 0, \forall \mathbf{x}$, we have $f(\mathbf{x}^*) - \lambda_k g(\mathbf{x}^*) < f(\mathbf{x}^*) - \lambda^* g(\mathbf{x}^*) = 0$. Thus, Eq.(3.10) becomes

$$\begin{aligned} \frac{f(\mathbf{x}_k) - \lambda_k g(\mathbf{x}_k)}{f(\mathbf{x}^*) - \lambda_k g(\mathbf{x}^*)} &\geq 1 \\ -\frac{g(\mathbf{x}^*)}{g(\mathbf{x}_k)} \cdot \frac{f(\mathbf{x}_k) - \lambda_k g(\mathbf{x}_k)}{f(\mathbf{x}^*) - \lambda_k g(\mathbf{x}^*)} &\leq -\frac{g(\mathbf{x}^*)}{g(\mathbf{x}_k)} \end{aligned}$$

This is the inequality at Eq.(3.9).

From Thm.3.2, $g(\mathbf{x}_k) \geq g(\mathbf{x}_{k+1}) \geq g(\mathbf{x}^*)$. Thus, $g(\mathbf{x}^*)/g(\mathbf{x}_k) \leq 1$, and $r_i = 1 - g(\mathbf{x}^*)/g(\mathbf{x}_k) \geq 0$.

According to Thm.3.2, if $1 \leq k < K-1$, $r_{k+1} < r_k$, where K is the number of iterations in Alg.3.1. If $k = K-1$, $r_{k+1} = r_K \leq r_{K-1}$. Moreover, $\mathbf{x}_K = \mathbf{x}^*$, thus $g(\mathbf{x}_K) = g(\mathbf{x}^*)$ and $r_K = 0$.

In sum, $r_1 > r_2 > \ldots > r_{K-1} \geq r_K = 0$. Thus, r_k decreases towards 0 with increasing k. $\square$

Note the error reduction bound $r_k = 1 - g(\mathbf{x}^*)/g(\mathbf{x}_k)$ will improve (become smaller) as k becomes larger. This implies very rapid convergence of the algorithm, especially when close to the optimal solution.

3.4.3 Number of iteration bound

Theorem 3.4. *For integer valued function $f(\mathbf{x})$ and $g(\mathbf{x}) > 0$, if the magnitude of $f(\mathbf{x})$ and $g(\mathbf{x})$ is bounded by a polynomial in n, i.e., $|f(\mathbf{x})| \leq p(n)$ and $g(\mathbf{x}) \leq p(n)$, then number of iterations in Alg.3.1 is $O(\log n + \log C_{max})$, where n is number of variables and C_{max} is the largest absolute value of any monomial in $p(n)$.*

Proof. We consider the smallest interval δ between any two different $\mathbf{x}$ and $\mathbf{x}'$ that results in different ratios $R(\mathbf{x}) \neq R(\mathbf{x}')$.

$$\delta = R(\mathbf{x}) - R(\mathbf{x}') = \frac{f(\mathbf{x})}{g(\mathbf{x})} - \frac{f(\mathbf{x}')}{g(\mathbf{x}')} = \frac{f(\mathbf{x})g(\mathbf{x}') - f(\mathbf{x}')g(\mathbf{x})}{g(\mathbf{x})g(\mathbf{x}')}$$

Since f and g are integer valued, $f(\mathbf{x})g(\mathbf{x}') - f(\mathbf{x}')g(\mathbf{x}) \geq 1$. Thus,

$$\delta \geq \frac{1}{g(\mathbf{x})g(\mathbf{x}')}$$

Since $g(\mathbf{x})$ is bounded by polynomial $p(n)$, we have $g(\mathbf{x}) \leq p(n) \leq C_{max} \cdot q(n)$ where C_{max} is the largest absolute value of any monomial in $p(n)$, and $q(n)$ is a polynomial whose coefficients are all ones. Thus,

$$\delta \geq \frac{1}{C_{max}^2 \cdot q^2(n)}$$

If $|R(\mathbf{x})-R(\mathbf{x}')| < \delta$, then $R(\mathbf{x}) = R(\mathbf{x}')$. According to Thm.3.3, each iteration the error is reduced by a ratio of at least $r_k = 1 - g(\mathbf{x}*)/g(\mathbf{x}_k) < 1$. Thus, there must exist $\theta < 1$ such that in each iteration, $|\lambda_k - \lambda^*| < \theta|(\lambda_{k-1} - \lambda^*)|, k > 1$.

Assume after N iteration, the error is reduced to be less than $1/C_{max}^2 \cdot q^2(n)$, then

$$
\begin{aligned}
(\lambda_0 - \lambda^*)\theta^N &= \frac{1}{C_{max}^2 \cdot q^2(n)} \\
(\frac{1}{\theta})^N &= (\lambda_0 - \lambda^*) C_{max}^2 \cdot q^2(n) \\
N &= \left\lceil \log_{\frac{1}{\theta}} (\lambda_0 - \lambda^*) C_{max}^2 \cdot q^2(n) \right\rceil \\
N &= \left\lceil \log_{\frac{1}{\theta}} (\lambda_0 - \lambda^*) + 2\log_{\frac{1}{\theta}} C_{max} + 2\log_{\frac{1}{\theta}} q(n) \right\rceil \qquad (3.11)
\end{aligned}
$$

After at most $N+1$ iterations, the error will be smaller than δ, as follows.

$$0 \leq \lambda_{N+1} - \lambda^* < \frac{1}{C_{max}^2 \cdot q^2(n)} \leq \delta$$

Therefore, $\lambda_{N+1} = \lambda^*$. The algorithm terminates.

Note $\lambda_0 - \lambda^* \leq \frac{\max|f|}{\min g} - \frac{-\max|f|}{\min g} = 2\frac{\max|f|}{\min g} \leq 2\frac{p(n)}{1}$. Convert $N+1$ (Eq.3.11) using big O notation, the algorithm terminates within $O(\log n + \log C_{max})$ number of iterations. □

Theorem 3.5. *The Newton's algorithm for robust interactive segmentation converges in $O(\log n + \log C_{max})$ iterations, where n is the number of pixels, and C_{max} is the largest absolute value of energy term coefficients of numerator and denominator.*

Proof. Note our robust graph-cut energy has integer valued f and g over binary variables $\mathbf{x}$.

The numerator contains n data terms $D_p, \forall p \in \mathcal{P}$, and at most n^2 smoothness terms $V_{pq}, \forall (p,q) \in \mathcal{N}$. Thus, the numerator is bounded by $n \max |D_p| + n^2 \max |V_{pq}|$.

The denominator contains only n unary user seed utility terms and constant M. Thus, the denominator is bounded by $n \max U_p + M$.

Let $C_{max} = \max\{|D_p|, |V_{pq}|, U_p, M\}$, then $|f(\mathbf{x})| \leq C_{max} \cdot (1 + n + n^2)$ and $g(\mathbf{x}) \leq C_{max} \cdot (1+n+n^2)$. According to Thm.3.4, the number of iterations in Alg.3.1 is $O(\log n + \log C_{max})$. $\square$

3.5 Discussion

3.5.1 Ratio energy

Ratio energy has been used for image segmentation in different ways. Wang et al.[100] used a ratio energy maximizing the average intensity difference between foreground and background for segmentation. Kolmogorov et al.[49] alleviates the shrinkage bias of graph-cut segmentation by using foreground volume as the denominator in the ratio energy. Here we use a different idea that the denominator is a utility function which encourages the user input to be respected as much as possible. How strong this encouragement is can be tuned by the parameter M in Eq.(3.1), or, equivalently α in Eq.(3.5). Thus, we can adjust the proposed method for users of different styles as we have shown in the different experiments in Sec.3.3.2, 3.3.3, 3.3.4.

3.5.2 Parameter α setting

To choose an appropriate α value, the basic guideline is that the larger α is, the more tolerant of user input errors the algorithm is. In Fig.3.7b, although by using

$\alpha = 1$ our method can tolerate the erroneous scribble at head, it does not tolerate many other errors of the scribble. In Fig.3.7c, with $\alpha = 10$ the proposed method is doing a much better job. However, there is still erroneous segmentation at the elbow. In Fig.3.7d, using $\alpha = 100$, our method achieves the best segmentation among the uses of those three α values.

On the other hand, a large α value means that the “anchoring” ability of the scribble pixels is reduced. In challenging images, the “anchoring” ability can be crucial to the correct segmentation. Fig.3.8 shows a challenge case of cheetah. When $\alpha = 1$, the paws and the tail of the cheetah are correctly segmented by following the guidance of the foreground scribbles. However, when $\alpha = 100$, the paws and part of the tails are incorrectly segmented as background.

Thus, for those images with the expectation of large user input errors, we can use large α value to accommodate those errors. For challenge images (e.g, with very fine structures, or similar foreground and background profile), the “anchoring” ability can be crucial to the accurate segmentation. Thus a small α value should be selected, and users are advised to provide more accurate scribbles in order to achieve an accurate segmentation.

However, we should note that no matter what α value is chosen, if the evidence for the foreground/background shown by the graph-cut energy is strong enough, the erroneous scribbles can be automatically corrected. Introducing parameter α allows us to control how to determine the evidence is strong enough.

3.5.3 Relationship to parametric max-flow

The proposed method requires a series of max-flow computation with λ as the parameter of some source-to-vertex or vertex-to-sink edges. This looks similar to *parametric max-flow problem* [49, 31], a combinatorial problem which solves a series of max-flow in the asymptotic complexity of computing a single max-flow. However, we are going to show that this technique cannot be formulated as a parametric max-flow problem.

The parametric max-flow problem aims to solve a series of max-flow problems (or equivalently minimum s-t cut problems) for a series of parameter values $\{\lambda_i\}$ in a *parametric network*. A parametric network is a graph/network $G(V \cup \{s, t\}, A_\lambda \cup A_0)$ in which A_λ is the set of arcs with their weight parameterized by λ and A_0 is the set of arcs with constant weight. s and t are source and sink vertices, respectively. V is the set of all other vertices.

A parametric network requires the arc weights to satisfy the following properties (Section 2.3. in [31]):

1. $A_\lambda = \{(s, v)\} \cup \{(v, t)\}, v \in V$. i.e., all parameterized arcs are either from source s or sink t.
2. $A_0 = \{(u, v)\}, u, v \in V$. i.e., all arcs not involving source s or sink t have constant weights.
3. arc weight $w_\lambda(s, v)$ is a nondecreasing (or nonincreasing, respectively) function of λ for all $v \in V$.
4. arc weight $w_\lambda(v, t)$ is a nonincreasing (or nondecreasing, respectively) function

of λ for all $v \in V$.

One appealing theoretical property of a parametric network satisfying the above criterion is that there can only be at most $|V| + 1$ distinct minimum s-t cuts. And all $|V| + 1$ cuts can be computed using the time complexity of computing a *single* s-t cut, instead of $|V| + 1$ cuts.

The proposed robust segmentation method satisfies the properties 1 and 2. However, it does not satisfy the properties 3 and 4, which requires the arc weight either monotonically increase from source to sink or monotonically decrease from source to sink. The problem comes from the user input utility function defined in Eq.(3.3),(3.4). To be more specific, observe the following arc weights.

$$w_\lambda(s, v_p) = D_p(x_p = \text{`}bg') - \lambda U(x_p = \text{`}bg'), \quad p \in \mathcal{S}_B \tag{3.12}$$

is nonincreasing since $U(x_p = \text{`}bg') > 0$

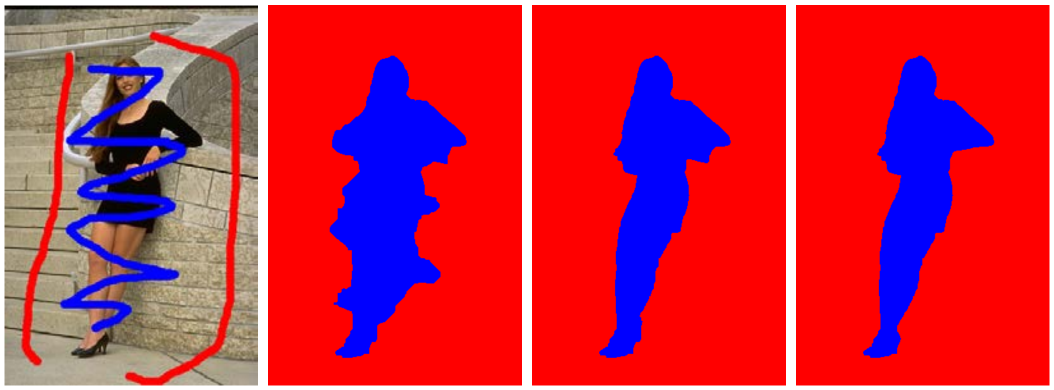
$$w_\lambda(v_p, t) = D_p(x_p = \text{`}ob') - \lambda U(x_p = \text{`}ob'), \quad p \in \mathcal{S}_F \tag{3.13}$$

is nonincreasing since $U(x_p = \text{`}ob') > 0$.

This behavior of the arc weights prevents parametric max-flow technique to be applied in the proposed method.

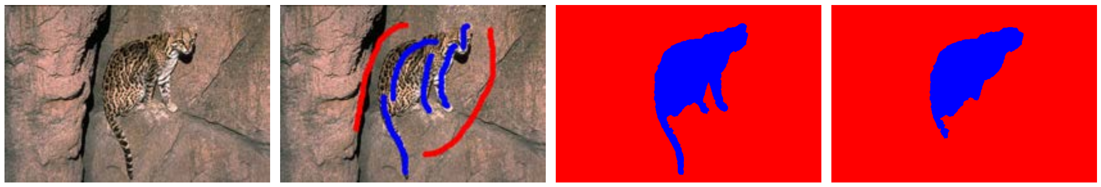
3.6 Conclusion

We propose a novel ratio energy function to tolerate user input errors in scribble-based interactive segmentation. It minimizes a graph-cut energy which incor-



(a) scribbles (b) $\alpha = 1$ (c) $\alpha = 10$ (d) $\alpha = 100$

Figure 3.7: The larger α is, the more error-tolerance the algorithm can achieve.



(a) input (b) scribbles (c) $\alpha = 1$ (d) $\alpha = 100$

Figure 3.8: The larger α is, the less "anchoring" ability the scribble has. Note when $\alpha = 100$, the paws and part of the tail of the cheetah is segmented as background although specified as foreground seeds.

porates both region and boundary information and maximizes the portion of scribbles that are respected in the segmentation result. The ratio energy can be optimized exactly using an efficient algorithm. Experiments based on synthetic and manual scribbles are conducted to validate the algorithm's robustness to large amount of user input errors and the ability to achieve high segmentation accuracy when presented with user input errors. Promising results are shown in our the experiments. Tight theoretical running time bounds are proven to guarantee fast convergence.

CHAPTER 4
SURFACE-OBJECT SEGMENTATION WITH MULTI-SCALE TECHNIQUE

A novel segmentation method is proposed to simultaneously segment topologically flexible objects and interacting adjacent surfaces with known topologies. Incorporating such surface-object interaction prior information plays an important role in accurate segmentation when the target object and surfaces have similar intensities and no clear edge lies between them. Moreover, a novel multi-scale approach is used to further boost segmentation accuracy of the target object, which utilizes both fine level pixel-wise information as well as coarse level region-wise information.

The problem is formulated as a Markov Random Field (MRF) energy minimization problem, which can be efficiently and exactly solved by computing a minimum *s*-*t* cut in an appropriately constructed graph. The novel approach can simultaneously segment topologically flexible objects (via graph cut) and the interacting adjacent surface with known topology (via graph searching). The ability of incorporating the surface-object interaction information and multi-scale information increases segmentation accuracy and robustness.

The performance of the proposed method is assessed on the application of lung tumor segmentation in 38 Mega-Voltage Cone-Beam CT (MVCBCT) datasets. The Dice coefficient (DSC) is improved from 0.759±0.102 to 0.876±0.027, and the average symmetric surface distance is improved from 4.417±3.211 *mm* to 1.370±0.468 *mm*. A student t-test shows that the improvements are statistically significant.

4.1 Introduction

There are compelling applications demanding a novel approach for simultaneous segmentation of mutually interacting objects/regions and surfaces, such as two nearby organs, tumors located in proximity to organ boundaries, cysts positioned close to other adjacent structures, and moving tumors along with lung boundaries in dynamic or longitudinal images. In this context, the target objects may have an arbitrary and unknown topological shape, while the interacting surfaces may have known topology.

We develop a new segmentation approach that would for the first time integrate graph cut ([12]) and graph searching ([52]) to combine the strengths of both methods while overcoming their individual drawbacks. The novel approach can simultaneously segment topologically flexible objects (via graph cut) interacting with adjacent surfaces with known topology (via graph searching). Incorporating such surface–object interrelations increases segmentation accuracy and robustness by utilizing clues from interacting surfaces. Moreover, a multi-scale scheme is developed to segment the topologically flexible objects incorporating both pixel-wise information and region-wise information based on a data-drive over-segmentation of the image ([76]). Information from different scales are mutually propagated to each other by a soft label consistency constraint. Finally, a graph encoding both surface-object interaction information and the multi-scale object information is constructed, and a minimum s-t cut in the graph defines the segmentation of the target object and the interacting surfaces.

Multi-scale methods has been widely used as a speedup technique to some existing segmentation methods ([51, 59, 77]). These methods usually work sequentially on different scales. Recently the multi-scale methods are also shown to be helpful for improving segmentation accuracy in natural images ([44, 21]). [21] builds affinity matrices at different scales and enforce the cross-scale constraint that the segmentation at coarse scale should be locally an average of the fine scale segmentation. The multi-scale segmentation is then conducted simultaneously over all scales by solving an approximate eigenvector problem. The cross-scale constraint propagates information across different scales to reach a consistent segmentation at all scales. The method coarsens an image based on a regular grid, which may blur details at the coarse levels. A data-driven coarsening scheme, such as over-segmentation, may preserve details better at the coarse levels. [44] used segments/regions in an over-segmented image (obtained by methods such as mean shift) as nodes in the coarse level. Each pixel at the original resolution is used as nodes in the fine level. A soft label consistency constraint is enforced between the coarse level and the fine level. The segmentation is computed by a convex optimization technique on both levels simultaneously. The problem is reduced to a Random walk with a restart (RWR) problem, for which a straightforward implementation requires either quadratic space and cubic pre-computation time or slow response time on queries ([94]).

[24] proposed a multi-region segmentation method which enforces containment, exclusion and attraction geometric constraints among multiple objects. The segmentation problem is reduced to computing a minimum s-t cut in an appropriate

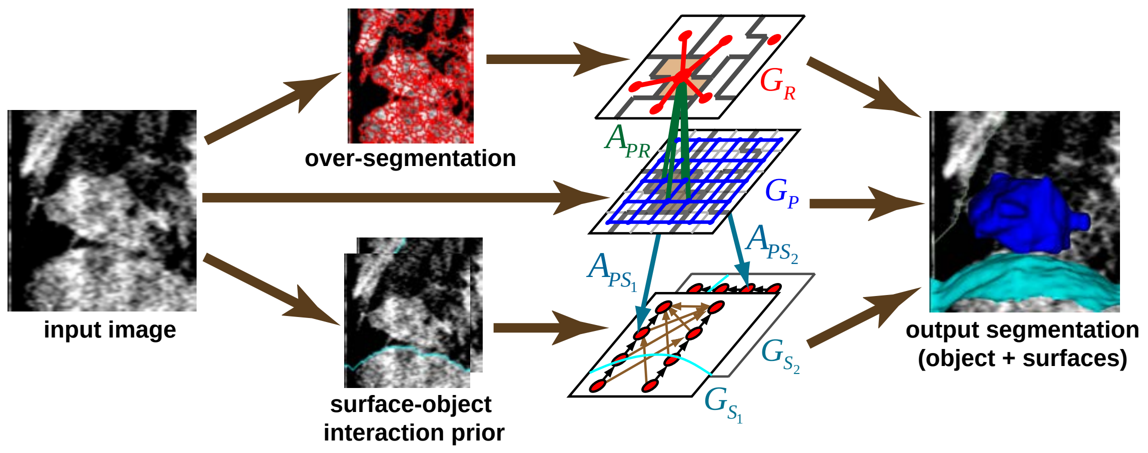


Figure 4.1: Workflow of the multi-scale segmentation with surface-object interaction prior. Surface-object interaction prior is encoded in the resulting graph. Region-wise information from an over-segmentation of the original image is also incorporated. The target object and the interacting surfaces are segmented simultaneously. More details are presented in Sec.4.2.

constructed graph. The graph contains a set of graph-cut based subgraphs. Inter-subgraph arcs are added to encode the geometric constraints. No topological prior information is enforced for any region/object.

In our preliminary study ([85]), we developed a novel surface-object segmentation method, in which the target object is segmented simultaneously with one or more interacting surfaces along a specific direction to improve segmentation accuracy and robustness. The simultaneous segmentation of the adjacent surface with the target object prevents the target object segmentation from "leaking" beyond the adjacent surface. The method integrates a graph-cut subgraph for the object segmentation with a graph-searching subgraph ([52]) for the adjacent surface segmentation.

In this work, we further generalize the surface-object segmentation method with the interaction of multiple surfaces from *different directions*, and demonstrate how to incorporate a novel multi-scale segmentation scheme into a single optimization process to achieve high segmentation accuracy. See Fig.4.1 for the workflow of the proposed method. The pixel-wise and region-wise information of the target object is encoded using graph-cut based graphs in a multi-scale scheme, handling objects with an arbitrary topology. The interacting surface information is encoded using a graph-searching based graph, which enforces the prior topology constraint of the surfaces. The interaction arcs between subgraphs for pixel-wise information and region-wise information mutually propagates the low-level cues and higher-order cues of the target object. The interaction arcs between the pixel-wise subgraph and the surface-segmentation subgraphs enforces the surface-object interaction prior information. A Markov Random Field (MRF) energy is then optimized by computing a minimum *s*-*t* cut in the appropriately constructed graph to achieve the segmentation of the target object and the interacting surfaces.

This section is organized as follows. In Sec.4.2, we describe the method. The MRF energy is introduced in Sec.4.2.1-4.2.3 and the graph construction is discussed in Sec.4.2.4-4.2.4.3. In Sec.4.3, we describe the experiment data and settings, energy term design and parameter settings. The quantitative result and the qualitative result of the experiment are also shown. We discuss the novelties and some issues of our method in Sec.4.4 and conclude in Sec.4.5.

4.2 Method

The proposed method formulates the segmentation method as an MRF energy minimization problem, which can be optimized by computing a minimum s-t cut. We first introduce the MRF energy for the multi-scale segmentation of the target object and the energy for the interacting surface segmentation. A novel graph transformation is then presented to encode the energy function.

Our energy comprises three energy terms: the multi-scale graph-cut based object segmentation term E_{MS} for segmenting the target object, the terrain-like surface segmentation term E_{TS} for segmenting the interacting surfaces adjacent to the target object, and the surface-object interaction term $E_{MS\text{–}TS}$ enforcing the interaction prior information between the target object and adjacent surfaces.

$$E = E_{MS}(\mathbf{f}_{\mathcal{P}}, \mathbf{f}_{\mathcal{R}}) + E_{TS}(\mathbf{S}_{\mathcal{S}}) + E_{MS\text{–}TS}(\mathbf{f}_{\mathcal{P}}, \mathbf{S}_{\mathcal{S}}) \tag{4.1}$$

where $\mathbf{f}_{\mathcal{P}}$ is the set of variables labeling each pixel as object or background, $\mathbf{f}_{\mathcal{R}}$ is the set of variables labeling each region in an over-segmented image as object or background (see Fig.4.2 for an example of over-segmentation), $\mathbf{S}_{\mathcal{S}}$ is the set of variables defining the locations of all interacting surfaces. Now we discuss each term in more details.

4.2.1 Multi-scale target object segmentation term

In this section, we consider a novel multi-scale segmentation energy which incorporates both pixel-wise information from the original image and region-wise information based on a data-drive over-segmentation of the original image.

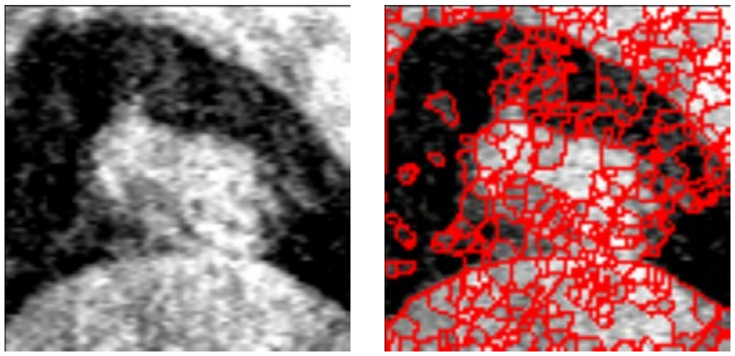

(a) input image (b) over-segmentation

Figure 4.2: Over-segmentation of an image generated by watershed.

Given an image, assume $\mathcal{P}$ is the set of all pixel locations and $\mathcal{I}$ is the set of intensities associated with all the pixels. For each pixel p, a label $f_p \in \mathcal{L} = \{0, 1\}$ is assigned. If a pixel is labeled as 0, then it is assigned a "background" label in the segmentation; otherwise, it is assigned an "object" label. Over-segmentation of an image is the partition of an image into *many* self-coherent regions/segments. A single meaningful object, such as a tumor, can be divided into multiple regions instead of being segmented as a single object. Techniques such as mean shift ([20]) and watershed ([76]) can be used to generate an over-segmentation of the image. An over-segmentation example by watershed is shown in Fig.4.2. Although the over-segmentation result does not directly correspond to anatomically meaningful objects, it still groups portions of image into meaningful/self-homogeneous regions. Let $\mathcal{R}$ be the set of segments in the over-segmentation. A pixel p is a *child pixel* of a region $r \in \mathcal{R}$ if $p \in r$. In this case, r is called the *parent region* of pixel p.

The multi-scale energy E_{MS} consists of three terms: a pixel layer term E_P, a region layer term E_R, and a pixel-region consistency term E_{PR} (Eq.(4.2)). See Fig.4.4

for an illustration of the pixel layer and the region layer.

$$E_{MS}(\mathbf{f}_{\mathcal{P}}, \mathbf{f}_{\mathcal{R}}) = E_P(\mathbf{f}_{\mathcal{P}}) + E_R(\mathbf{f}_{\mathcal{R}}) + E_{PR}(\mathbf{f}_{\mathcal{P}}, \mathbf{f}_{\mathcal{R}}) \tag{4.2}$$

$$E_P(\mathbf{f}_{\mathcal{P}}) = \sum_{p \in \mathcal{P}} D_p(f_p) + \eta^P \sum_{(p,q) \in \mathcal{N}_{\mathcal{P}}} V_{pq}(f_p, f_q) \tag{4.3}$$

$$E_R(\mathbf{f}_{\mathcal{R}}) = \sum_{r \in \mathcal{R}} D_r(f_r) + \eta^R \sum_{(r_1,r_2) \in \mathcal{N}_{\mathcal{R}}} V_{r_1,r_2}(f_{r_1}, f_{r_2}) \tag{4.4}$$

$$E_{PR}(\mathbf{f}_{\mathcal{P}}, \mathbf{f}_{\mathcal{R}}) = \sum_{r \in \mathcal{R}} \sum_{p \in r} \Theta_{p,r}(f_p, f_r) \tag{4.5}$$

The pixel layer term $E_P(\mathbf{f}_{\mathcal{P}})$ (Eq.(4.3)) is a graph-cut energy term containing a data term $D_p(f_p)$ and a smoothness term $V_{pq}(f_p, f_q)$ over all pixels $\mathcal{P}$. The data term is a unary term which inversely measures the likelihood of pixel p belonging to the object or background. For example, $D_p(f_p = 1) < D_p(f_p = 0)$ if pixel p is more likely to be in the object instead of background. Minimizing the data term encourages pixels which are more likely to belong to the object to be indeed labeled as object, and vice versa. The smoothness term is a pairwise term defined over the pixel neighborhood system $\mathcal{N}_P$, which can be a 4 or 8 neighborhood system in 2D and a 6 or 26 neighborhood system in 3D. In this work, we use a 6-neighborhood setting for 3D images. For a pair of neighboring pixels $(p, q) \in \mathcal{N}_{\mathcal{P}}$, $V_{pq}(f_p, f_q) = 0$ if $f_p = f_q$ and $V_{pq}(f_p, f_q) > 0$ if $f_p \neq f_q$. This encourages neighboring pixels to have the same label, which helps to generate more spatially coherent segmentations. η^P is the balancing coefficient between the data term and the smoothness term in the pixel layer.

The region layer term $E_R(\mathbf{f}_{\mathcal{R}})$ (Eq.(4.4)) consists of a data term and a smoothness term similar to $E_P(\mathbf{f}_\mathcal{P})$, except that the variable $\mathbf{f}_\mathcal{R}$ represents regions in the over-segmentation instead of pixels in the original image as in $\mathbf{f}_\mathcal{P}$. The data term $D_r(f_r)$ for a region $r \in \mathcal{R}$ can be computed from some aggregate statistics of its children pixels, such as the average data term over its children pixels. It can also incorporate more sophisticated information such as texture description within the region. The region neighboring system $\mathcal{N}_\mathcal{R}$ can be defined in different ways. In this work, we define two regions (r_1, r_2) to be neighboring if the Euclidean distance between their centroids is within a certain threshold. η^R is a balancing coefficient between the two terms in the region layer.

The pixel-region consistency term $E_{PR}(\mathbf{f}_\mathcal{P}, \mathbf{f}_\mathcal{R})$ (Eq.(4.5)) penalizes the difference between each pixel label f_p and its parent region label f_r. If $f_p = f_r$, then $\Theta_{p,r}(f_p, f_r) = 0$. If $f_p \neq f_r$, then $\Theta_{p,r}(f_p, f_r) > 0$. This encourages the pixel label to agree with its parent region label, i.e., the segmentation at the coarse level and the fine level should be consistent as much as possible. Note this consistency constraint is soft–the children pixels can have different labels from their parent regions as long as the corresponding penalty is paid. This allows the boundary to be extracted accurately at the pixel resolution, instead of having to be the same as the boundary in the over-segmentation.

4.2.2 Terrain-like surface segmentation term

A 3D *terrain-like surface* is a surface that intersects exactly once with every column along one specific direction. See Fig.4.3 for an example of a terrain-like surface

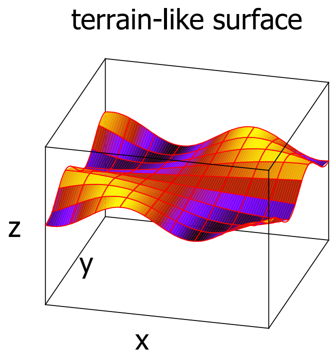


Figure 4.3: Terrain-like surface along the z-direction.

along the z-direction. Assuming the image size is $X \times Y \times Z$, then a terrain-like surface along the z-direction is defined as $S : \{0, \ldots, X-1\} \times \{0, \ldots, Y-1\} \rightarrow \{0, \ldots, Z-1\}$. In other words, for any 2D coordinate (x, y), $S(x, y)$ returns the surface height along the z-direction, i.e., the location of the terrain-like surface at (x, y). Graph-searching ([52]) is an effective approach for segmenting terrain-like surfaces.

Suppose $\mathcal{S}$ is the set of surfaces we want to segment, then for every surface $S \in \mathcal{S}$, the terrain-like surface segmentation term $E_{TS}(\mathbf{S}_{\mathcal{S}})$ is the sum of *on-surface* cost c_p for all pixels p that are on the surface S (Eq.(4.6)).

$$E_{TS}(\mathbf{S}_{\mathcal{S}}) = \sum_{S \in \mathcal{S}} \eta^S \sum_{p \in S} c_p \tag{4.6}$$

The cost c_p is the inverse likelihood of pixel being *on* the surface. This cost can be obtained by edge detection. The surface S is also subject to smoothness constraint, which restricts the surface from jumping abruptly at nearby locations. The smoothness constraint is defined along the x-direction as $|S(x, y) - S(x+1, y)| \leq \Delta_x, 0 \leq x < X - 1$, where Δ_x is a predefined threshold limiting the surface jump at nearby (x, y) locations along the x-direction. Similarly, the smoothness along the y-direction is defined as $|S(x, y) - S(x, y+1)| \leq \Delta_y, 0 \leq y < Y - 1$. η^S is a balancing coefficient between the surface segmentation term for surface S and other terms.

4.2.3 Surface-object interaction term

A surface is *interacting* with the target object if the target object is at least d_{min} away from the surface. From now on, we assume that the object is *below* the z-terrain-like surface S for at least distance d_{min}. Note this assumption does not lose any generality. Because the cases when the object is *above* the z-terrain-like surface and the cases when the terrain-like surface is actually x or y-terrain-like can be reduced to this case by rotating and flipping the image. This rotating and flipping of the image does not affect the graph construction, as will be discussed in Sec.4.2.4.3.

The surface-object interaction term $E_{MS\text{–}TS}(\mathbf{f}_\mathcal{P}, \mathbf{S}_\mathcal{S})$ is used to enforce the prior information that the target object O is at least d_{min} below surface S (Eq.(4.7)).

$$E_{MS\text{–}TS}(\mathbf{f}_\mathcal{P}, \mathbf{S}_\mathcal{S}) = \sum_{S \in \mathcal{S}} \sum_{\substack{p(x,y,z) \\ S(x,y)-z<d_{min}}} \Phi_p^S(f_p) \tag{4.7}$$

To be precise, for all pixels $p(x, y, z) \in O$, we would like to enforce the prior $S(x, y) - z \geq d_{min}$. Thus, $\Phi_p^S(f_p = 0) = 0$ since $f_p = 0$ means pixel p is background while the surface-object constraint only concerns *object* pixels. Moreover, $\Phi_p^S(f_p = 1) > 0$ is the penalty paid when $S(x, y) - z < d_{min}$. If $\Phi_p^S(f_p = 1) = +\infty$, then we enforce d_{min} as a hard constraint that no object voxels can be higher than d_{min} below surface S. Otherwise, the minimum surface-object distance is enforced as a soft constraint–

as long as the penalty is paid, voxels violating the minimum surface-object distance constraint can still be labeled as object.

4.2.4 Graph construction

A graph consisting of multiple subgraphs is constructed encoding the MRF energy terms in Eq.(4.1). Computing a minimum s-t cut in the constructed graph minimizes the MRF energy exactly.

Two subgraphs G_P, G_R are built for encoding the object segmentation term $E_{MS}(\mathbf{f}_{\mathcal{P}}, \mathbf{f}_{\mathcal{R}})$, which encodes pixel-wise information and region-wise information respectively. A set of arcs A_{PR} is added between G_P and G_R to enforce segmentation consistency between the fine level (pixel-layer) and coarse level (region-layer). For each surface $S \in \mathcal{S}$, one subgraph G_S is built to encode the corresponding surface segmentation term. A set of arcs A_{PS} is added between the pixel-wise object segmentation subgraph G_P and each interacting surface segmentation subgraph $G_S, S \in \mathcal{S}$. This encodes the surface-object interaction term. A more detailed description for each subgraph is given in Sec.4.2.4.1,4.2.4.2, 4.2.4.3.

4.2.4.1 Graph construction for multi-scale segmentation

To encode the multi-scale object segmentation term (Eq.(4.2)), two subgraphs $G_P(V_P, A_P), G_R(V_R, A_R)$ and a set of interaction arcs A_{PR} between them are constructed. The pixel-layer subgraph G_P and region-layer subgraph G_R encode the pixel layer term and the region layer term respectively. The interaction arcs A_{PR} between G_P and G_R encode the pixel-region consistency term between the two layers.

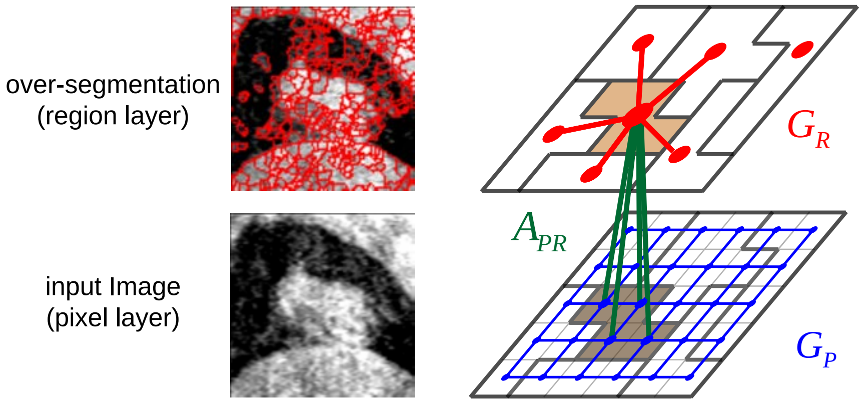


Figure 4.4: The construction of graph G_{MS} for multi-scale object segmentation. (Best read in color.) An over-segmentation of the input image is generated as a coarse level representation of the original image. Two subgraphs are constructed. Subgraph G_P encodes pixel-wise information. Blue arcs in G_P encodes the pixel-wise smoothness term. Subgraph G_R encodes region-wise information. Red arcs in G_R encodes the region-wise smoothness term. The pixel-region interaction arcs A_{PR} (green arcs) enforces the soft pixel-region consistency term.

The pixel layer subgraph $G_P(V_P, A_P)$ is built using graph-cut construction ([12]). For each pixel $p \in \mathcal{P}$, a vertex v_p is added. Every vertex v_p has an incoming arc from source vertex s and an outgoing arc to sink vertex t, carrying weight $D_p(f_p = 0)$ and $D_p(f_p = 1)$ respectively. For every pair of neighboring pixels $(p, q) \in \mathcal{N}_P$, an arc from v_p to v_q with weight $V_{pq}(f_p = 1, f_q = 0)$ and an arc from v_q to v_p with weight $V_{pq}(f_p = 0, f_q = 1)$ are added (blue arcs in Fig.4.4).

Given any s-t of the subgraph G_P, we interpret the labeling f_p of every pixel $p \in \mathcal{P}$ as follows. If vertex v_p is in the source set, then we assign $f_p = 1$ and pixel p is labeled as object. If vertex v_p is in the sink set, then we assign $f_p = 0$ and pixel p is labeled as background.

Now we show that the above graph construction correctly encodes the pixel layer term $E_P(\mathbf{f}_{\mathcal{P}})$ (Eq.(4.3)). When vertex v_p is in the source set, the arc (v_p, t) is cut and the associated penalty $D_p(f_p = 1)$ is correctly enforced. Similarly, when vertex v_p is in the sink set, the arc (s, v_p) is cut and the associated penalty $D_p(f_p = 0)$ is correctly enforced. If two neighboring pixels p, q have the same labels in the segmentation $(f_p = f_q)$, then both arc (v_p, v_q) and arc (v_q, v_p) are not in the cut. No penalty is enforced in this case. But if they have different labels in the segmentation, then either arc (v_p, v_q) is cut (when v_p is in the source set, v_q is in the sink set) and the associated penalty $V_{pq}(f_p = 1, f_q = 0)$ is correctly enforced, or arc (v_q, v_p) is cut (when v_p is in the sink set, v_q is in the source set) and the associated penalty $V_{pq}(f_p = 0, f_q = 1)$ is correctly enforced.

The region layer subgraph $G_R(V_R, A_R)$ is built similarly to G_P. For each region $r \in \mathcal{R}$, a vertex v_r is added. Every vertex v_r has an incoming arc from source vertex s and and an outgoing arc to sink vertex t, carrying weight $D_r(f_r = 0)$ and $D_r(f_r = 1)$ respectively. For every pair of neighboring regions $(r_1, r_2) \in \mathcal{N}_R$, an arc from v_{r_1} to v_{r_2} with weight $V_{r_1,r_2}(f_{r_1} = 1, f_{r_2} = 0)$ and an arc from v_{r_2} to v_{r_1} with weight $V_{r_1,r_2}(f_{r_1} = 0, f_{r_2} = 1)$ are added (red arcs in Fig.4.4).

The inter-subgraph arcs A_{PR} are added as follows. For every pixel $p \in r$, an arc from v_p to its parent region vertex v_r is added with weight $\Theta_{p,r}(f_p = 1, f_r = 0)$. An arc from v_r to v_p is also added with weight $\Theta_{p,r}(f_p = 0, f_r = 1)$ (green arcs in Fig.4.4). When a pixel p has the same label as its parent region r, none of (v_p, v_r) and (v_r, v_p) is cut. If the pixel p is labeled as object but its parent region r is labeled as background, then the arc (v_p, v_r) is cut and the penalty $\Theta_{p,r}(f_p = 1, f_r = 0)$ is correctly enforced.

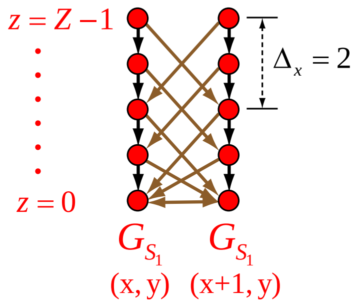
Similarly, if the pixel p is labeled as background but its parent region r is labeled as object, then the arc (v_r, v_p) is cut and the penalty $\Theta_{p,r}(f_p = 0, f_r = 1)$ is correctly enforced.

This completes the graph construction for the multi-scale object segmentation. Fig.4.4 illustrates how G_P, G_R and A_{PR} are constructed. The arcs connecting source and sink vertices are omitted to avoid clutter.

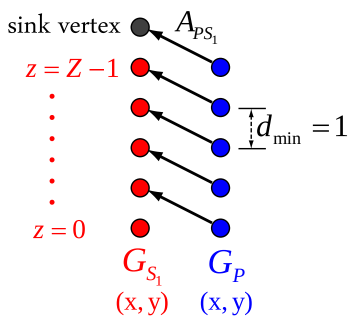
4.2.4.2 Graph construction for surface segmentation

A subgraph $G_S(V_S, A_S)$ is constructed for segmenting each interacting terrain-like surface $S \in \mathcal{S}$. For each pixel p, a vertex v_p^S is created in subgraph G_S. The graph-searching method ([52]) is used to add arcs within each subgraph to encode the energy in Eq.(4.6). The idea is to construct the graph such that given any finite *s*-*t* cut, the source set vertices are consecutive and the sink set vertices are consecutive within each (x, y) column. Moreover, all source set vertices will always lie *below* the sink set vertices in each column. Thus, the *s*-*t* cut of G_S can be interpreted as the surface segmentation using the following mapping: all source set vertices are defined as "below" the surface and all sink set vertices are defined as "above" the surface. $S(x, y)$ is defined as the "highest" source set vertex in (x, y) column. We assign the arc weight carefully so that the cut capacity of any finite *s*-*t* cut encodes the on-surface cost in Eq.(4.6). As a result, by computing a minimum *s*-*t* cut, we get a surface segmentation which minimizes the on-surface cost.

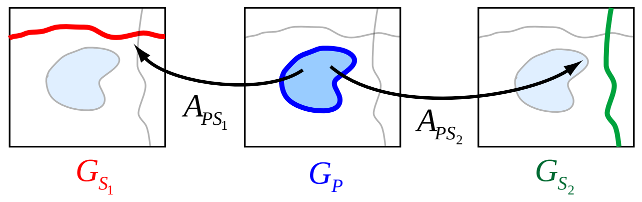
More specifically, three types of arcs are added:



(a) interacting surface segmentation subgraph G_S



(b) surface-object interaction arcs A_{PS}



(c) pixel layer subgraph G_p and two interacting terrain-like surface subgraphs G_{S_1} and G_{S_2}

Figure 4.5: Segmenting interacting surface and enforcing surface-object interaction prior. (a) demonstrates the surface segmentation graph G_{S_1} for segmenting surface S_1. Black arcs are intra-column arcs enforcing the terrain-like topology of the surface. Brown arcs are inter-column arcs enforcing a smoothness constraint of 2 voxels between (x, y)-column and $(x + 1, y)$-column. (b) demonstrates the surface-object interaction arcs A_{PS_1} enforcing a minimum surface-object distance of 1 voxel. (c) shows how the pixel layer subgraph G_P (for segmenting the target object), interacts with two terrain-like surface segmentation subgraphs G_{S_1} and G_{S_2}.

1). intra-column arcs which enforce the monotonicity/terrain-like topology of the surface. Infinite weighted arcs are added from $v^S_{p(x,y,z+1)}$ to $v^S_{q(x,y,z)}$ for $0 \leq z < Z-1$ (black arcs in Fig.4.5a). These arcs make sure that all source set vertices must lie *below* all sink set vertices within each column.

2). inter-column arcs which enforce the smoothness constraint between adjacent (x, y)-column and (x', y')-column. Infinite weighted arcs from $v^S_{p(x,y,z)}$ to $v^S_{q(x',y',\max(0,z-\Delta))}$ for all $0 \leq z \leq Z-1$, where Δ is the smoothness constraint between (x, y)-column and (x', y')-column (either Δ_x or Δ_y, depending on the neighboring relationship between (x, y) and (x', y')). Brown arcs in Fig.4.5a enforces the smoothness constraint that $|S(x, y) - S(x + 1, y)| \leq \Delta_x = 2$. If $|S(x, y) - S(x + 1, y)| > \Delta_x$, then at least one of these inter-column arcs will be in the s-t cut. But all inter-column arcs have infinite weight, thus such an s-t cut cannot be an optimal solution due to its infinite s-t cut capacity.

3). source-sink arcs which encode the on-surface cost of all pixels. For pixel $p(x, y, z), 0 < z \leq Z-1$, we add arc from source s to v^S_p with weight $\max(-c_{p(x,y,z)} + c_{p(x,y,z-1)}, 0)$, and an arc from v^S_p to sink t with weight $\max(c_{p(x,y,z)} - c_{p(x,y,z-1)}, 0)$. For every bottom pixel $p(x, y, 0)$, an arc from source s to v^S_p is added with infinite weight to make sure the bottom pixel at each (x, y) column is in the source set. Otherwise, we may result in one column whose vertices are all in the sink set, which is a trivial solution that should be avoided. Suppose $(C, \overline{C})$ is one s-t cut of G_S with C as the source set, and $w_p = c_{p(x,y,z)} - c_{p(x,y,z-1)}$. Then the arcs $(s, v^S_p), v^S_p \in \overline{C}$ and $(v^S_p, t), v^S_p \in C$ are in the cut. The s-t cut capacity is

$$
\begin{aligned}
& -\sum_{\substack{w_p<0,\\ v_p^S\in\overline{C}}} w_p + \sum_{\substack{w_p\geq 0,\\ v_p^S\in\overline{C}}} 0 + \sum_{\substack{w_p<0,\\ v_p^S\in C}} 0 + \sum_{\substack{w_p\geq 0,\\ v_p^S\in C}} w_p && (4.8)\\
=& -\sum_{\substack{w_p<0,\\ v_p^S\in\overline{C}}} w_p + (-\sum_{\substack{w_p<0,\\ v_p^S\in C}} w_p + \sum_{\substack{w_p<0,\\ v_p^S\in C}} w_p) + \sum_{\substack{w_p\geq 0,\\ v_p^S\in C}} w_p && (4.9)\\
=& -\sum_{w_p<0} w_p + \sum_{v_p^S\in C} w_p && (4.10)
\end{aligned}
$$

Note the first term $-\sum_{w_p<0} w_p$ is a constant for all s-t cuts. The second term $\sum_{v_p^S\in C} w_p = \sum_{p\in S} c_p$, which is the sum of original on-surface costs. Thus, we have encoded the on-surface cost correctly with the s-t cut capacity.

Fig.4.5a shows how the surface segmentation subgraph G_S is constructed. The source-sink arcs are omitted to avoid clutter.

4.2.4.3 Graph construction for surface-object interaction

To enforce the surface-object interaction term (Eq.(4.7)), a set of arcs A_{PS} is added between the pixel-wise object segmentation subgraph G_P and each interacting surface segmentation subgraph $G_S, S \in \mathcal{S}$. Fig.4.5c shows how A_{PS} is added when the target object is interacting with two terrain-like surfaces.

For every pixel $p(x, y, z), 0 \leq z < Z - d_{min}$, we add an arc from $v_p \in G_P$ to $v_q^S \in G_S$ with $q(x, y, \max(0, z + d_{min}))$. For every pixel $p(x, y, z), z \geq Z - d_{min}$, we add an arc from $v_p \in G_P$ to sink vertex t. In both cases, the arc carries weight $\Phi_p^S(f_p = 1)$. Fig.4.5b shows how A_{PS} is constructed.

To verify that the construction correctly encodes the surface-object interaction, one can check that the arc $(v_{p(x,y,z)}, v^S_{q(x,y,z+d_{min})})$ is in the s-t cut if there exists some object voxel violating the minimum surface-object distance constraint ($p(x, y, z)$, $f_p = 1$ with $z > S(x, y) - d_{min}$). Because $v_{p(x,y,z)}$ is in the source set since $f_p = 1$, and $v^S_{q(x,y,z+d_{min})}$ is in the sink set since $S(x, y) < z + d_{min}$ and all vertices above $S(x, y)$ belongs to the sink set. In this case, the penalty $\Phi^S_p(f_p = 1)$ is correctly enforced.

Note the pixel layer subgraph G_S is invariant to the topology of target object. Thus, we are free to flip and rotate the image to reduce every surface-object interaction situation to the case where the object is below the z-terrain-like surface with at least d_{min} distance. This preprocessing does not cause conflict when we have multiple interacting surfaces with different directions.

4.3 Experiment

We assessed the performance of the proposed method on the application of primary lung tumor segmentation in Mega-Voltage Cone-Beam CT (MVCBCT). This application is challenging because of the poor image quality (high noise), similar intensity profiles of tumor and normal tissue, and the adjacency of tumor and lung boundary. This increases the likelihood of tumor segmentation leaking into tissues outside lung(see Fig.4.9e for a segmentation by the traditional graph-cut method).

4.3.1 Data

Thirty-eight volumetric MVCBCT datasets gathered under IRB approval (IRB #200707726) at the University of Iowa were used to evaluate the performance of the proposed method. The datasets were acquired from three patients with non-small cell lung cancer over eight weeks of radiation therapy. To be more specific, a total of 20 MVCBCT scans were obtained. Each scan contained one full-inhalation phase and one full-exhalation phase volumetric image. Two out of the 40 volumetric images were rejected by experts due to poor image quality prior to any work proposed here. Our experiment was conducted on the remaining 38 volumetric images. Each image contained $128 \times 128 \times 128$ voxels with voxel size of $1.07 \times 1.07 \times 1.07 \times\ mm^3$. Manual tracings of lung tumors were obtained from experts and were used as the reference standard when assessing the performance of the proposed approach.

4.3.2 Experiment settings

The proposed approach advocates two novel features: incorporating surface-object interaction priors, and using multi-scale information instead of just pixel-wise information. To assess the contributions to the performance of each feature, four groups of experiments were conducted: 1). Traditional graph-cut segmentation (denoted as 'GC'); 2). Graph-cut segmentation with surface-object interaction but without the multi-scale information (denoted as 'TS'); 3). Multi-scale graph-cut segmentation without surface-object interaction (denoted as 'MS'); 4). Multi-scale segmentation with surface-object interaction (denoted as 'TS+MS').

The segmentation performance was assessed using two metrics: Dice similarity coefficient (DSC) and the average symmetric surface distance (ASSD). The *Dice similarity coefficient* is used to measure how well two volumes overlap with each other.

Assume A and B are two volumes, then the DSC between the two volumes is defined as $2|A \cap B|/(|A| + |B|)$, which ranges between 0 and 1. The larger the DSC is, the better the two volumes are aligned, with 1 indicating a perfect overlapping.

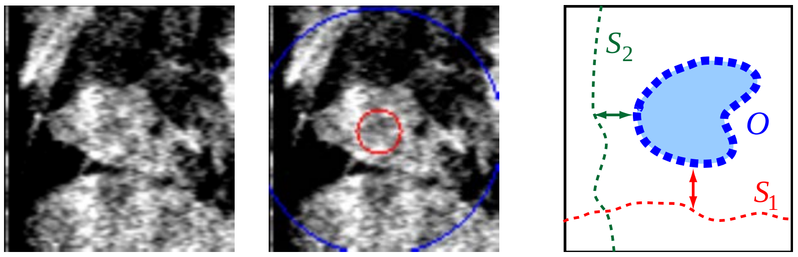
The *average symmetric surface distance* (ASSD) is used to measure how close two segmented surfaces are. Let $d(x, A)$ denote the shortest distance between a point x and any point on the surface A. The ASSD between the segmented boundary surfaces A and B by two different methods is defined in Eq.(4.11). It measures the average distance from any point on a contour/surface to the other contour/surface. The ASSD metric has a range of $[0, +\infty]$. The smaller ASSD is, the better the two segmented surfaces/contours agree with each other. If $ASSD = 0$, then the two segmentations are identical.

$$ASSD = \frac{\sum_{a \in A} d(a, B) + \sum_{b \in B} d(b, A)}{|A| + |B|} \tag{4.11}$$

We report the DSC's and ASSD's between the manually traced tumor contours and the segmentations returned by GC, TS, MS and TS+MS. A two-tailed student t-test was conducted between every pair of methods. A p-value smaller than 0.05 is considered statistically significantly different.

4.3.3 Initialization

We used the same initialization approach used in [85]. The user manually specified two concentric spheres, such that all pixels within the small sphere belongs to the object, and all pixels outside the large sphere belongs to the background. The



(a) input image (b) initialization (c) interacting surfaces

Figure 4.6: Manual initialization. Two inputs are required: two spheres serving as object and background seeds (as in (b), and the surface-object interaction relationship (as in (c). The surface-object interaction relation includes the number of interacting surfaces, the direction of surfaces and whether the surface is above or below the object. No manual contour is required for neither object or interacting surfaces, since all of them will be simultaneously segmented by the proposed method.

segmentation was then conducted in the bounding box of the large sphere. Fig.4.6b shows one such example.

The interacting surfaces to be segmented anatomically corresponds to the lung boundary. We only require the user to specify the surface-object interaction relationship, including the number of interacting surfaces, the direction of surfaces, and whether the surface is above or below the object (Fig.4.6c). The surface-object interaction relationship is determined by visually inspecting the relative proximity of the tumor and lung boundary. No surface contours need to be manually drawn, since they will be simultaneously segmented by the proposed algorithm. The number of terrain-like surfaces segmented by 'TS' and 'TS+MS' methods ranges from two to three. The same set of surfaces are segmented by 'TS' and 'TS+MS' methods on each dataset.

4.3.4 Energy term design and parameter settings

The pixel layer data term is designed as follows. For pixels outside the large sphere, $D_p(f_p = 1) = \infty, D_p(f_p = 0) = 0$ to enforce the hard constraint that they are all background pixels. For pixels inside the small sphere, $D_p(f_p = 1) = 0, D_p(f_p = 0) = \infty$ to enforce the hard constraint that all pixels within the small sphere are object pixels. For pixels between the two spheres, we first estimate the mean object pixel intensity $\bar{I}_{ob}$ and the standard deviation of object pixel intensities $\bar{\sigma}_{ob}$ from all pixels within the small sphere. Assuming the object pixel intensities can be described by a Gaussian distribution, we compute $D_p(f_p = 1) = -\log \Pr(I_p|f_p = 1) \propto \exp(\|I_p - \bar{I}_{ob}\|^2/2\bar{\sigma}_{ob}^2)$, and $D_p(f_p = 0) = -\log(1 - \Pr(I_p|f_p = 1)) = -\log(1 - \exp(-D_p(f_p = 1)))$.

The pixel layer smoothness term is designed as follows. $V_{pq}(f_p, f_q) = w_{pq}^P$ if $f_p \neq f_q$, in which $w_{pq}^P \propto \exp(-\|I_p - I_q\|^2/\sigma_P^2)$. This smoothness term is small when neighboring pixels have very different intensities, which indicates an edge between two pixels.

We applied the watershed method to generate the over-segmentation ([76]). Two regions are defined to be neighboring if the Euclidean distance between centroids of the two regions is within 10 mm. The region layer data term is computed as the mean value of its children pixel data term. $D_r(f_r = 1) = \sum_{p\in r} D_p(f_p = 1)/|r|$ and $D_r(f_r = 0) = \sum_{p\in r} D_p(f_p = 0)/|r|$. The region layer smoothness term is defined similar to the pixel smoothness term: $V_{r_1,r_2}(f_{r_1}, f_{r_2}) = w_{r_1,r_2}^R$, in which $w_{r_1,r_2}^R \propto \exp(\|D_{r_1}(f_{r_1} = 1) - D_{r_2}(f_{r_2} = 1))\|^2/\sigma_R^2$.

The variance parameters for the pixel and region smoothness terms are empirically set to $\sigma_P = 7, \sigma_R = 1000$. The weighting coefficients for the pixel layer term, the region layer term and the surface segmentation term are set as follows: $\eta^P = 1$, $\eta^R = 0.4$, $\eta^S = 1000$ for all surfaces. The surface smoothness constraint for all surfaces are set to be 1 voxel along all direction. The minimum distance between the target object and all surfaces are set to be $d_{min} = 0$, i.e., the object does not cross any of the auxiliary surfaces. All datasets share the same parameter setting.

4.3.5 Results

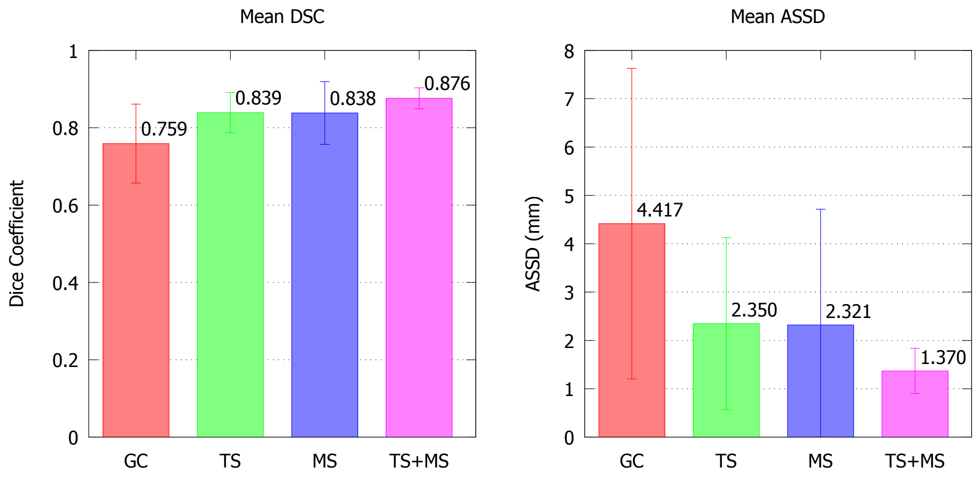


Figure 4.7: Quantitative DSC and ASSD results of four methods: Graph-cut without interacting surface segmentation (GC), Graph-cut with terrain-surface segmentation (TS), multi-scale segmentation without surface segmentation (MS), and multi-scale segmentation with terrain-surface segmentation (TS+MS). Mean value and standard deviation are shown in the figure.

The quantitative results are summarized in Fig.4.7. By using multi-scale segmentation alone (MS) or incorporating surface-object interaction priors alone (TS), we were able to achieve similar accuracy improvement over using traditional graph-cut (GC). By combining the multi-scale method with the interacting surfaces segmenta-

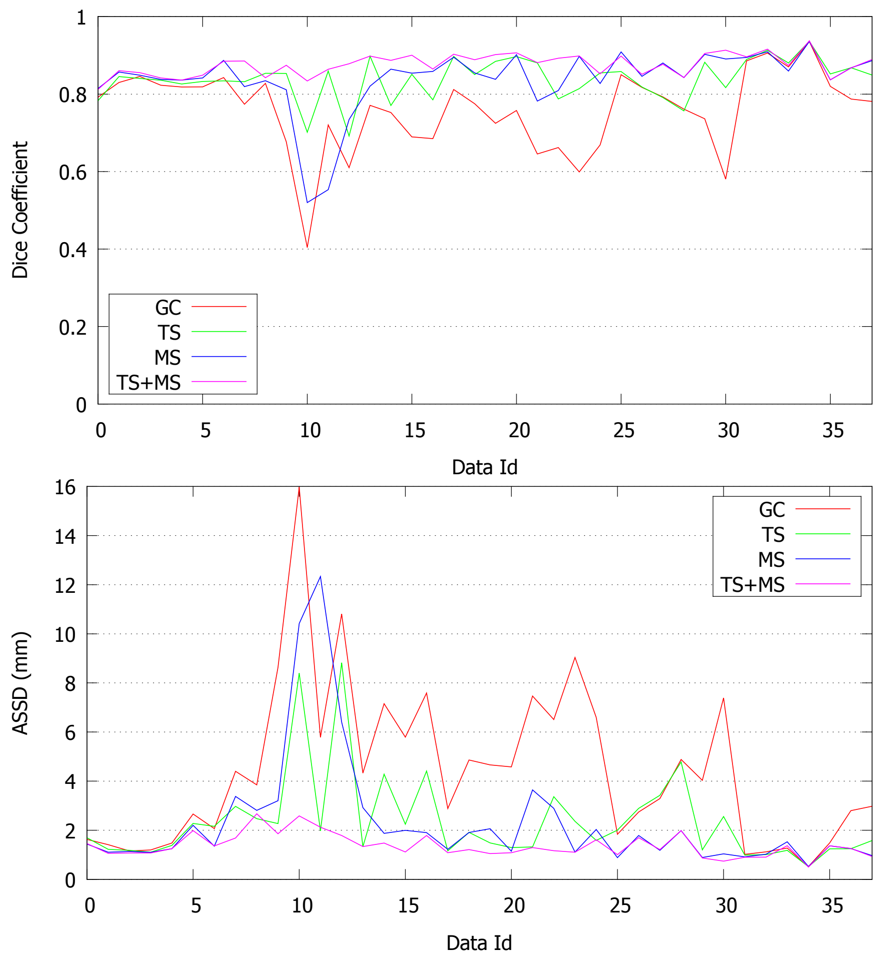


Figure 4.8: Quantitative DSC and ASSD results (dataset by dataset). (Best view in color.) We can see that MS+TS achieves consistently accurate result among all datasets, which utilizes both surface-object interaction prior and the multi-scale information of the target object.

tion (TS+MS), the segmentation accuracy was further improved.

The t-tests on DSC's show that: a). the difference between TS and GC, and the difference between MS and GC are both statistically significant; b). the difference between TS+MS and MS, and the difference between TS+MS and TS are both statistically significant. The t-test on ASSD shows the same statistical significance result as DSC.

This shows that the segmentation accuracy is improved by incorporating either surface-object interaction prior or multi-scale information of the target object. The best segmentation accuracy is achieved when utilizing both the surface-object interaction and multi-scale information.

The dataset-by-dataset quantitative results are given in Fig.4.8. More consistent and accurate segmentation is achieved by incorporating the surface-object interaction prior and the multi-scale information of the target object.

Qualitative results are shown in Fig.4.9. Due to the weak boundary and similar intensity profiles of tumor and surrounding tissues, the traditional graph-cut method produced unsatisfying segmentation (Fig.4.9e). By simultaneously segmenting two terrain-like surfaces together with the tumor, we were able to prevent a large amount of segmentation leakage (Fig.4.9f). But there was still some leakage on the right part of the image. Note there is no clear terrain-like boundary surface on the right side of the tumor. Thus, the leakage problem in Fig.4.9f may not be resolved by incorporating an extra terrain-like surface. By using multi-scale segmentation (MS), we are able to take advantage of the long range neighborhood in the region layer, which helps differentiate tumors from lung tissues. We can see the segmentation in Fig.4.9g is much better than that in Fig.4.9e. But there is still some leakage at the left part in Fig.4.9g. By incorporating the surface-object interaction priors (TS+MS), the leakage problem in Fig.4.9g is prevented and the segmentation aligns well with manual contour (Fig.4.9h).

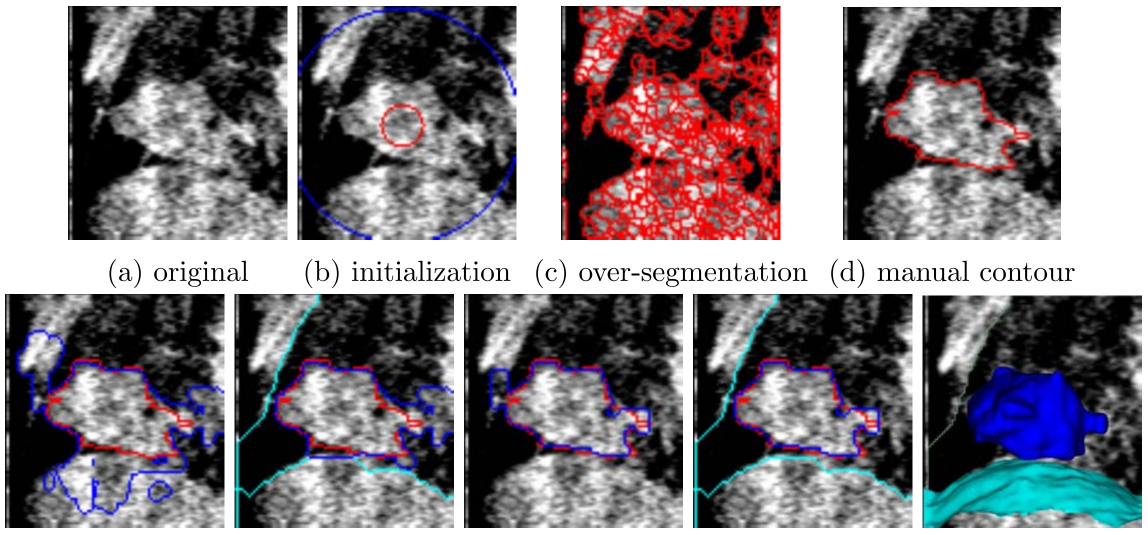
The visualization of a tumor segmentation and the two interacting surfaces segmentation are shown in Fig.4.9i. Note one of the two surfaces is only shown as a 2D contour, since this surface blocks the view of the tumor if visualized as a 3D surface.

Generating the segmentation for each dataset requires 9 minutes on average. This includes 2 minutes for user to specify the two spheres as discussed in Sec.4.3.3, and 7 minutes for the algorithm to run on a Linux workstation (2.8GHz, 256GB memory).

4.4 Discussion

4.4.1 Novelty of the proposed method

[85] enforced the surface-object interaction along the z-direction (both above and below the z-terrain like surface). In this work, we generalize our previous segmentation method ([85]) to handle multiple terrain-like surfaces along different directions and further incorporate the multi-scale segmentation scheme into a single optimization process. Our method can achieve a globally optimal segmentation solution with respect to the objective function in low-order polynomial time by solving a maximum



(a) original (b) initialization (c) over-segmentation (d) manual contour

(e) GC (f) TS (g) MS (h) TS+MS (i) TS+MS 3D

Figure 4.9: Illustrative result. Red contours in (d), (e), (f), (g), (h) and (i) are manual tracings by an expert, i.e., the reference standard. Blue contours in (e), (f), (g), (h) and (i) are the semi-automatic segmentations generated by different methods. As shown in (f), incorporating surface-object interaction prior information prevents lots of "leakage" as in (e). Further including multi-scale information achieves even more accurate segmentation, as in (h), (i).

flow problem. Our approach results in a novel integration of the traditional graph-cut method [12] and the graph searching method ([52]).

[44] studied the multi-scale segmentation problem and also modeled it as a graph representation with two layered subgraphs – a region layer and a voxel layer, as in our proposed method (Sec.4.2.1). However, our method differs from their work in the following aspects. First, the region layer in [44] is fully connected. This is feasible for 2D images, but is computationally prohibitive for 3D images, which may easily contain over 10,000 segments in an over-segmentation. On the contrary, our algorithm only connects nodes within a limited neighborhood. Second, the approach in [44] needs to solve a random walk with restart (RWR) problem. To solve it exactly, either a quadratic space and cubic pre-computation time, or a slow response time on queries is needed ([94]). The optimization approach cannot guarantee the global optimality. In contrast, our approach reduces the problem to a maximum flow problem in a sparse graph with only $O(M|V|)$ arcs, where M is the maximum number of neighboring regions for any region $r \in \mathcal{R}$. This problem can be solved efficiently in $O(M|V|^2)$ time with a globally optimal solution ([65]).

4.4.2 Importance of simultaneous multi-scale segmentation

The benefits of including the region layer is two-folds: first, the region layer includes some aggregate information which is not obvious/available in the pixel layer. For example, if the pixel intensity changes gradually in the pixel layer, then it's hard to locate a clear edge. But the mean intensity of two adjacent over-segmented regions may reveal the gradual intensity change trend more easily.

Second, the region layer helps propagate long-range information. Obviously children pixels of the same region are connected to the same region vertex in the graph, even if they are not neighbors according to the neighborhood setting in the 3D image. Furthermore, the region vertices are also connected to each other within a relatively large neighborhood. This helps further propagate local information to far-away locations, which helps discriminate the target more easily.

In contrary to most multi-scale techniques which process different scales sequentially ([51, 59, 77]), our method processes the coarse level and the fine level simultaneously. While processing different scales sequentially can reduce running time and memory consumption, we cannot expect the segmentation accuracy to improve in general. Processing different scales simultaneously, on the other hand, can achieve better accuracy by allowing coarse level information and fine level information to propagate in both directions ([43, 21]).

4.4.3 Limitations

One limitation of our method is that the running time and memory consumption is increased due to the *simultaneous* processing of two object segmentation subgraphs at different scales. Instead of sequentially processing the coarse level and fine level for object segmentation, we process two levels simultaneously. Although the segmentation accuracy is improved, this also makes the graph size larger and the running time longer.

4.5 Conclusion

We proposed a novel segmentation method which simultaneously segments topologically flexible object and multiple interacting adjacent surfaces with know topologies. The topologically flexible object is segmented via graph cut ([12]). While the interacting surfaces with known topologies are segmented via graph searching ([52]). A multi-scale approach is also incorporated to boost the target object segmentation performance. Region-wise information from a data-driven over-segmentation of the image is utilized in the multi-scale segmentation. The problem is reduced to an MRF energy minimization problem, which could be efficiently and exactly solved by computing a single minimum *s*-*t* cut in an appropriately constructed graph. The proposed method's performance is assessed on 38 MVCBCT datasets to segment primary lung tumors. By incorporating the surface-object interaction prior and the multi-scale information, the proposed method segments lung tumors more accurately and robustly in the challenging datasets.

CHAPTER 5
4D CT-PET CO-SEGMENTATION

Four-dimensional CT scans provides valuable motion information of patient throughout different respiratory phases. PET, on the other hand, provides functional information about tumor, which differentiate tumor from normal tissue effectively. However, manually contouring structures of interest on 4D CT is prohibitively tedious due to the large amount of data. We propose an automatic method to segment lung tumor simultaneously for 4D CT scans in all phases and PET scan. The problem is modeled as an optimization problem based on Markov Random Fields (MRF) which involves region, boundary terms and a regularization term between PET and CT scans. The problem is solved optimally by computing a single max flow in a properly constructed graph. As far as the authors know, this is the first work in simultaneously segmenting tumor in 4D CT while incorporating PET information. Experiments on 3 lung cancer patients are conducted. The average Dice coefficient is improved from 0.680 to 0.791 compared to segmenting tumor volume in 4D CT phase by phase without incorporating PET information. The proposed method is efficient in terms of running time since the method only requires computing a max flow for which efficient algorithm exists. The memory consumption is linearly scalable with respect to number of 4D CT phases, which enables our method to handle multiple 4D CT phases with reasonable memory consumption.

5.1 Introduction

Four-dimensional CT scans are acquired by obtaining multiple CT scans at different stage/phase of patient respiratory cycle. The respiratory cycle is divided into a number of breathing phases–one phase at the end of inspiration, another phase at the end of expiration and several evenly distributed phases in between. Compared to traditional free-breathing CT scan, 4D CT scans provide valuable motion information which could help reduce late complications resulting from overdoes to critical organs and deliver higher level of does to targets to improve tumor control when taken into account of treatment planning[71].

Although the prohibitive huge amount of data in 4D CT made its automatic segmentation urget, few literatures were reported in this area. Weiss *et al.* investigated temporospatial variations of tumor and normal tissue during inspiration in lung cancer patients[103]. But they used manually defined contours. You *et al.* proposed a semi-automatic segmentation method which propagates the contour which is defined on a 3D CT scan to all the other phases by using principal surfaces[111]. Ehrhardt *et al.* presents a variational approach for simultaneous segmentation and registration applied to temporal image sequences[27]. The variational approach uses numerical method to minimize a cost function that combines intensity-based registration, level-set segmentation as well as prior shape and intensity knowledge. But the numerical method may suffer from roundoff errors and may not converge to the global minimum. All the automatic methods above did not incorporate PET information.

Co-segmentation of PET-CT pairs recently attracts attention among researchers.

Han *et al.* presents a globally optimal tumor segmentation in PET-CT images[36]. They formulate the problem as a Markov Random Field (MRF) based segmentation with a regularization term that penalizes the segmentation difference between PET and CT. The solution is globally optimal and can be obtained in low-order polynomial time by computing a single max flow. But their paper only illustrates their co-segmentation method on PET and single CT scan.

5.2 Methods

Our method borrows the idea from Han's method[36]. First PET and the CT in one specific phase is registered. Then we build a subgraph for PET and CT in each phase. A context term is introduced in the graph-cut framework which penalizes the segmentation difference between PET and CT. By solving a single max-flow in this graph, we can optimally obtain different segmentation for PET and CT in each phase in terms of a MRF energy function. The regularization term makes sure the CT segmentation result in each phase incorporates information from the PET and vice, versa.

5.2.1 Registration between PET and CT

A registration of PET and CT in a specific phase is performed. The PET is upsampled using a cubic B-spline interpolation so that the PET will have the same resolution as CT. The idea is that the registered PET and CT should have a one-to-one spatial correspondence for each voxel pair. The phase of CT on which the registration is performed should be more or less the "average" of all the CT phases in terms of tumor location and shape. No registration between different phases of CT is performed. Instead of relying on fusion of CT images, we rely on our method's ability to generate different segmentations for PET and CT to account for the motion of tumor in different respiratory phases.

5.2.2 Problem formulation

The segmentation problem is formulated as a Markov Random Field (MRF) problem described in Eq.(5.1). Minimizing the energy will return the segmentation result.

$$\mathcal{E}_{PET-CT} = E_P(\mathbf{f}_P) + \sum_{i=1}^{K} E_C^i(\mathbf{f}_C^i) + \sum_{i=1}^{K} E_{P-C}^i(\mathbf{f}_P, \mathbf{f}_C^i) \tag{5.1}$$

K is the number of phases contained in 4D CT. $\mathbf{f}_P$ is the vector of binary labeling ('object' vs. 'background') of PET pixels, which defines the segmentation of PET image. $\mathbf{f}_C^i, 1 \leq i \leq K$ is the vector of binary labeling of pixels in the i-th CT image, which defines the segmentation of CT in the i-th phase. $E_P(\mathbf{f}_P)$ is defined as a typical graph-cut energy function consisting of region and boundary term for PET image.

$$E_P(\mathbf{f}_P) = \sum_{u \in \mathcal{I}_P} d_u(f_u) + \sum_{(u,v) \in \mathcal{N}_P} w_{u,v}(f_u, f_v) \tag{5.2}$$

$\mathcal{I}_P$ is the set of all pixels in PET image. $\mathcal{N}_P$ is the set of all neighboring pixel pairs in PET image. $d_u(f_u)$ is the region term for assigning label f_u to pixel u. $w_{u,v}(f_u, f_v)$ is the boundary penalty for assigning label f_u, f_v to neighboring pixels

u, v. Similarly, the energy for the CT scan in the i-th phase $E_C^i(\mathbf{f}_C^i)$ consists of a region term $d_{u'}^i(f_{u'}^i)$ and a boundary term $w_{u',v'}^i(f_{u'}^i, f_{v'}^i)$:

$$E_C^i(\mathbf{f}_C^i) = \sum_{u' \in \mathcal{I}_C^i} d_{u'}^i(f_{u'}^i) + \sum_{(u',v') \in \mathcal{N}_C^i} w_{u',v'}^i(f_{u'}^i, f_{v'}^i) \tag{5.3}$$

The regularization term $E_{P-C}^i(\mathbf{f}_p, \mathbf{f}_C^i)$ is defined to penalize the segmentation difference between PET and CT.

$$E_{P-C}^i(\mathbf{f}_P, \mathbf{f}_C^i) = \sum_{u \in \mathcal{I}_P, u' \in \mathcal{I}_C^i} \gamma_{u,u'}^i(f_u, f_{u'}) \tag{5.4}$$

$$\gamma_{u,u'}^i(f_u, f_{u'}) = \begin{cases} 0 & \text{if } f_u = f_{u'} \\ \varphi_{u,u'} & \text{otherwise} \end{cases} \tag{5.5}$$

in which u and u' are pixels with the same spatial coordinates but in different images. When u and u' are labeled with the same label ($f_u = f_v$), there is no penalty introduced. In this case, the PET and CT information agree with each other for voxel pair u and u'. If the labels for u and u' are different ($f_u \neq f_v$), then the PET and CT information disagrees with each other for voxel pair u and u'. A penalty $\varphi_{u,u'}$ is paid for the difference between labeling of u and u'. This penalty term encourages CT segmentation to agree with the PET segmentation. However, the proposed method does allow different segmentation between PET and CT, as long as the penalty is paid for each differently-labeled voxel pair u and u' in PET and CT.

5.2.3 Graph construction

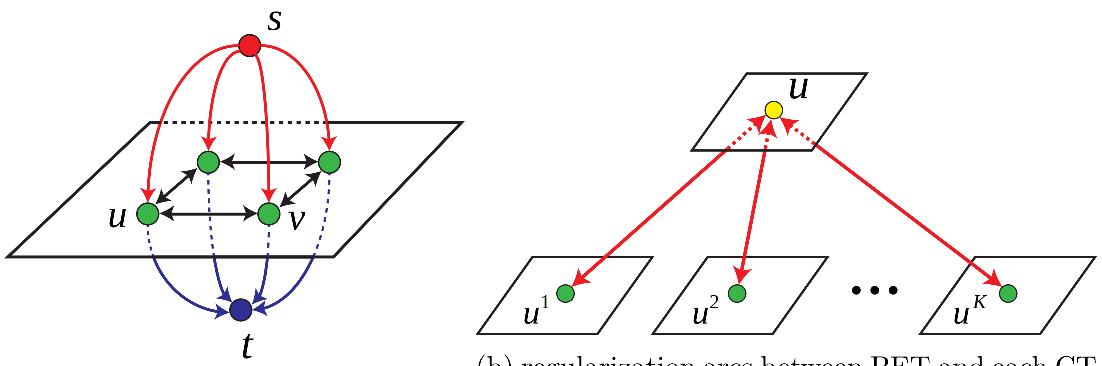
The energy term for PET $E_p(\mathbf{f}_P)$ and each phased CT $E^i_C(\mathbf{f}^i_C), 1 \leq i \leq K$ can be encoded using standard graph-cut framework. To be more specific, for PET scan and each CT scan in different phases, we build a subgraph which itself is built using Boykov's graph-cut framework[12]. Fig.5.1a shows how such a subgraph is built. Assume the PET energy term in Eq.(5.2) is being encoded by a subgraph $G_P = \{V_P \cup \{s,t\}, A_P\}$. For each voxel u in the image, we create a vertex in the graph. Here we abuse the notation to denote the vertex corresponding to voxel u also as vertex u. A dummy source node s and a sink node t are also introduced.

For every vertex $u \in V_P$, we introduce an arc from source s to itself, (s,u) with arc weight $D_u(f_u = `bg')$, which is the region term penalty for assigning voxel $u$ as background. For every vertex $u \in V_P$, we also introduce an arc from itself to sink $t$, $(u,t)$ with arc weight $D_u(f_u = `ob')$, which is the region term for assigning voxel u as object/tumor. The red and blue arcs in Fig.5.1a shows such arcs. For every pair of neighboring voxel u and v, two arcs (u,v) and (v,u) with weights $w_{u,v}(f_u = `ob', f_v = `bg')$ and $w_{u,v}(f_u = `bg', f_v = `ob')$ are introduced to enforce the boundary term in Eq.(5.2). The black double-arrowed arcs in Fig.5.1a show the two arcs for every pair of neighboring voxels. For each CT scan in different phases, such a subgraph is built to encode the CT energy term in Eq.(5.3).

The regularization terms exist between PET image and each CT image in different phases. For every node u in the PET image $\mathcal{I}_P$, an arc is added from u to u^i (u and u^i have the same spatial coordinates) in the CT image $\mathcal{I}^i_C$ for the i-th phase.

An arc is also added from u^i to u. Both arcs carry the same penalizing weight φ_{u,u^i} for the segmentation difference between PET and CT in the i-th phase. Fig.5.1b shows the graph building for the regularization term.

Solving a max flow in the constructed graph will return the optimal segmentation result in terms of the energy function in Eq.(5.1), as described by Boykov *et al.*[12]. Due to the regularization term, the segmentation result of CT in all phases incorporate the information of PET.



(a) subgraph construction

(b) regularization arcs between PET and each CT phase

Figure 5.1: Graph construction for encoding the MRF problem. a shows how the graph is built for PET and each CT phase. The red and blue arcs are used to enforce the region term and the black arcs are used to enforce the boundary term. b shows the regularization arcs between every voxel in PET and its corresponding voxel in each CT in different phase.

5.3 Experiments

5.3.1 Experiment settings

Three PET-4D-CT scan pairs obtained from three different patients with lung cancer are used in the experiment. The free-breathing PET images have 168×168 voxels in each slice. The number of slices varies from 313 to 487. The voxel spacing is 3.39×3.39×2.03 mm for all PET scans. Two 4D-CT datasets consist of 10 phases: 0EX, 20EX, 20IN, 40EX, 40IN, 60EX, 60IN, 80EX, 80IN, 100IN. For these two patients, the number of slices for each phase varies from 141 to 145. One 4D CT dataset contains one extra phase 0IN besides the above 10 phases. For this patient, the number of slices varies from 238 to 244 for each phase. Each CT slice of the 3 patients has 512×512 voxels. The voxel spacing for the 3 patients is 0.98×0.98×2 mm for all CT scans. Contours of tumor in each phase are drawn on the corresponding CT scans by clinical physician.

For the first 2 subject, CT in phase 0%EX is used as reference phase when registering PET. For the third subject, CT in phase 100%IN is used when registering PET. As with every graph-cut based method, object and background seed have to be specified. To relieve the manual labor burden, we use the initialization method used by Song *et al.*[84] The user needs to specify one center voxel and two radii. The sphere defined by the center voxel and the smaller radius specifies the object seed, which means all voxels *within* the sphere must be labeled as 'object' (tumor in our application). The sphere defined by the center voxel and the bigger radius specifies the background seed, which means all voxels *outside* the sphere must be labeled as 'background'. The seed could be specified on PET and CT in every phase. However, for simplicity, in this experiment, we only specify one set of seed and use them on PET and CT in every phase.

Assuming the intensities of the object follows a Gaussian distribution, the region term cost for PET and CT images are calculated by fitting the Gaussian model generated from the object seed voxels[84], which are voxels within the sphere with the smaller radius in the seed initialization. The boundary term is simply the gradient in each image.

5.3.2 Regularization term design

The regularization term $\varphi_{u,u'}$ is generated based on the "similarity" of voxel pairs in PET and CT in each phase. First the region term for PET and CT in each phase are computed as described before. Then all the object region terms within each image is normalized from 0 to 255. The absolute value of difference between the normalized object region term for PET and the normalized object region term for CT in a specific phase is scaled by a linear transformation (Eq.(5.6)) and used as the regularization term. More specifically, assume u and u' are a pair of voxels with the same spatial coordinates in PET and the CT in i-th phase. The normalized object region term for u is $d_u(f_u = \text{`}ob\text{'})$ and the normalized object region term for $u^i$ is $d^i_{u'}(f^i_{u'} = \text{`}ob\text{'})$, then the regularization term is defined as

$$\varphi_{u,u'} = \eta \cdot (255 - |d_u(f_u = \text{`}ob\text{'}) - d^i_{u'}(f^i_{u'} = \text{`}ob\text{'})|) \tag{5.6}$$

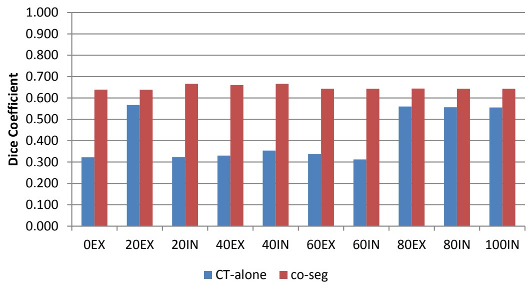
in which η is a constant which is set to 5 in this experiment. The rationale behind this design is that if the object region term for PET and for CT have similar values, then we should pay a high penalty if this voxel pair turns out to have different labels. Because they are similar according to the object region term values, and thus are encouraged to have the same labeling.

5.3.3 Validation

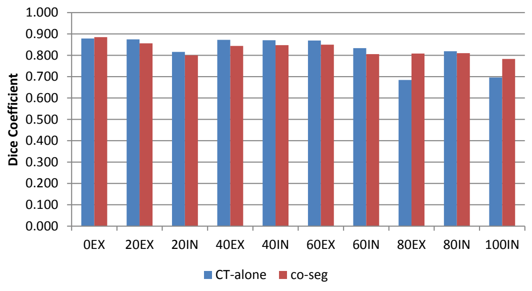
Dice Similarity Coefficient (DSC) is used to validate the proposed approach. DSC between two volumes X and Y are computed according to Eq.(5.7). The segmentation of tumor volume on CT in each phase is compared to the expert manual delineation of the tumor volume on CT in each phase.

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|} \tag{5.7}$$

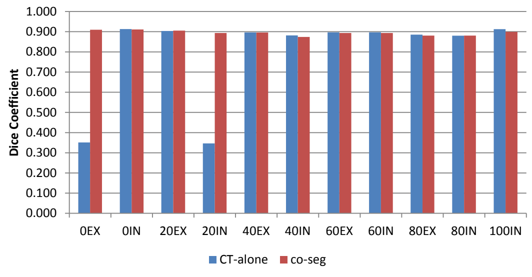
We also present the segmentation results by graph-cut segmentation using CT only, without PET information. Besides the regularization term, the CT-only method has the same setup with the proposed co-segmentation method. DSC for both CT-only and the proposed PET-4D-CT co-segmentation methods are presented and compared.



(a) subject 1



(b) subject 2



(c) subject 3

Figure 5.2: Dice coefficient for 3 subjects in each 4D CT phase.

5.3.4 Results

The DSC for CT segmentation in all phases are shown in Table.5.1. As the bolded numbers indicate, the average Dice coefficient is increased by using both PET and CT information instead of using CT only. Overall, DSC increases from 0.680 to 0.791. Fig.5.2 illustrates the results more intuitively. The DSC for PET-4D-CT co-segmentation is more stable than using CT only. For example, the DSC using CT only for the segmentation of 0EX phase of subject 3 is only 0.351. With the guidance of PET, the DSC is increased to 0.910. Fig.5.3 shows the segmentation of this phase. Using CT only, the segmentation "leaks" into the adjacent normal tissue (Fig.5.3d,5.3h). With the guidance of PET, the segmentation conforms to the tumor volume (Fig.5.3e5.3i).

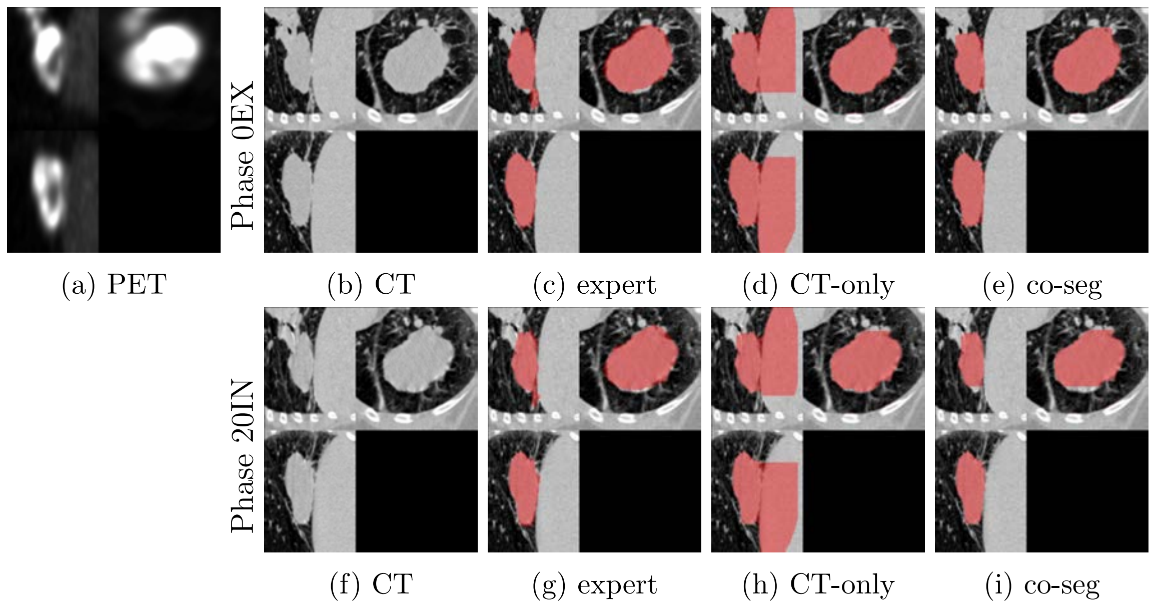


Figure 5.3: Illustrative results of phase 0EX and phase 20IN of subject 3. Note the guidance of PET prevents the segmentation from leaking into the adjacent normal tissues.

Table 5.1: Dice coefficient for each subject.

phase	CT-alone	co-seg
0EX	0.322	0.639
20EX	0.567	0.639
20IN	0.323	0.666
40EX	0.330	0.660
40IN	0.354	0.666
60EX	0.339	0.643
60IN	0.312	0.643
80EX	0.560	0.644
80IN	0.557	0.643
100IN	0.556	0.643
Average	**0.422**	**0.649**

(a). subject 1

phase	CT-alone	co-seg
0EX	0.879	0.885
20EX	0.875	0.856
20IN	0.816	0.800
40EX	0.872	0.844
40IN	0.871	0.847
60EX	0.869	0.850
60IN	0.833	0.805
80EX	0.684	0.808
80IN	0.819	0.810
100IN	0.696	0.783
Average	**0.821**	**0.829**

(b). subject 2

phase	CT-alone	co-seg
0EX	0.351	0.910
0IN	0.913	0.911
20EX	0.903	0.905
20IN	0.346	0.893
40EX	0.896	0.896
40IN	0.881	0.874
60EX	0.897	0.894
60IN	0.897	0.894
80EX	0.885	0.881
80IN	0.880	0.881
100IN	0.913	0.899
average	**0.797**	**0.893**

(c). subject 3

5.4 Discussion

We borrow the idea of incorporating a regularization term to combine PET and CT information in the segmentation from Han's work[36]. However, Han's work only generates one set of contour in spite of the fact that both PET and CT images, from which different parts of the tumor might be shown, are given. For the 4D-CT data, the fact that the tumor volume is different from different phases is especially important. The most significant novelty of our regularization term is that it allows different segmentation in CT and PET. This property is due to the fact that the regularization penalty in Eq.(5.6) is finite and limited. While encouraging the CT segmentation to agree with PET segmentation, the regularization term does allow different segmentations between PET and CT. As long as such difference will result in a smaller overall energy after paying the regularization penalty, the segmentation between PET and CT can be different.

The ability to generate different segmentation between PET and CT explains why the proposed method works even when CT scans in different phases are not registered in this preliminary experiment. The tumor volume difference in different phases caused by the respiratory motion could be dealt with by our method when the difference is within a certain range.

5.5 Conclusion

We propose an automated co-segmentation method which simultaneously segments lung tumor in free-breathing PET and multiple 4D CT phases. The method formulates the problem as a MRF optimization problem with PET/CT region/boundary terms and PET/CT regularization term which penalize the segmentation difference between PET and CT. The optimization problem is solved exactly by computing a single max flow, for which efficient algorithm exists. This method's memory consumption is linearly scalable in terms of number of CT phases, which makes the memory consumption reasonable. As far as the authors know, this is the first work to address the simultaneous segmentation of 4D CT and PET. The experiments show that the proposed method is promising in improving the segmentation accuracy of tumor volume in each respiratory phase.

CHAPTER 6
COMBINING LOCATION AND INTENSITY PET-CT CONTEXT IN TUMOR CO-SEGMENTATION

Tumor segmentation utilizing both PET and CT scans is advantageous since it combines the functional information from PET and the structural information from CT. Although multiple methods have been proposed for tumor co-segmentation, most of them are relying solely on intensity difference between pairs of corresponding PET and CT voxels to regularize the segmentation difference between two different modalities.

We propose to incorporate a novel geometrical location context term into tumor co-segmentation. One the one hand, the proposed location context prior makes co-segmentation more flexible by lowering the penalty for PET-CT segmentation differences around PET uptake region boundaries, i.e., where most PET and CT segmentation differences are expected. On the other hand, it guarantees effective information propagation between PET and CT by maintaining high context prior penalty throughout other part of the image.

The proposed context term is validated on 44 PET-CT pairs from non-small cell lung cancer patients. By incorporating the proposed location context term, the Dice coefficient is improved from 0.77±0.10 to 0.79±0.09, and average symmetric surface distance error is improved from 2.36±1.38 mm to 2.14±1.29 mm.

6.1 Introduction

6.1.1 Co-segmentation and intensity context

We roughly follow the co-segmentation MRF formulation in Song *et al.* [83]. Suppose $\mathcal{P}$ and $\mathcal{P}'$ are the set of CT and PET voxels, respectively, and $p \in \mathcal{P}$ and $p' \in \mathcal{P}'$ are a pair of CT and PET voxels with the same physical coordinate. Then the MRF formulation aims to minimize the following energy function.

$$E(\mathbf{x}) = \sum_{p \in \mathcal{P}} E^{CT}(x_p) + \sum_{p' \in \mathcal{P}'} E^{PET}(x_{p'}) + \sum_{(p,p')} C_{p,p'}[x_p \neq x_{p'}] \tag{6.1}$$

where $x_p, x_{p'}$ are variables representing segmentation results in CT and PET images. Eq.(6.1) consists of segmentation terms for CT and PET, and the context term $C_{p,p'}$. The two CT and PET segmentation terms tends to generate segmentations that follow individual image information. The context term penalizes the difference between the two segmentations. For modality $\Omega \in \{CT, PET\}$, the corresponding segmentation term is defined in Eq.(6.2).

$$\sum_{q \in \mathcal{P}^{\Omega}} E^{\Omega}(x) = \sum_{q \in \mathcal{P}^{\Omega}} D^{\Omega}(x_q) + \sum_{q_1, q_2 \text{ are neighbors}} V^{\Omega}_{q_1 q_2}(x_{q_1}, x_{q_2}) \tag{6.2}$$

where D^{Ω} is a region term reflecting how likely a voxel belongs to object/background, and $V^{\Omega}_{q_1 q_2}$ is a boundary term reflecting how likely an edge lies between neighboring voxels q_1 and q_2.

Suppose D^{CT}_p and $D^{PET}_{p'}$ are normalized region terms for CT voxel p and PET voxel p' that share the same physical coordinate, the intensity context term is defined as

$$C^{int}_{p,p'} = \theta \cdot (1 - |D^{CT}_{p} - D^{PET}_{p'}|) + K \tag{6.3}$$

The more similar CT and PET region terms are, the larger the intensity context term is. A base penalty K is enforced no matter what the difference is.

6.1.2 Location context

However, it is easy to see geometric location information also provides highly useful information. For example, the core of highly metabolically active region is likely to be tumor according to PET. At these locations, we would like to propagate this information to CT so that the same CT region is also labeled as tumor. On the other hand, at region far away from functionally active region, we would like CT to follow the hint and label the region as background. At regions in between, especially around the tumor boundaries, a smaller PET-CT context term is desirable to allow different CT and PET segmentation boundaries.

In another word, the context term should be high at regions either inside, or far away from the PET highlighting regions. Because the PET information in these regions are reliable and worth to be propagated to CT. On the other hand, the context term should be low around PET highlighting region boundaries. Because we expect different tumor segmentation boundaries from CT and PET scans around here.

To measure the distance to metabolically active region, the raw PET intensities is first converted into standard uptake values (SUV) [9]. We then get a segmentation S^{SUV} by following the clinical practice of using SUV threshold values of 2.5 or $40\% SUV_{max}$ to segment tumor region based solely from PET [63]. Then we compute the distance map of S^{SUV} so that every voxel p has a distance $dist(p) = \min_{q \in S^{SUV}} dist(p,q)$. The proposed location context term $C^{loc}_{p,p'}$ is then defined as a truncated quadratic function (Eq.(6.4)).

$$C^{loc}_{p,p'} = \min(1, a(dist(p) - b)^2 + c) \tag{6.4}$$

Parameters a, b, c define a quadratic function taking $dist(p)$ as input. Parameter b determines at which distance we have a minimum location context. Parameter c gives a base location context penalty, similar to base intensity context penalty K in Eq.(6.3). Parameter a controls how wide the parabola curve should be.

6.1.3 Combining intensity and location contexts

We multiply the location context term with the traditional intensity context term as the overal context prior (Eq.(6.5)). This way, the location-sensitive behavior we desired can be enforced regardless of the intensity context. This is also why we choose multiplication, instead of addition, to combine the two terms.

$$C_{p,p'} = C^{int}_{p,p'} \cdot C^{loc}_{p,p'} \tag{6.5}$$

Fig.6.1 shows the overall context term generation process. On the first row of Fig.6.1, the intensity context term is computed from the intensity difference of CT and PET region terms. On the second row of Fig.6.1, the location context is computed from

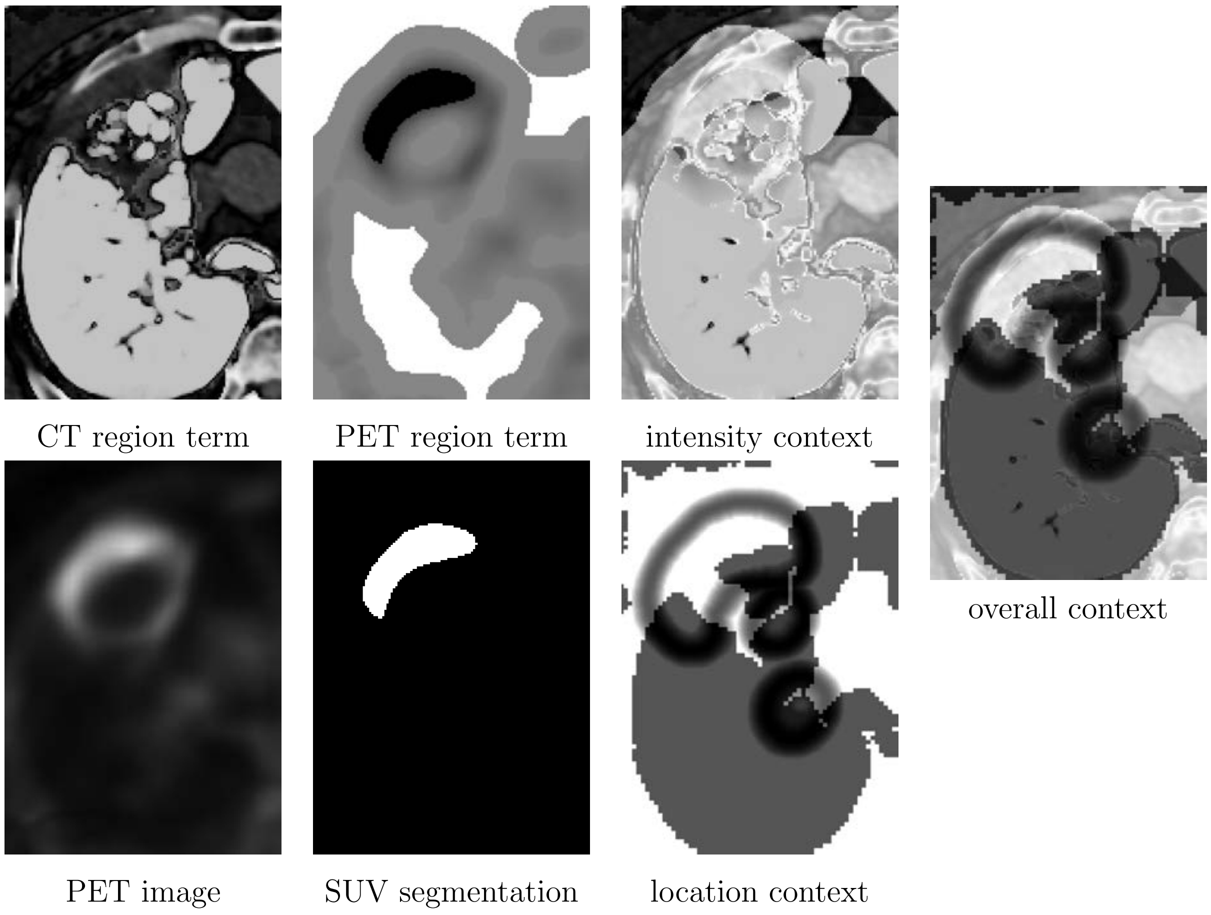


Figure 6.1: Combine intensity and location contexts as the overall context term.

a SUV highlighting region segmentation from PET scan. Note the proposed location context is lowest at a certain distance from the SUV PET segmentation (controlled by parameter b in Eq.(6.4)). This is the region where more differences are allowed between CT and PET segmentation. A rough lung segmentation is also used to further attenuate the location context term within lung area, due to the more structural boundary information from CT within the lung area.

The two context terms are multiplied to obtain the overall context term (last column in Fig.6.1). The new combined context term, which includes both location and intensity information, is used in the co-segmentation.

6.2 Experiment

6.2.1 Data

A total of 54 PET-CT scan pairs from different patients with primary non-small cell lung cancer are obtained. The image spacing varies from $0.78 \times 0.78 \times 2mm^3$ to $1.27 \times 1.27 \times 3.4mm^3$. The intra slice image size is 512×512. The number of slices varies from 112 to 293. Manual expert tracings of the primary tumor volume is available for each scan.

The 54 scans are separated disjointly as a 10-scan training set and 44-scan testing set. The selection procedure starts by sorting all scans according to tumor volume. Then one scan out of every five scan is selected as training scan. This stratified strategy makes sure the training set is representative of the whole population in terms of tumor volume. All parameters are tuned on the training set. All reported results are from the testing set.

6.2.2 Experiment settings

The same initialization procedure in [83] is employed. The user first specify two concentric spheres with the different radii to serve as object and background seeds. More specifically, all voxels inside the smaller sphere are used as object seed. All voxels outside the larger sphere are used as background seed. To compute highlighting SUV regions, we use $\max(2.5, 40\% SUV_{max})$ as the threshold to generate the SUV highlighting region segmentation S^{SUV} [63], where SUV_{max} is the maximum values in one SUV uptake region.

The co-segmentation is then run using only the intensity context term, and using the combined context term with both location and intensity information. The segmentation accuracy are measured by the Dice coefficient (DSC) and average symmetric surface distance (ASSD). Dice coefficient measures the volume overlap of two segmentations A and B. It is defined as $2|A \cap B|/(|A| + |B|)$, with a range of [0,1]. The higher the DSC is, the better volume overlap the two segmentations have. Average symmetric surface distance measures the surface positioning error between the boundary surface of two segmentations. Suppose G and H are the voxels on the boundary surfaces of segmentations A and B, respectively, then ASSD is defined as $(\sum_{g \in G} \min_{h \in H} dist(g, h) + \sum_{h \in H} \min_{g \in G} dist(g, h))/(|G| + |H|)$. In another word, it is the average distance of every surface voxel of one segmentation to the other. ASSD has a range of $[0, +\infty)$, with a smaller value correspond to better segmentation alignment.

6.2.3 Parameter selection

A grid search strategy is used to select the parameters in both types of context terms. In the experiment using only the intensity context, a range of parameters θ and K are generated. All combinations of the two parameters are tried on the training set. The parameters returning highest training set DSC is used to run the co-segmentation on the test set.

In the experiment using both intensity and location contexts, A range of candidate values for each of (θ, K, a, b, c) are generated. All combinations are tried on the training set. The parameters returning highest training set DSC is used to run the co-segmentation on the test set.

6.2.4 Results

Fig.6.2 illustrate two examples of co-segmentation using only intensity context and incorporating location context. Clearly the location context lowers the PET-CT consistence penalty around the tumor, which happens to be where the CT segmentation and PET segmentation most likely differ. From the first example (first two rows in Fig.6.2), it is clear that enforcing a uniformly strong intensity context penalty "clipped" the CT segmentation to miss the lower left corner. The second example (last two rows in Fig.6.2) shows that enforcing a uniformly strong intensity-only context not only degrades the CT segmentation, but also the PET segmentation. Notice that PET segmentation in the third row, second column is missing a lower right component compared to the fourth row, second column. This is due to the fact that if the lower right component in PET is segmented, then the strong context term would force the CT segmentation to have roughly the same boundary as the PET component. But the CT boundary is quite different from the PET boundary in this area. So the strong intensity context term forces the consensus between PET and CT segmentations by labeling both as background at this region.

Quantitatively, Table.6.1 shows the DSC and ASSD when using only intensity context term, and using both location and intensity contexts. By incorporating the

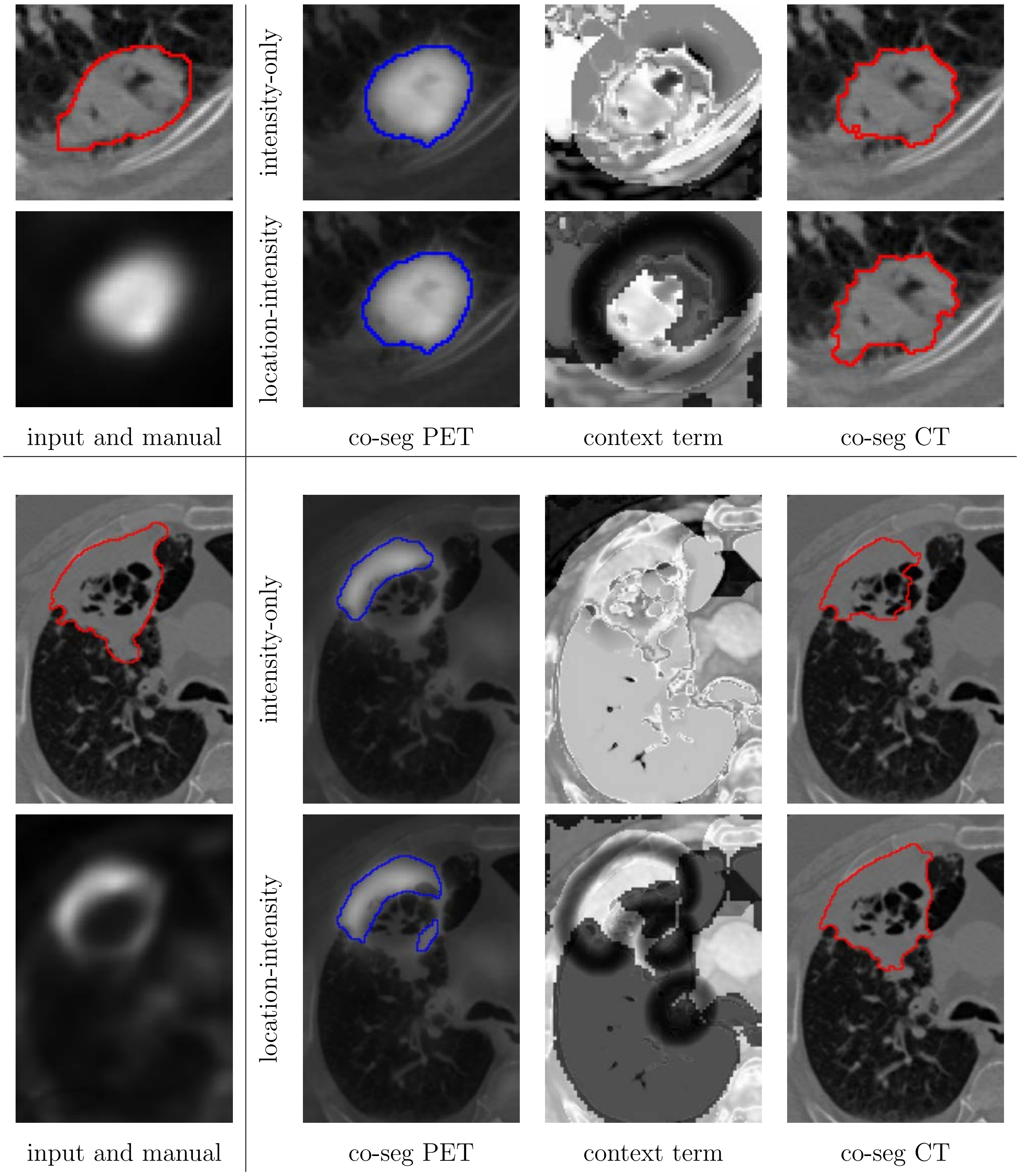


Figure 6.2: Incorporating location context term leads to more flexible and accurate segmentation. It lowers the context penalty at regions where PET and CT most likely disagree from each other.

Table 6.1: Co-segmentation accuracy.

context term	DSC	ASSD (mm)
intensity-only	0.77 ± 0.10	2.36 ± 1.38
location-intensity	$\mathbf{0.79 \pm 0.09}$	$\mathbf{2.14 \pm 1.29}$

location context, the DSC is improved from 0.77 to 0.79. The surface error ASSD is also improved from 2.36 *mm* to 2.14 *mm*. A two-tailed paired t-test shows that the co-segmentation accuracy using combined location and intensity context term is statistically significantly different from just using the intensity context term (p-value < 0.05).

6.3 Conclusion

Based on the observations that PET and CT segmentations are likely to agree at the core or far away from the tumor, and disagree around the tumor boundary, we propose a truncated quadratic location context term to allow more flexible co-segmentation. The experiment on 44 testing primary lung cancer PET-CT scan pairs demonstrates that the combined location and intensity context term leads to more accurate co-segmentation.

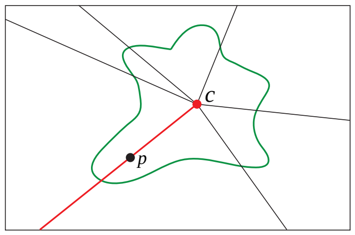
CHAPTER 7
OPTIMAL MULTI-SURFACE SEGMENTATION WITH STAR-SHAPE PRIOR

7.1 Introduction

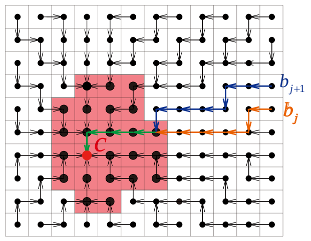
We present a novel graph-based optimal segmentation method which can simultaneously segment multiple star-shaped surfaces. Minimum and maximum surface distance constraints can be enforced between different surfaces. In addition, the segmented surfaces are ensured to be smooth by incorporating surface smoothness constraints which limit the variation between adjacent surface voxels. A consistent digital ray system is utilized to make sure the segmentation result is star-shaped and consistent. The problem is formulated as an MRF optimization problem which can be efficiently and exactly solved by computing a single min s-t cut in an appropriately constructed graph. The adoption of the consistent digital ray system enables to make full use of the discrete nature of the input images, and to enforce the star-shape prior and those geometric constraints without any further interpolation. To the best of our knowledge, the concept of consistent digital rays is for the *first* time introduced into the field of medical imaging. The method is applied to the segmentation of the optic disc and cup on 70 registered fundus and SD-OCT images from glaucoma patients. The result shows improved accuracy by applying the proposed method (versus using a classification-based approach).

An object is *star-shaped* if there exists one point (called the *star center*) of the object such that for every other point in the object, the whole Euclidean ray segment between them lies within the object (Fig.7.1a). The star-shaped prior requires minimal prior information (only location of the star center), while allowing much shape variation.

Veksler [97] presented a method incorporating the star-shaped prior into the graph-cut segmentation framework [12]. However, the digital ray segments used in this method are constructed in an ad hoc way which could lead to inconsistent segmentation results (for example, the segmentation results may contain artificial 'holes'). To make full use of the discrete nature of the input image, we propose to adopt the consistent digital ray system [19] to enforce the star-shaped prior.



(a)



(b)

Figure 7.1: (a) An example star-shaped object. (b) A consistent digital ray system and a digital star-shaped object (pink grids). c is the star center.

Chun et al. [19] developed a method to construct a consistent digital ray segment system for an N-D grid (image), which approximates the Euclidean ray segments with an asymptotically optimal guarantee of Hausdorff distance between the constructed digital ray segments and their Euclidean counterparts. A consistent digital ray system is a spanning tree $\mathcal{T}$ with each vertex corresponding to exactly one image voxel and tree edges being part of the grid topology of the input image (Fig.7.1b). A *digital ray segment* $dig(c, p)$ is defined as the unique portion of the halfline emanating from the star center c, with p as its second endpoint. An object $\mathcal{O}$ in the image is *star-shaped* with respect to c if for every voxel $q \in \mathcal{O}$, the whole digital ray segment $dig(c, q)$ is in the object $\mathcal{O}$.

We develop a novel graph-based segmentation method which simultaneously segments multiple star-shaped surfaces with a common star center. The problem is formulated as a Markov Random Field (MRF) optimization problem whose energy function consists of a region term, a boundary term and a star-shaped prior term. The optimization problem is exactly solved in low-order polynomial time by computing a single s-t cut/max-flow in an appropriately constructed graph. A novel feature of our method is that the star-shaped prior is enforced with respect to a consistent digital ray system, which makes sure that the segmentation result is consistent and visually well approximates a Euclidean star-shaped object. In our method, no interpolation of the input image is needed. The surface smoothness and the distance constraints between surfaces are enforced naturally in the discrete grid space of the image. Although Li et al. [52, 82] developed a novel method for simultaneously segmenting multiple interacting surfaces, the geometric constraints are enforced in the Euclidean space, which may require interpolation of the input image, potentially introducing artifacts. Delong et al.[23] proposed a multi-region segmentation approach which segments multiple interacting objects simultaneously. Their method has no control of the target object topology. The applicability of our method is demonstrated on the simultaneous segmentation of the optic cup and disc in fundus images.

7.2 Methods

7.2.1 MRF Formulation of Segmentation with Star-shape Prior

The segmentation problem is formulated as an MRF energy consisting of a region term which measures how well voxels fits into the object/background model, a boundary term which penalizes the discontinuity between object and background and a star-shaped prior term which enforces the star-shaped prior to the segmented surfaces. Minimizing the energy yields the optimal segmentation with respect to the given cost function.

Suppose we want to segment N surfaces $\mathcal{S}^i \in \mathcal{L}_N = \{0, 1, 2, \ldots, N-1\}$ in an input image $\mathcal{P}$. $\mathcal{N}_\mathcal{P}$ is the neighborhood setting for all adjacent voxels. We assume a 4-connected neighborhood setting in the following sections. For each surface $\mathcal{S}^i$ and each voxel $p \in \mathcal{P}$, we introduce a binary variable x_p^i: $x_p^i = 1$ means p is in the interior of the object whose boundary surface is $\mathcal{S}^i$; $x_p^i = 0$ indicates p is in the exterior of the object. The energy function is then defined as Eq.(7.1). $D_p^i(x_p^i)$ is the region term penalty for assigning label x_p^i to voxel p for segmenting surface $\mathcal{S}^i$. $V_{pq}^i(x_p^i, x_q^i)$ is the boundary term penalty for assigning labels x_p^i and x_q^i to two neighboring voxels p and q for surface $\mathcal{S}^i$. η is a weighting coefficient between the region term and the boundary term.

$$\mathcal{E}(\mathbf{x}) = \sum_{\substack{p\in\mathcal{P}\\ i\in\mathcal{L}_N}} D_p^i(x_p^i) + \eta \sum_{\substack{(p,q)\in\mathcal{N}_\mathcal{P}\\ i\in\mathcal{L}_N}} V_{pq}^i(x_p^i, x_q^i) + \sum_{\substack{q=parent(p)\\ i\in\mathcal{L}_N}} H_{pq}^i(x_p^i, x_q^i) \tag{7.1}$$

The third term $H_{pq}^i(x_p^i, x_q^i)$ is used to enforce the star-shaped prior. We adopt the consistent digital ray system computed by Chun et al.'s algorithm [19] to define the star-shaped prior. Recall that we use $dig(c,p)$ to denote a consistent digital ray segment between the star center c and the grid point (voxel) p. We say voxel q is the *parent* of voxel p, denote by $parent(p) = q$, if $(p,q) \in \mathcal{N}_{\mathcal{P}}$ and $q \in dig(c,p)$. In this case, p is the *child* of q. The star-shaped prior term $H^i_{pq}(x^i_p, x^i_q)$ is then defined as follows.

$$H^i_{pq}(x^i_p, x^i_q) = \begin{cases} 0 & \text{if } x^i_p = x^i_q \\ \infty & \text{if } x^i_p = 1 \text{ and } x^i_q = 0 \\ 0 & \text{if } x^i_p = 0 \text{ and } x^i_q = 1 \end{cases} \tag{7.2}$$

7.2.2 Transforming to the minimum s-t cut problem

In this section, we present our algorithm for optimally segmenting multi-surfaces with a star-shaped prior by minimizing the energy function $\mathcal{E}(x)$ in Eq.(7.1). The basic idea is to transform the optimization problem as computing a minimum s-t cut in a constructed graph encoding the energy function as well as the geometric constraints of the target surfaces, including the individual surface smoothness constraints and the surface distance constraints.

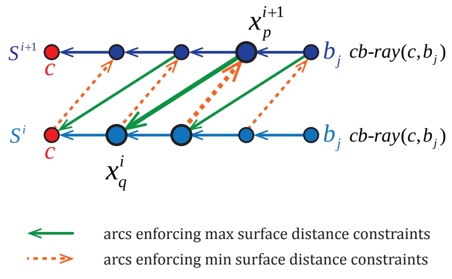
The constructed graph G consists of two dummy vertices, a source s and a sink t, and N subgraphs G^i. For each surface $\mathcal{S}^i$ we construct a subgraph $G^i = \{V^i, A^i_{st} \cup A^i_b \cup A^i_{star}\}$, in which V^i is the set of vertices with each corresponding to exactly one voxel in $\mathcal{P}$. To simplify the notation, we use x^i_p to denote both the random variable in Eq.(7.1) and its corresponding vertex in G^i. The region and boundary terms can be encoded by arcs A^i_{st} and A^i_b utilizing Boykov's graph-cut framework [12]. The arcs A^i_{star} are used to enforce the star-shaped prior, as described below.

Encoding the star-shaped prior. For any pair of neighboring vertices x^i_p and x^i_q with $(p, q) \in \mathcal{N}_\mathcal{P}$, if $q = parent(p)$ according to the consistent digital ray segment tree $\mathcal{T}$, then we put an arc from x^i_p to x^i_q with an infinite weight ∞. In this way, if the voxel corresponding to the child vertex x^i_p belongs to the object (belongs to the source set in the min *s*-*t* cut), then the voxel corresponding to its parent vertex x^i_q must also belong to the object. Otherwise, the induced *s*-*t* cut is not finite. The arrowed arcs in Fig.7.1b shows A^i_{star} for a specific surface $\mathcal{S}^i$. A^i_{star} exactly encodes the star-shaped prior term in Eq.(7.2) and thus ensures the segmented surfaces are star-shaped.

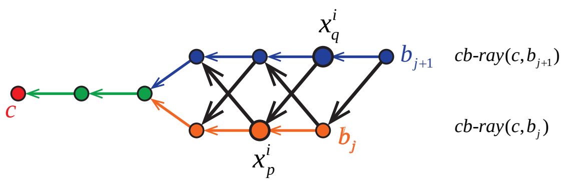
The use of a consistent digital ray system [19] is critical for the incorporation of the star-shaped prior without further image interpolation. In Veksler's method [97], the Euclidean rays are resampling in an ad hoc fashion to enforce the star-shaped prior, which may lead to visually inconsistent segmentation (e.g., with artificial 'holes'). In fact, the digital resampling in Veksler's method cannot guarantee some basic Euclidean geometry axioms [19] about the rays.

Encoding the min-max surface distance constraints. Without loss of generality, we assume that surface $\mathcal{S}^i$ is in the interior of the object whose boundary surface is $\mathcal{S}^{i+1}$. Assume that voxel p is on $\mathcal{S}^{i+1}$ and $q \in dig(c, p)$ is on $\mathcal{S}^i$. The *surface distance* between $\mathcal{S}^{i+1}$ and $\mathcal{S}^i$ along $dig(c, p)$ is defined as $dist(c, p) - dist(c, q)$, where

$dist(x, y)$ is the Euclidean distance between voxels x and y. We require that the surface distance between $\mathcal{S}^{i+1}$ and $\mathcal{S}^i$ should be no larger than a constant SD_{max} and no less than a constant SD_{min}. These min-max surface distance constraints can be enforced by introducing additional arcs between G^{i+1} and G^i.


x_p^{i+1}
S^{i+1}
c
b_j cb-ray(c, b_j)
S^i
c
b_j cb-ray(c, b_j)
x_q^i
arcs enforcing max surface distance constraints
arcs enforcing min surface distance constraints

(a) surface distance constraints



(b) smoothness constraints

Figure 7.2: Enforcing the geometric constraints. a Arcs with an infinite weight are added between two subgraphs for the segmentation of surfaces $\mathcal{S}^{i+1}$ and $\mathcal{S}^i$, respectively, to enforce the surface distance constraints. b The arcs (black) with an infinite weight are introduced between the vertices corresponding to two adjacent center-boundary rays $cb\text{-}ray(c, b_j)$ and $cb\text{-}ray(c, b_{j+1})$ to enforce the surface smoothness constrains.

A *center-boundary ray* is a digital ray segment between the star center c and an image boundary voxel b, denoted by $cb\text{-}ray(c, b)$, i.e., $cb\text{-}ray(c, b) = dig(c, b)$(see Fig.7.1b). We enforce the min-max surface distance constraints between $\mathcal{S}^{i+1}$ and $\mathcal{S}^i$ along each center-boundary ray $cb\text{-}ray(c, b)$.

For any vertex x_p^{i+1} in G^{i+1} with $p \in cb\text{-}ray(c, b)$, we find the vertex x_q^i in G^i with $q \in cb\text{-}ray(c, b)$, such that $dist(c, p) - dist(c, q) \leq SD_{max}$ and $dist(c, p) - dist(c, parent(q)) > SD_{max}$. Then we add an arc (x_p^{i+1}, x_q^i) with an infinite weight ∞. Intuitively, if p is on surface $\mathcal{S}^{i+1}$, then q must be the interior of the object whose boundary surface is $\mathcal{S}^i$ due to the infinite weight of the arc introduced. This ensures the surface distance along $cb\text{-}ray(c, b)$ does not exceed SD_{max}, enforcing the maximum surface distance constraints.

Similarly, we can enforce the minimum surface distance constraints between $\mathcal{S}^{i+1}$ and $\mathcal{S}^{i}$ along each center-boundary ray $cb\text{-}ray(c, b)$. For any vertex x_p^{i+1} in G^{i+1} with $p \in cb\text{-}ray(c, b)$, we compute the vertex x_q^i in G^i with $q \in cb\text{-}ray(c, b)$, such that $dist(c, p) - dist(c, q) \geq SD_{min}$ and $dist(c, p) - dist(c, child(q)) < SD_{min}$. Then add an arc (x_q^i, x_p^{i+1}) with infinite weight to enforce the minimum surface distance constraint. Fig.7.2(a) illustrates the arcs enforcing the min-max surface distance constraints.

Encoding the surface smoothness constraints. Suppose that two voxels p and q are both on the surface $\mathcal{S}^i$, and $p \in cb\text{-}ray(c, b_j), q \in cb\text{-}ray(c, b_{j+1})$, in which b_j and b_{j+1} are two adjacent voxels on the image boundary. The *smoothness variation* of surface $\mathcal{S}^i$ between $cb\text{-}ray(c, b_j)$ and $cb\text{-}ray(c, b_{j+1})$ is then defined as $|dist(c, p) - dist(c, q)|$. The surface smoothness constraint ensures that the smoothness variation between any two adjacent center-boundary rays does not exceed a non-negative constant SM_{max}. For each G^i and any vertex x_p^i with $p \in cb\text{-}ray(c, b_j)$, compute the vertex x_q^i with $q \in cb\text{-}ray(c, b_{j+1})$ but $q \notin cb\text{-}ray(c, b_j)$, such that $dist(c, p) - dist(c, q) \leq SM_{max}$ and $dist(c, p) - dist(c, parent(q)) > SM_{max}$. We then add an arc (x_p^i, x_q^i) with an infinite weight of ∞ to enforce the constraint that $dist(c, p) - dist(c, q) \leq SM_{max}$. Similarly, we can enforce the constraint $dist(c, q) - dist(c, p) \leq SM_{max}$. Fig.7.2(b) shows how the arcs are added for the smoothness constraints.

7.3 Application to Optic Cup and Disc Segmentation

The applicability of the reported approach is demonstrated on the automated segmentation of optic cup and disc surfaces (borders). Although the application is in 2D, our method is applicable for higher dimensional image segmentation.

Stereo fundus color photographs and spectral-domain optical coherence tomography (SD-OCT) scans of the optic nerve head are obtained from 70 eyes of 35 patients with glaucoma at the same day. Each fundus image contains 1019×768 voxels with voxel size of $30 \times 30\mu$m. Each SD-OCT scan contains $200 \times 200 \times 1024$ voxels with voxel size of $30 \times 30 \times 2\mu$m. The 2D fundus image is registered with the SD-OCT scan and cropped to 101×101 for convenience of processing. The manual reference standard on the 2D fundus image is obtained from three expert segmentations based on consensus, i.e., each voxel is assigned a label which receives the majority of votes.

7.3.1 Cost function for region term and boundary term

For the region term, first we generate three probability maps which provide the probabilities for each voxel to be cup (region within the inner boundary), rim (region between the inner and outer boundaries), and background voxel (region outside the outer boundary) respectively, then we convert them to the region term form in Eq.(7.1).

A machine-learning based approach which combines information from the 3D SD-OCT scan and 2D fundus image is used [51]. The dataset is divided into 10 folds and at each time one fold is left out and the classifier is trained on the rest of nine folds to predict the three probability maps of the one fold that has been left out. Denote X^0, X^1, X^2 as the three probability maps for cup region, rim region, and background, respectively. The region term is then computed: $D_p^i(0) = C; D_p^i(1) = -\log(X_p^i) - (-\log(X_p^{i+1})) + C$ $(i = 0, 1)$, in which C is a large enough constant to make sure both $D_p^i(0)$ and $D_p^i(1)$ are nonnegative.

We used the class-uncertainty method [58] to compute the boundary term for the disc surface. For each voxel p, the class-uncertainty method computes the uncertainty u_p, of classifying each voxel as object or background. Similarly, for any q with $(p, q) \in \mathcal{N}_{\mathcal{P}}$, the uncertainty u_q is also computed. Boundary voxels tend to have the highest uncertainty when doing such classification. The boundary term is defined as $V_{pq}^1(0, 1) = V_{pq}^1(1, 0) = -\log(\frac{u_p+u_q}{2})$. Currently we cannot obtain reliable boundary information for the cup surface, so we set the boundary term for the cup surface to be zero.

7.3.2 Parameter settings and experiment settings

The parameters in the experiment are set as follows. The minimum and maximum surface distance constraints are loosely set to be 1 voxel and 50 voxels respectively. The smoothness constraints for the cup and disc surfaces are set 2 and 3 voxels respectively. η is 5. All datasets share the same parameter settings.

The star centers for 65 of the fundus images are automatically estimated by thresholding the probability map for cup region to get an estimated segmentation of cup and computing centroid of it as the star center. For the remaining 5 fundus cases, we manually specify the star center location due to a large error of automatic estimation.

The segmentation results are compared to the reference standard obtained from three independent observers. The manual segmentation by three observers are also compared with each other which is presented as *inter-observer variability*, e.g., obs1 vs. obs2, etc. As a reference, the automatic segmentation using the most likely label according to the probability map in the region cost is also conducted, denoted as the *region-only* segmentation.

Dice coefficient (DSC) is used to evaluate the results in terms of overlapping volume. Unsigned surface distance error and signed surface distance error are used to evaluate the results in terms of surface positioning error. As shown in Fig.7.3c, the region-only segmentation does not always have a well-defined boundary. We use the method in [51] to generate an "average" surface for the region-only segmentation and measure the unsigned/signed surface distance error using this "average" surface.

7.3.3 Results

Table.7.1 shows the DSC for inter-observer variability, the region-only segmentation and the proposed segmentation. The bold numbers show the best results. A paired student t-test shows that the proposed method shows significant improvement in the rim DSC compared to the region-only segmentation. There is no significant difference for the cup DSC.

Table.7.2 and Table.7.3 show the unsigned and signed surface distance errors.

Table 7.1: Dice coefficient (larger is better).

	obs1 vs. obs2	obs2 vs. obs3	obs1 vs. obs3	region-only	proposed
cup	0.791 ± 0.173	0.817 ± 0.153	0.806 ± 0.122	0.827 ± 0.159	**0.829** ± **0.148**
rim	0.580 ± 0.170	**0.645** ± **0.178**	0.510 ± 0.212	0.628 ± 0.155	0.643 ± 0.151

Table 7.2: Unsigned surface distance error in unit voxel (smaller is better).

	obs1 vs. obs2	obs2 vs. obs3	obs1 vs. obs3	region-only	proposed
cup	3.00 ± 1.59	3.13 ± 1.24	2.80 ± 1.56	2.61 ± 1.51	**2.54** ± **1.26**
disc	3.32 ± 1.23	2.81 ± 1.28	**1.84** ± **0.93**	2.32 ± 1.12	2.30 ± 1.15

A paired student *t*-test shows the proposed method achieves significant improvement in the unsigned/signed surface distance error for the disc surface, and in the signed surface distance error for the cup surface. There is no significant difference in the unsigned surface distance error for the cup surface.

Illustrative results are shown in Fig.7.3. In the first row, the star-shaped prior and smoothness constraints removes intensive noise in the region-only segmentation. In the second row c, note the cup surface and disc surface are attached to each other in the left portion (temporal side). But enforcing the minimum surface distance constraint of 1 voxel makes sure the cup and disc surfaces are detached from each other (second row d).

Given the cost function available, the processing time for each 101×101 image

Table 7.3: Signed surface distance error in unit voxel.

	obs1 vs. obs2	obs2 vs. obs3	obs1 vs. obs3	region-only	proposed
cup	0.40 ± 2.08	-1.84 ± 1.84	-2.24 ± 2.11	0.69 ± 2.30	**0.16** ± **2.2**
disc	-2.36 ± 1.27	-1.01 ± 1.02	1.29 ± 1.29	-0.07 ± 1.70	**-0.01** ± **1.72**

is about 0.3 seconds.

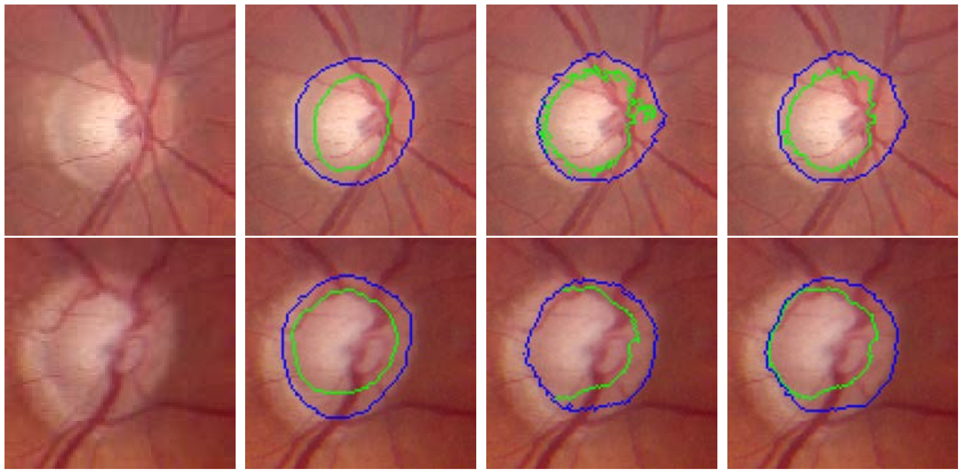


Figure 7.3: Illustrative results of region-only and star-shaped segmentation.

7.4 Conclusion

We presented a novel graph-based optimal segmentation method which can simultaneously segment multiple interacting star-shaped surfaces. Consistent segmentation results are achieved by using the consistent digital ray system [19]. Minimum and maximum surface distance constraints and smoothness constraints are also enforced. Validation experiments on 70 clinical eye scans showed the accuracy improvement by applying the proposed method.

CHAPTER 8
OPTIMAL MULTI-OBJECT SEGMENTATION WITH NOVEL GRADIENT VECTOR FLOW SHAPE PRIOR

8.1 Introduction

Shape prior are widely used in medical image segmentation due to the similar intensity profile and weak boundary between target object and background [64, 97, 99, 87, 86, 7, 101]. For example, the popular deformable model [2, 102, 60, 61] and LOGISMOS [106, 64, 110, 86, 87] methods (layered optimal graph image segmentation of multiple objects and surfaces) use mesh in physical space to ensure the segmentation is roughly aligned with some initial model or pre-segmentation. However, the mesh representation is prone to self-intersection, or “mesh folding” problem, which requires complex and expensive algorithms to mitigate. In this chapter, we propose a novel shape prior directly embedded in the voxel grid space, based on gradient vector flow (GVF) of a pre-segmentation. The flexible and powerful shape prior can be straightforwardly extended to simultaneously segment multiple interacting objects with minimum separation distance constraint. The problem is formulated as a Markov random field problem whose exact solution can be efficiently computed in a single minimum *s*-*t* cut in an appropriately constructed graph.

The proposed algorithm is validated on the two multi-object segmentation applications: brain tissue segmentation in MRI images, and the bladder/prostate segmentation in CT images. Both sets of experiment show superior or competitive performance of the proposed method to other state-of-art methods.

8.1.1 Shape prior by mesh

Deformable model first initializes a mesh representing the target object close to the desirable location and orientation [26, 2, 102]. The boundary and regional information are then used to evolve the mesh vertices along its normal direction to adhere to image content. An elastic force is also often used to limit the amount of warp to enforce the shape prior [26, 102]. Such evolution is repeated multiple times until convergence.

LOGISMOS is a popular multi-surface segmentation method with the ability of enforcing minimum and/or maximum surface distance constraints among multiple target surfaces [106, 52, 110]. The method first builds a mesh based on an initial pre-segmentation of the target object. The image is then resampled usually along the normal direction at each mesh vertex to constitute a graph node column. The graph nodes in these columns represent the set of all possible target surface locations sampled at each mesh vertex normal direction. Various arcs are then added among the nodes to ensure the surface intersects each column exactly once, enforce a hard smoothness constraint to limit the abrupt surface jump between adjacent columns, and a soft convex function encouraging the surface shape to further conform to the base mesh [109, 87, 86]. The globally optimal surface location is then found by solving a single minimum s-t cut in the constructed graph. In sum, the shape prior is enforced by first limiting the surface search space within the graph column reach, and then enforcing a hard and a soft smoothness constraint among adjacent columns.

However, the mesh-based shape prior representation have several shortcomings in practice. Firstly, the mesh may "self-intersect" or "fold" in the returned solution, causing undesired dramatic topology change from the prior shape. The problem is especially noticeable at high curvature locations such as concavities and bifurcations (see Figure.8.1). To avoid this problem, the deformable method need to run expensive self-intersection detection and remeshing algorithms after each evolution step [66, 70, 100]. For LOGISMOS, various methods are proposed to prevent the graph node column intersection. The electrical field [109], flow lines [67] and gradient vector flow [64] build curved graph columns instead of traditional straight normal direction columns. The omnidirectional displacement method [42] distributes graph nodes uniformly in a sphere centered at each mesh vertex instead of along a column. However, the resulting MRF problem can no longer be solved by minimum s-t cut and the high computation cost restricts it to be only applied at high curvature regions instead of whole image.

Secondly, it is difficult to enforce interaction priors among multiple objects, since all mesh vertices and edges are represented in continuous physical coordinate. For deformable model method, complex mesh collision detection algorithm have to be run at the end of each evolving iteration to avoid two close objects from intersecting each other [46, 90]. For LOGISMOG-based methods, nesting surfaces must share the same base mesh in order to enforce the minimum/maximum surface distance constraints [64]. This makes it impossible to enforce independent shape priors for different nesting surfaces. For multiple excluding surfaces, Song et. al. [86, 87] have to merge the graph columns from prostate and bladder meshes in the "interactive"

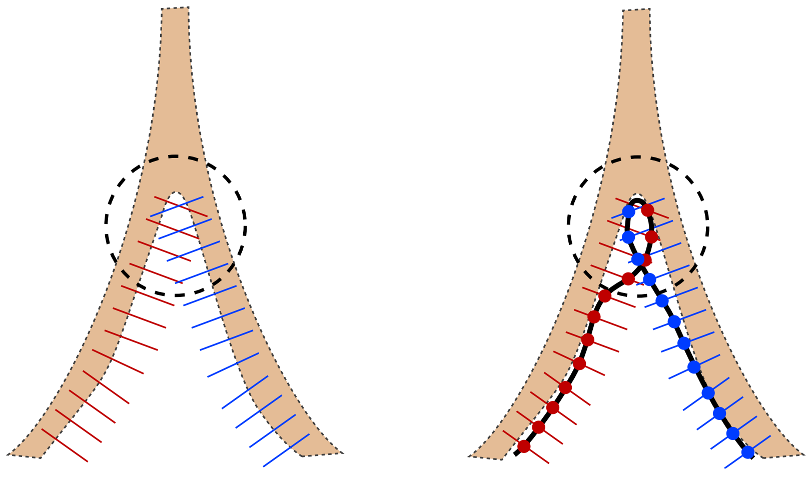

(a) column intersection (b) folded mesh

Figure 8.1: Mesh folding at bifurcation locations. (a) shows part of LOGISMOS graph columns built based on pre-segmentation (light brown region). The red columns and blue columns are intersecting in the circled area. This may lead to folded mesh as shown in (b) in the solution. Although deformed model do not explicitly build such columns, the mesh folding may still happen during mesh evolving.

region to enforce the exclusion relationship between them.

8.1.2 Shape prior directly in voxel space

Another group of methods embed the shape prior directly in the voxel space, fundamentally eliminating the mesh intersection problem. Veksler et. al. [97] introduced a flexible star-shaped prior which requires every point in target object to be visible to a predefined *star center* point through a straight line totally within the object. Infinite arcs mimicing straight rays spanning from star center are added in a graph-cut based graph to encode the star-shaped prior. However, these arcs are obtained from a heuristic algorithm. Bai et. al. [10] uses *consistent digital ray* to systematically build the infinite arcs so that a tight Hausdorff distance bound can be obtained between the discretized rays and its Euclidean counterpart. Isack et. al. [38]

proposes “hedgehog” shape prior which extends the star-shaped prior. A vector field based on the distance transform of the user scribbles is used to limit the segmented surface normals to be within certain angles tolerance from the scribble normals. This ensures the skeleton of the segmentation to be roughly the same as the user scribbles. However, it may be difficult to draw scribbles capturing the object skeleton in 3D medical images, especially for complex surfaces such as white tissue surface in brain.

If more detailed template or pre-segmentation is available, then the distance transform of the aligned template can be incorporated into the pairwise term of the graph-cut framework to softly penalize the difference between segmentation and the template [99, 7]. However, it is difficult to balance this soft shape prior term with other image information terms.

In contrast, the star-shaped prior [97, 10] and “hedgehog” shape prior [38] clearly defines a *class* of valid shapes. Any given shape can be crisply classified as either valid or invalid. The shape prior term is always enforced as a hard constraint which rejects, not just penalizes, invalid shapes. Thus, we do not have to balance the shape prior term and the image information term. Note these shape priors are not overly restrictive since large variations are allowed in the set of valid shapes.

8.1.3 Proposed method contributions

Gradient vector flow (GVF) [107] defines a diffusion of the gradient vectors from a grayscale or binary edge map. When computed from edge map, it defines a vector roughly pointing towards the closest strongest edge at each voxel. In this chapter, however, we are going to show that if computed from a volumetric pre-segmentation, instead of an edge map, the GVF encodes descriptive shape information from the pre-segmentation.

We propose a novel gradient vector flow shape prior which is embedded *directly* into the voxel space, an Markov random field (MRF) formulation to encode it, and an efficient way to optimize the proposed MRF energy. In summary, our contributions are:

- A novel and flexible gradient vector flow shape prior embedded directly in the voxel space, avoiding the mesh intersection problem in mesh shape prior representation.
- No need to balance the shape prior term and image information term, since the proposed GVF shape prior crisply classifies every possible segmentation as either valid or invalid.
- An efficient MRF formulation to encode the shape prior and image information, whose exact solution can be computed in a single minimum s-t cut.
- Easy to incorporate inclusion/exclusion multi-object interaction constraints.

8.2 Methods

We first introduce the gradient vector flow and the GVF shape prior, then we show how to incorporate such shape prior into an MRF formulation and how to segment multiple interacting objects simultaneously.

8.2.1 Gradient vector flow

Suppose $\mathcal{P}$ is the set of image voxels in a 3D grid, then binary segmentation assigns every voxel $p \in \mathcal{P}$ a label $f_p \in \mathcal{L}$, where $\mathcal{L} = \{0, 1\}$ is the set of available labels. Assuming a pre-segmentation $\hat{S} = \{p|\hat{f}_p = 1, p \in \mathcal{P}\}$ is available, then the gradient of the pre-segmentation $\nabla\hat{S}$ is only nonzero around the pre-segmentation boundary (Fig.8.2a). In the narrow range where $\nabla\hat{S}$ is nonzero, the gradient vectors encode the pre-segmentation shape information in the sense that they are the most direct direction pointing towards the inside of the pre-segmentation.

In order to enforce the shape prior in a larger range, we need to use GVF to propagate the gradient vectors. The GVF takes the raw gradient vector field $\nabla\hat{S}$ as input, and computes another vector field **h** to minimize the following energy.

$$\mathcal{E}^{GVF}(\mathbf{h}) = \iiint_{\mathcal{P}} \|\nabla\hat{S}\|^2\|\mathbf{h} - \nabla\hat{S}\|^2 + \mu\|\nabla\mathbf{h}\|^2\, dx\, dy\, dz \tag{8.1}$$

The first term is the product of gradient magnitude and squared difference of output vector field and input gradient. The second term is the sum of squares of the partial derivatives of the output GVF vectors. More specifically, the first term is dominant when input gradient $\nabla\hat{S}$ is strong, which encourages the output vector field **h** to align well with input gradient. The second term is dominant when input gradient is small, which ensures the output vector field to change smoothly.

In summary, at regions with strong input gradient, GVF follows the input gradient vector field; at regions with weak or no input data, GVF interpolates from nearby locations to ensure the output vector field changes smoothly. The constant

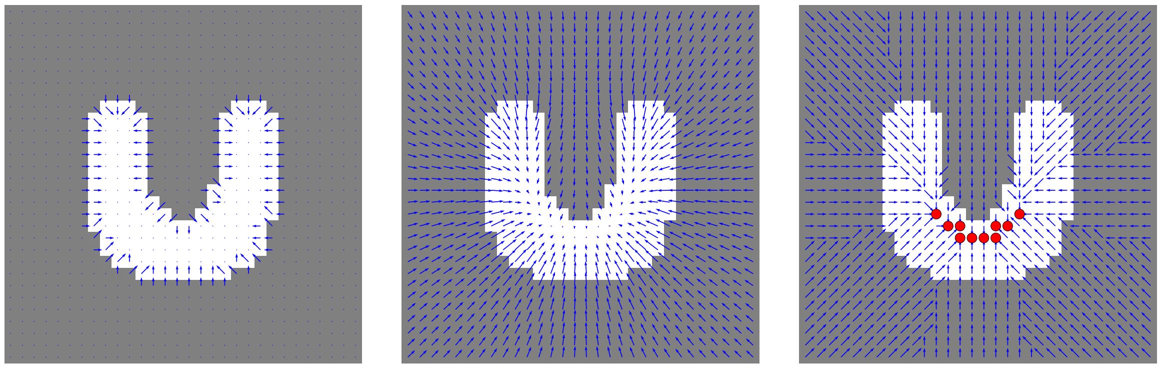

(a) Gradient (b) GVF (c) discretized GVF

Figure 8.2: Use GVF to encode shape prior information throughout the image. (a) shows that raw gradient of the pre-segmentation is restricted to a narrow band around the boundary. GVF propagate it smoothly throughout the image as in (b). A discretized GVF in (c) builds a GVF path for every non-core voxels connecting to one of the core voxels (red circles).

μ controls how strong the interpolation should be. The net effect is that the input gradient vector field is smoothly propagated from a narrow band surrounding the pre-segmentation boundary to all of the image. Fig.8.2b shows how a GVF vector field propagates gradient information (Fig.8.2a) to all voxels of the image.

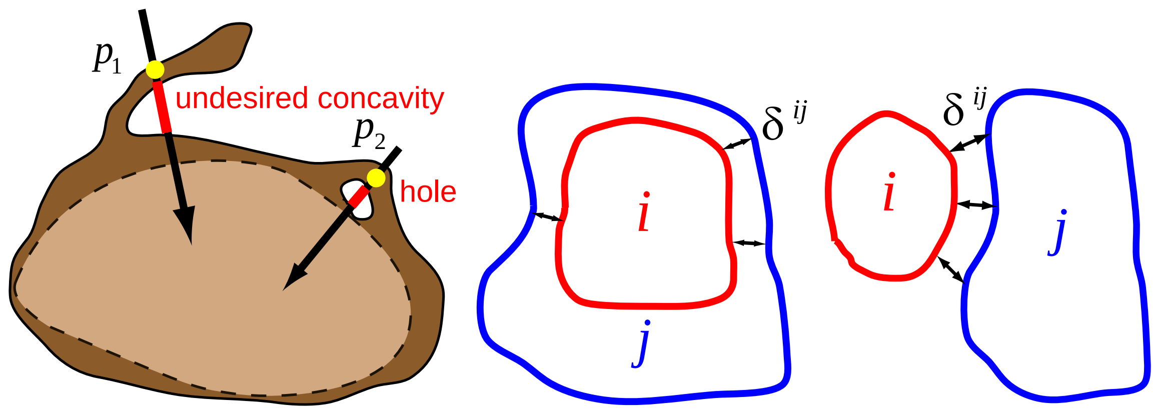
8.2.2 GVF shape prior in grid space

From Fig.8.2b, we can see that the GVF magnitude is very small at the "core" of pre-segmentation due to the competition of boundary gradients from multiple directions. Formally we define the object *"core"* $\mathcal{C}$ as the set of voxels inside the pre-segmentation with GVF magnitude smaller than a small threshold $\theta_{\mathcal{C}}$, i.e., $\mathcal{C} = \{p \in \mathcal{P} | p \in \hat{S}, \|\mathbf{h}_p\| < \theta_{\mathcal{C}}\}$. The red circled voxels in Fig.8.2c are the core voxels of the white pre-segmentation.

For every non-core voxel in the GVF field, we can find a smooth path following which one can travel to one of the core voxels. If one voxel is part of the object, then all voxels along the GVF path connecting it to the core should also be part of object if we want to roughly preserve the shape of the pre-segmentation. Otherwise, a hole or undesired cavity may appear along the GVF paths.

Since segmentation requires us to label each voxel in a grid space, we discretize the GVF field so that the GVF vector at each voxel points exactly towards one of its neighbor voxels. More concretely, the GVF vector field $\mathbf{h}$ is transformed into the discretized GVF vector field $\mathbf{h}^D$. For every pixel p, $\mathbf{h}_p^D = \overrightarrow{pq}$ where $q = \operatorname{argmax}_{q \in \mathcal{N}_p} \mathbf{h}_p \cdot \overrightarrow{pq}/(\|\mathbf{h}_p\| \|\overrightarrow{pq}\|)$. $\mathcal{N}_p$ is the set of neighboring voxels of voxel p. In another word, $\mathbf{h}_p^D$ is the vector from p to its neighbor voxel with the smallest angle error when approximating GVF vector $\mathbf{h}_p$. See Fig.8.2c for an example of the discretized GVF field.

For any non-core voxel $p \notin \mathcal{C}$, the *GVF path* connecting itself to the object core is defined as $GP(p) = q_0, q_1, \ldots, q_K$, such that $\overrightarrow{pq}, \overrightarrow{q_0 q_1}, \ldots, \overrightarrow{q_{K-1} q_K} \in \mathbf{h}^D$, and $q_K \in \mathcal{C}$. The *GVF shape prior* is defined as: if one voxel p is part of the object, then all voxels along the GVF path $GP(p)$ are also part of the object. Geometrically, it makes sure all voxels along the GVF path connecting p to the object core are all part of foreground if p is. In Fig.8.3a, voxel p_1 and p_2 are part of the foreground, but part of the GVF path connecting them to the object core (thick black lines) are not labeled as foreground (highlighted by thick red lines). Thus, the GVF shape prior based on the pre-segmentation (light brown region) is violated by the current segmentation (deep brown region), due to undesired concavities and holes.



(a) GVF shape prior (b) inclusion (c) exclusion

Figure 8.3: GVF shape prior and inclusion/exclusion multi-object interaction prior. (a) demonstrates that exaggerated concavity and holes not existing in pre-segmentation would violate the GVF shape prior. (b) and (c) show the inclusion and exclusion interaction relationship between object i and j with minimum distance δ^{ij} in between.

8.2.3 Shape prior MRF penalty function

For a single voxel p, the GVF shape prior can be enforced by enforcing the following penalty function between p and all $q \in GP(p)$.

$$\phi(f_p, f_q) = \infty \cdot [f_p = 1, f_q = 0] \tag{8.2}$$

where $[\cdot]$ is the indicator function which returns 1 when the enclosed condition is true and 0 otherwise. The penalty function $\phi(f_p, f_q)$ returns ∞ when p is part of object, and q is background. Otherwise, the penalty function returns 0. In a minimization MRF problem, this ensures all solution violating the GVF shape prior are not valid.

Due to the possible large number of voxels along the GVF path, enforcing

$\phi(f_p, f_q)$ between p and all $q \in GP(p)$ naively is computationally expensive, especially considering this process have to be repeated for every voxel $p \in \mathcal{P}$. Fortunately, it can be shown [97] that GVF shape prior can be sufficiently enforced by using only $\phi(f_p, f_{q0}), \phi(f_{q0}, f_{q1}), \ldots, \phi(f_{q_{K-1}}, f_{qK})$ where $q_0, q_1, \ldots, q_K = GP(p)$. This way, the penalty function only need to be enforced between p and its nearest neighbor indicated by the discretized GVF vector $\mathbf{h}_p^D$. This leads to the observation that to enforce the GVF shape prior for all voxels $p \in \mathcal{P}$, we only need to iterate through the discretized GVF vector field $\mathbf{h}^D$ (Eq.(8.3)). Note $\|\mathbf{h}^D\| = \|\mathcal{P} - \mathcal{C}\|$, so we only need to enforce the GVF penalty function between linear number of voxel pairs.

$$\sum_{\overrightarrow{pq} \in \mathbf{h}^D} \phi(f_p, f_q) = \sum_{\overrightarrow{pq} \in \mathbf{h}^D} \infty \cdot [f_p = 1, f_q = 0] \tag{8.3}$$

8.2.4 MRF formulation

Suppose $\mathcal{N}_\mathcal{P}$ is the neighborhood system in image $\mathcal{P}$, then the overall MRF energy with GVF shape prior is as follows.

$$\mathcal{E}(\mathbf{f}_\mathcal{P}) = \sum_{p \in \mathcal{P}} D(f_p) + \sum_{(p,q) \in \mathcal{N}_\mathcal{P}} V_{pq}(f_p, f_q) + \sum_{\overrightarrow{pq} \in \mathbf{h}^D} \phi(f_p, f_q) \tag{8.4}$$

$D_p(f_p)$ is a data term describing the appearance information of voxel p. The more likely it belongs to foreground, the smaller $D_p(f_p = 1)$ is, and the larger the $D_p(f_p = 0)$ is. $V_{pq}(f_p, f_q)$ is a pairwise smoothness term based on the boundary/edge information between neighboring voxels p and q. If $f_p = f_q$, then $V_{pq}(f_p, f_q) = 0$. Otherwise, $V_{pq}(f_p, f_q)$ is a nonnegative number that is inversely proportional to the likelihood of an edge between neighboring voxels p and q. In general, the smoothness term encourages a smoother boundary between object and background. The more likely an edge lies between neighboring voxels, the smaller the smoothness penalty is.

The third term in Eq.(8.4) enforces the GVF shape prior as in Eq.(8.3). Note this term is submodular since $\phi(0,0) + \phi(1,1) < \phi(1,0) + \phi(0,1)$. Thus, the overall energy is submodular, whose optimal solution can be computed by a minimum s-t cut in an appropriately built graph [50] based on graph-cut method [12].

8.2.5 Simultaneous multi-objects segmentation

Medical image applications often require *multiple* objects to be segmented simultaneously, e.g., prostate and bladder [87, 86], white matter and gray matter in brain [64]. It is advantageous to incorporate anatomical prior information between multiple objects when possible. Based on a graph-cut formulation, Delong et. al. [24] enforced inclusion or exclusion relationships between two objects with a predefined minimum distance in between. Fig.8.3b and 8.3c show examples of such inclusion and exclusion interactions between two objects i and j, with a minimum distance δ^{ij} between them.

Since the proposed energy in Eq.(8.4) can be solved by graph-cut, it is straightforward to apply the same multi-object constraints (minimum distance between nested surfaces or excluded objects). The trick to construct K binary variables $f_p^1, f_p^2, \ldots, f_p^K \in \{0,1\}$ for each voxel p, where $f_p^k = 1$ indicates voxel p is part of k-th object, and not part of k-th object otherwise. Multiple energy functions using $\mathbf{f}_{\mathcal{P}}^k$ are encoded by multiple graph-cut based subgraphs. Various arcs are then introduced between pairs of objects with inclusion/exclusion interaction to enforce the minimum distance constraints.

It is also possible to enforce the *maximum* surface distance between pairs of nested surfaces when they share the same pre-segmentation. In this case, the two surfaces share the common GVF paths which serve as graph columns in grid space. This common column structure enables the maximum surface distance constraint to be enforced using techniques used in LOGISMOS [10].

8.3 Experiment

We validated the proposed method on two applications: the prostate and bladder segmentation in CT images, and the brain tissue segmentation in MRI T1 images. The first application aims to segment two *mutually exclusive* objects, whereas the brain tissue segmentation aims to segment two *nesting* surfaces which separates three different types of tissues (white matter, gray matter, and cerebrospinal fluid).

8.3.1 Experiment settings

For both applications, we first obtain an initial segmentation, i.e., pre-segmentation. The proposed GVF shape prior is defined based on this pre-segmentation. The final segmentation output accuracy is assessed by two metrics: Dice similarity coefficient (DSC) and average symmetric surface distance (ASSD). The *Dice similarity coefficient* is used to measure how well two volumes overlap with each other. Assume A and B are two volumes, then the DSC between the two volumes is defined as

$2|A \cap B|/(|A| + |B|)$, which ranges between 0 and 1. The larger the DSC is, the better the two volumes are aligned, with 1 indicating a perfect overlapping. The *average symmetric surface distance* (ASSD) is used to measure how close two segmented surfaces are. Let $d(x, A)$ denote the shortest distance between a point x and any point on the surface A. The ASSD between the segmented boundary surfaces A and B by two different methods is defined as $ASSD = \sum_{a \in A} d(a, B) + \sum_{b \in B} d(b, A)|A| + |B|$. It measures the average distance from any point on a contour/surface to the other contour/surface. The ASSD metric has a range of $[0, +\infty]$. The smaller ASSD is, the better the two segmented surfaces/contours agree with each other. If $ASSD = 0$, then the two segmentations are identical.

8.3.2 Brain segmentation: data and compared methods

For brain tissue segmentation, 18 T1-weighted scans of normal subjects from the Internet Brain Segmentation Repository (IBSR) [1] were used (commonly know as "IBSR 18" in literature) [104, 95]. The image size is $256 \times 128 \times 256$ with voxel spacing range from $0.837 \times 1.5 \times 0.837mm^3$ to $1 \times 1.5 \times 1mm^3$. Each scan comes with a brain mask marking voxels inside skull. Each voxel inside the brain mask is labeled by expert as one of three labels: white matter (WM), gray matter (GM), and cerebrospinal fluid (CSF). Using this public dataset allows us to directly compare the proposed method to various state-of-art methods.

Valverde *et al.* [95] validated 10 brain segmentation algorithms on the IBSR dataset, aiming to evaluate a wide set of available technique and tools (refer Table.1

in [95] for software sources and versions). We compare our method to the results reported. These ten algorithms are summarized below: FAST [113] uses expectation maximization with K-means initialization to optimize an MRF model; SPM5 [6] and SPM8 [5] are two versions of the SPM toolbox based on iterative Gaussian mixture model, atlas registration and bias correction; GAMIXTURE [92] estimates a Gaussian mixture model by genetic algorithm (GA); ANN [91] implements a self organizing map clustering image data; FCM [69] uses a fuzzy c-means clustering algorithm; KNN [22] uses a k-nearest neighbor algorithm based on automatic registration of prior probability atlases; SVPASEG [93] uses iterative conditional modes (ICM) method with genetic algorithm initialization to optimize an MRF model; FANTASM [68] extends FCM by adding a spatial term in the objective function; and PVC [78] builds a maximum-a-posteriori (MAP) model with ICM optimization. We refer readers to Table.1 in [95] for software sources and versions used in reporting these numbers.

In addition to the above 10 methods, we also include the result reported by the following state-of-art methods: discriminative model-constrained EM approach combines supervised discriminative modeling and unsupervised statistical expectation maximization (EM) segmentation into an integrated Baysian framework; adaptive Markov modeling based on mutual information [8]; hidden Markov chain model [17] which unifies partial volume effect, bias field correction, and a probabilistic atlas; prior knowledge driven multiscale segmentation [3] embeds an atlas prior in a multi-scale pyramid.

8.3.3 Brain segmentation: workflow

8.3.3.1 Preprocessing

The images are first masked by the brain mask to exclude voxels outside the skull, reoriented to have the standard orientation ('RAI' orientation), and resampled to have unit voxel spacing. We use 2-fold cross validation for this experiment. The 18 images are randomly split into two disjoint set, with one of them being training set and another being test set. Then the role of training/testing set are reversed for both sets. To account for the inter-scan intensity profile difference, we first compute a mean brain histogram within the training set. Then we run histogram matching to match every test image to the mean training brain histogram.

8.3.3.2 Pre-segmentation

The BRAINSABC software[1] [112] is used to classify every voxel within the brain mask into one of the three tissue classes: WM, GM, and CSF. BRAINSABC first deformably registers input image to an atlas, then performs atlas-based tissue classification using an expectation-maximization approach [96]. The largest connected component consisted of white matter voxels are used as the pre-segmentation of the inner surface, i.e., WM-GM boundary. The largest connected component consisted of white matter *and* gray matter voxels is used as the pre-segmentation of the outer surface, i.e., GM-CSF boundary. The GVF shape priors are defined with respect to these pre-segmentations. To simplify notation, we would call the WM-GM boundary

[1]Available online: `https://github.com/BRAINSia/BRAINSTools`

as the WM surface, and the GM-CSF boundary as the GM surface. The WM surface is set to be included in the GM surface with a minimum distance of 1 mm.

8.3.3.3 Energy term design

A random forest using a simple four-dimensional feature vector is then used to generate voxelwise probability maps for all three types of tissues. The four features are x/y/z coordinates and intensity for each voxel. Using techniques from Sec.8.2.5, two variables $f_p^{\mathrm{WM}}, f_p^{\mathrm{WM/GM}} \in \{0, 1\}$ are introduced for each voxel p. The corresponding data terms $D(\cdot)$ in Eq.(8.4) for the two labels are defined in Eq.(8.5) and (8.6).

$$\begin{aligned} D_p^{\mathrm{WM}}(f_p^{\mathrm{WM}} = 1) &\propto -\log \Pr(\mathrm{WM}|p) \\ D_p^{\mathrm{WM/GM}}(f_p^{\mathrm{WM/GM}} = 1) &\propto -\log \Pr(\mathrm{WM/GM}|p) \end{aligned} \tag{8.5}$$

$$D_p^{\mathrm{l}}(f_p^l = 0) \propto -\log(1 - \Pr(l|p)), l \in \{\mathrm{WM}, \mathrm{WM/GM}\} \tag{8.6}$$

Note our two surfaces WM-GM and GM-CSF highlights at different intensity ranges, it makes sense to adjust the contrast of image when computing the boundary terms $V(\cdot)$ in Eq.(8.4). More specifically, the intensity is transformed by a sigmoid function controlled by parameters α^l and β^l in Eq.(8.7), which enhances the contrast roughly within the range $[\beta^l - 3\alpha^l, \beta^l + 3\alpha^l]$. Here $l \in \{\mathrm{WM}, \mathrm{GM/WM}\}$. These parameters are set to $(\alpha^{\mathrm{WM}}, \beta^{\mathrm{WM}}) = (10, 180)$ and $(\alpha^{\mathrm{GM/WM}}, \beta^{\mathrm{GM/WM}}) = (30, 120)$ by visually checking the training set images. Since all images' histograms are matched to the reference mean training brain histogram, the above contrast adjustment parameters should be robust to different scans.

$$\bar{I}_p^l \propto \frac{1}{1+\exp(-\frac{I_p-\beta^l}{\alpha^l})} \tag{8.7}$$

$$V_{pq}(f_p^l \neq f_q^l) \propto \exp(-(\bar{I}_p^l - \bar{I}_q^l)/\sigma^2) \tag{8.8}$$

$$V_{pq}(f_p^l = f_q^l) = 0 \tag{8.9}$$

The gradient between neighboring voxels p and q is then used as the boundary term (Eq.(8.8)). The more different intensities neighboring voxels have, the smaller the boundary penalty is. On the other hand, if the neighboring voxels have very similar intensities, then the boundary penalty will be large, strongly encouraging two voxels to share the same labeling. The parameter σ in Eq.(8.8) is set to 0.1 based on the training set scans.

8.3.4 Brain segmentation: results

One example segmentation from different views is shown in Fig.8.4. The three columns show the original, manual contour and the GVF shape prior segmentations respectively. We can see the segmentation aligns well with manual contour. Fig.8.5 shows the 3D rendering of WM-GM and GM-CSF surfaces from three subjects. The proposed method is shown to agree well with the manual contour in 3D space.

Quantitatively we compare the Dice coefficient (DSC) of WM and GM regions with various published methods (Sec.8.3.2). Table.8.1 lists the DSC mean and standard deviation of various state-of-art methods and the proposed method. Our method is giving the best accuracy on GM, with a large margin compared to most of

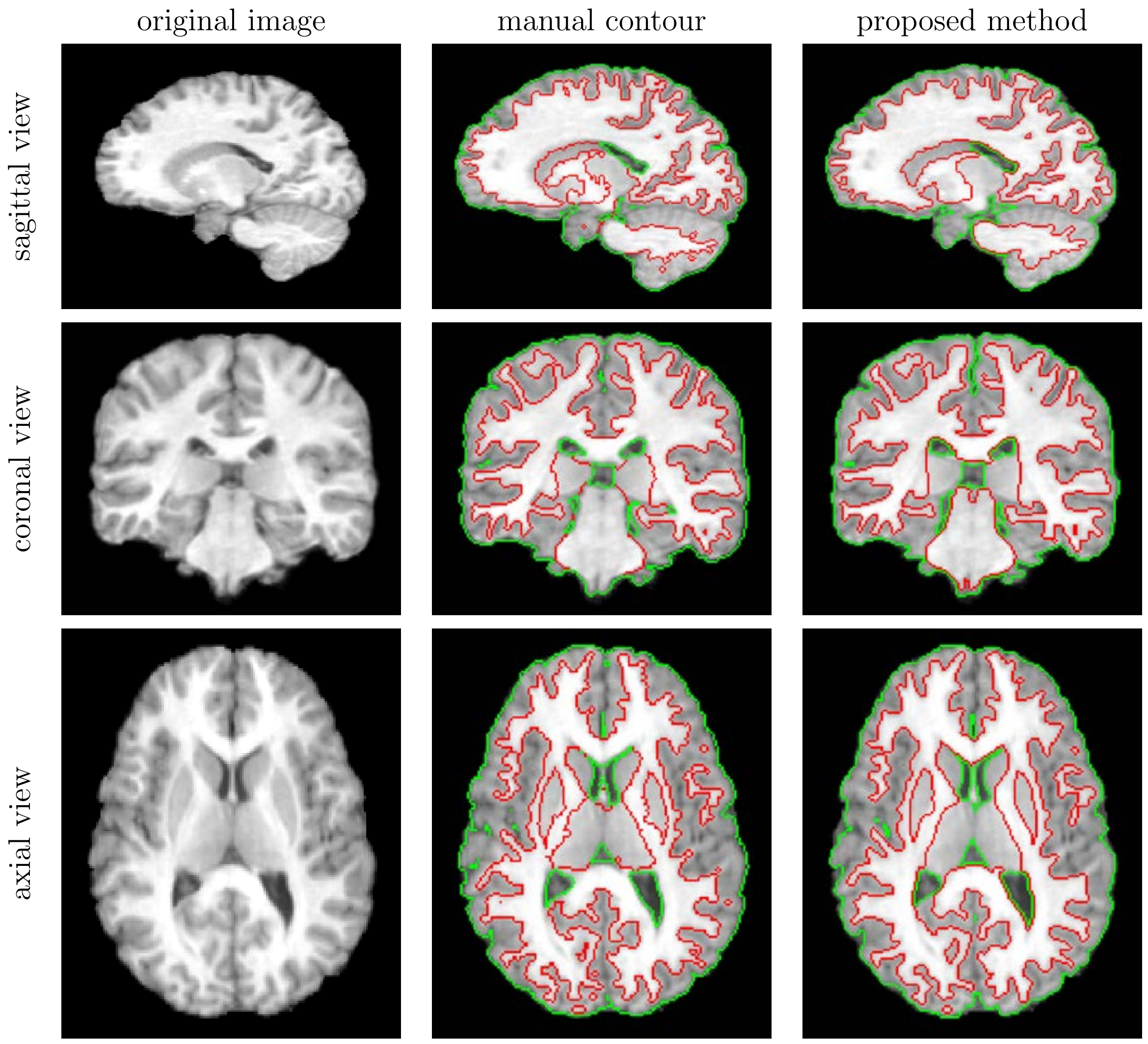


Figure 8.4: Example segmentation using proposed GVF shape prior. The red contour is WM-GM boundary. The green contour is GM-CSF boundary.

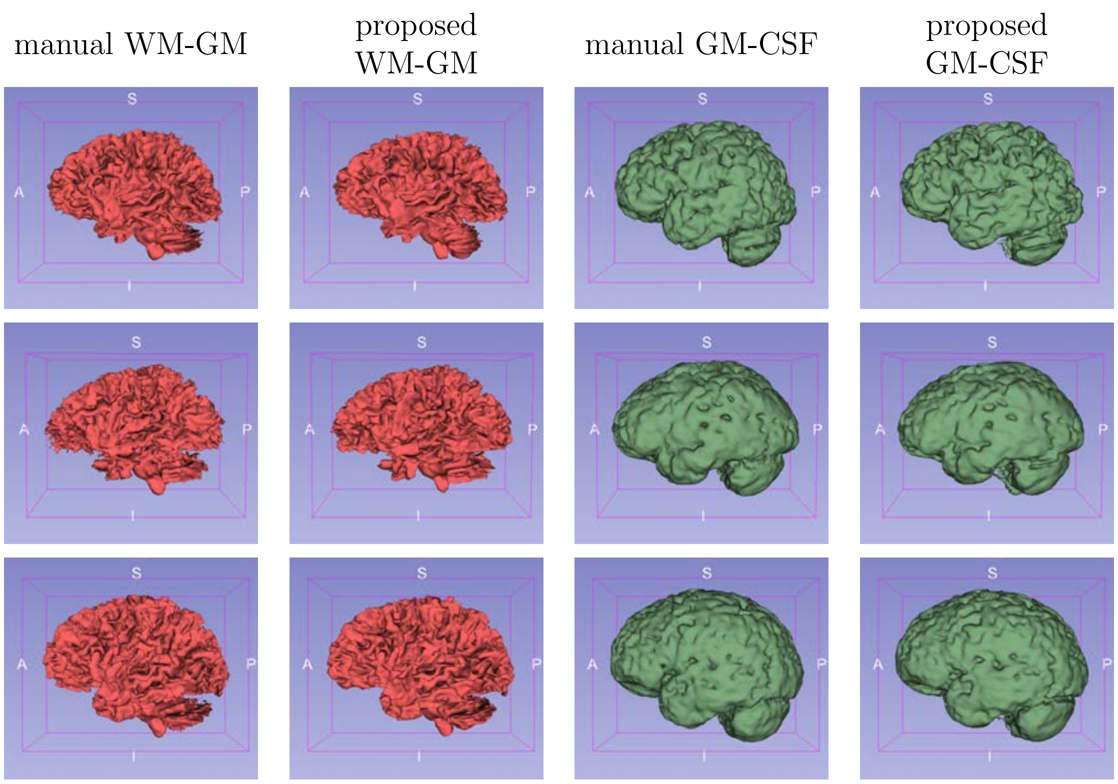


Figure 8.5: 3D renderings of WM-GM and GM-CSF surfaces of manual contour and the proposed GVF shape prior segmentation, from 3 different subjects.

the other methods. The WM accuracy is also very competitive. In fact, if we visualize the results in a column chart (Fig.8.6), it is obvious that the proposed method gives accurate segmentation of both WM and GM. In contrast, most other methods performs well on WM, but relatively poorly on GM. The surface distance error metric (ASSD) is not reported since many published methods did not report them.

8.3.5 Brain segmentation: sensitivity to pre-segmentation

Since the shape prior is defined based on a pre-segmentation, it is natural to wonder how sensitive the method is with respect to the pre-segmentation. To study the sensitivity, we artificially warped the image on a $20 \times 20 \times 20$ grid using B-Spline.

Table 8.1: Proposed method's brain tissue segmentation DSC compared to other state-of-art methods.

method	WM	GM
FAST [113]	0.89 ± 0.02	0.74 ± 0.04
SPM5 [6]	0.86 ± 0.02	0.68 ± 0.07
SPM8 [5]	0.88 ± 0.01	0.81 ± 0.02
GAMIXTURE [92]	0.87 ± 0.02	0.78 ± 0.08
ANN [91]	0.87 ± 0.03	0.70 ± 0.07
FCM [69]	0.88 ± 0.03	0.70 ± 0.06
KNN [22]	0.86 ± 0.03	0.79 ± 0.03
SVPASEG [93]	0.86 ± 0.02	0.81 ± 0.03
FANTASM [68]	0.88 ± 0.03	0.71 ± 0.06
PVC [78]	0.83 ± 0.07	0.70 ± 0.08
Awate *et al.* [8]	**0.89** ± **0.02**	0.81 ± 0.04
Akselrod-Ballin *et al.* [3]	0.87	0.86
Bricq *et al.*[17]	0.87 ± 0.02	0.80 ± 0.06
Wels *et al.* [104]	0.87 ± 0.05	0.83 ± 0.12
proposed	0.87 ± 0.03	**0.88** ± **0.02**

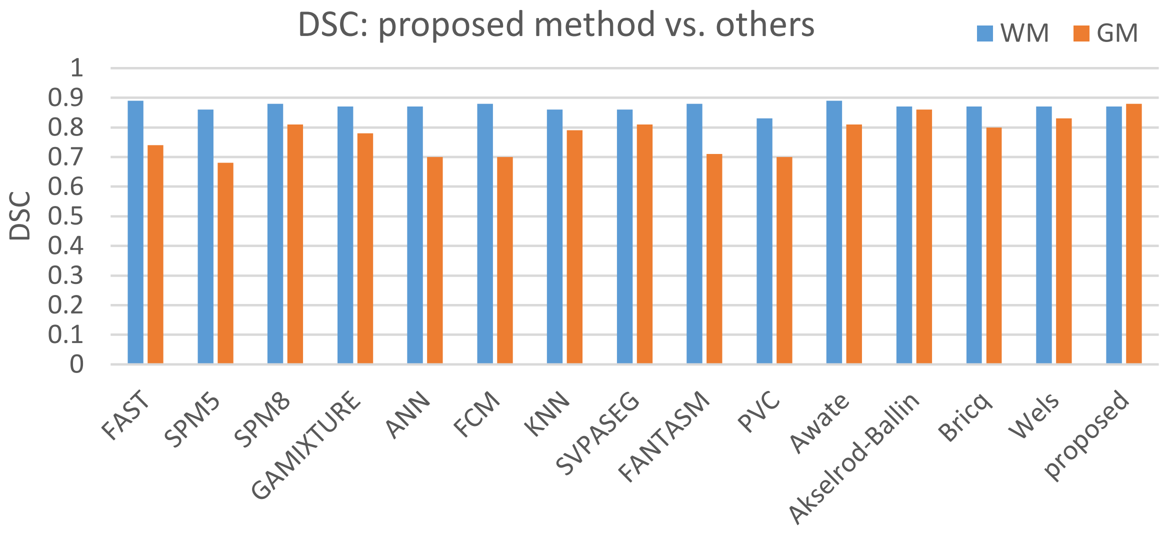


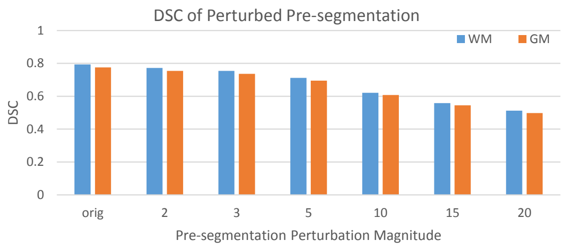
Figure 8.6: DSC of two tissue types (WM and GM) for each method listed in Table.8.1. The proposed method achieves accurate segmentation on both types of tissue, while most other methods are performing well on WM, but relatively poor on GM.

At each B-Spline grid point, a random Gaussian noise deform force with zero mean and standard deviation of σ^{ptb} along each dimension is applied. Then the proposed GVF shape prior segmentation algorithm is run using pre-segmentation perturbed with various perturbation magnitudes σ^{ptb}. The DSC and ASSD between the perturbed and the manual expert contour are measured to quantify the deformation magnitude (reported as 'pre-seg DSC/ASSD' in Table.8.2). The DSC and ASSD between output of the proposed algorithm and the manual expert contour are reported to validate how sensitive the proposed method is to pre-segmentation (reported as 'final DSC/ASSD' in Table.8.2).

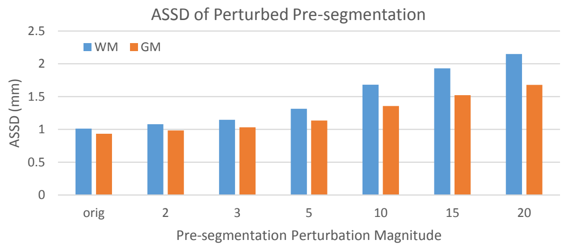
As the perturbation magnitude parameter σ^{ptb} increases from 0 to 5, the proposed method's final output barely change at all, even though the DSC between the deformed pre-segmentation and manual contour dropped by about 0.04 in DSC for both WM and GM. As the perturbation magnitude comes even larger, the proposed method accuracy starts to decrease more significantly. But even at the largest perturbation magnitude ($\sigma^{ptb} = 20$), the DSC only drops around 0.03 for both WM and GM, while the deformed pre-segmentation only has 0.512 and 0.498 DSC. In terms of surface distance error ASSD, the pre-segmentation is already perturbed by 1.14 mm on WM and 0.75 mm on GM, but the output ASSD only drops by 0.22 and 0.30 mm. Fig.8.7 visually conveys this trend. In the first row of Fig.8.7, As the deformed pre-segmentation rapidly deviates from the original one (Fig.8.7a,8.7b), the final proposed method output accuracy only decreases mildly (Fig.8.7c,8.7d).

Table 8.2: Sensitivity of proposed method with respect to pre-segmentation deformation. The first column is the perturbation magnitude parameter σ^{ptb} value. The surface error (ASSD) values are reported in unit mm.

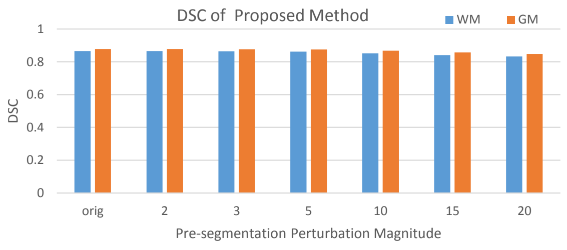
ptb mag	pre-seg DSC		final DSC		pre-seg ASSD		final ASSD	
	WM	GM	WM	GM	WM	GM	WM	GM
0	0.794	0.775	0.866	0.878	1.01	0.93	0.59	0.67
2	0.772	0.754	0.865	0.877	1.08	0.99	0.59	0.68
3	0.754	0.736	0.864	0.877	1.15	1.03	0.60	0.68
5	0.712	0.696	0.862	0.876	1.32	1.13	0.62	0.71
10	0.621	0.608	0.852	0.868	1.68	1.36	0.69	0.79
15	0.559	0.545	0.841	0.857	1.93	1.52	0.75	0.88
20	0.512	0.498	0.832	0.848	2.15	1.68	0.81	0.97



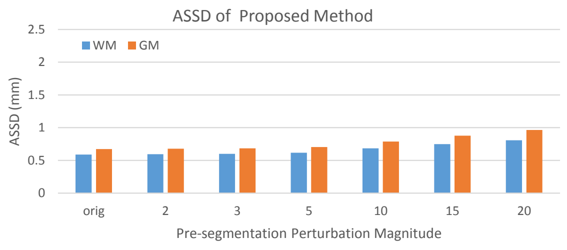
(a) DSC of perturbed pre-seg vs. manual contour



(b) ASSD of perturbed pre-seg vs. manual contour



(c) DSC of proposed method using perturbed pre-seg



(d) ASSD of proposed method using pertrubed pre-seg

Figure 8.7: Sensitivity of the proposed method with respect to pre-segmentation deformation (generated from Table.8.2.). As the deformation greatly deviates from the pre-segmentation, the proposed method only suffers a mild decrease in performance, for both tissue types and both metrics.

8.3.6 Bladder/prostate segmentation: data and compared methods

For prostate and bladder segmentation, 21 volumetric CT images from different patients with prostate cancer were used. The image spacing ranged from $0.98 \times 0.98 \times 3.00 mm^3$ to $1.60 \times 1.60 \times 3.00 mm^3$. Expert manual contours of bladder and prostate are available for each scan. Eight scans are randomly selected to be used as training set. The other 13 scans are used as testing set. All parameters are tuned on training set. All reported results are from testing set.

The proposed GVF shape prior segmentation method is compared to the mesh-based method proposed by Song *et al.* [86]. They iteratively evolve the bladder and prostate meshes by using the LOGISMOS methods. In each iteration, the bladder and prostate meshes are first evolved independently, and then deformed jointly by defining a "mutually interacting region" to ensure the two object meshes do not intersect. A soft shape prior is also incorporated in the prostate mesh to encourage the shape conforming to a trained model.

8.3.7 Bladder/prostate segmentation: workflow

8.3.7.1 Pre-segmentation

We use the same pre-segmentation used by Qi *et al.* [86]. The prostate shape is relatively consistent across scans, thus a point distribution model built from training images are aligned to each test scan. The Procrustes analysis method is employed to get the prostate pre-segmentation. Due to the large shape variations in bladder, a geodesic active contour method is used to generate the bladder pre-segmentation. The prostate and bladder are set to be two exclusive objects with a minimum separation distance of 0 mm.

8.3.7.2 Energy term design

Similar to the brain tissue segmentation, a random forest with a four-dimensional feature vector (normalized x/y/z coordinates and intensity) is trained to generate a probability map of bladder and prostate. Due to the poor soft-tissue contrast in CT scans, a Chan-Vese model [18] is also trained to more accurately model the intensity difference between prostate and bladder. The two probability maps generated by random forest and Chan-Vase model are added together with equal weight as the data term ($D_p(\cdot)$ in Eq.(8.4)). The smoothness term design is the same as in brain tissue segmentation (Sec.8.3.3).

8.3.8 Bladder/prostate segmentation: results

The proposed method's accuracy is presented in Table.8.3. Our method is giving better accuracy on bladder based on both DSC and ASSD metrics. On prostate, our method is giving a better DSC but slightly worse ASSD values. A 2-tailed paired t-test shows that our method is statistically significantly different from Song *et al.* [86] on bladder ($p < 0.05$) on both DSC and ASSD metric. On prostate, both methods are not statistically different.

Fig.8.8 shows one example of the proposed segmentation and the mesh-based LOGISMOS method. The mesh-based LOGISMOS segmentation gives very irregular surfaces compared to the smooth appearance of the manual contour and the proposed

Table 8.3: Bladder/prostate segmentation accuracy compared to mesh-based LOGISMOS method [86].

metric	object	Song *et al.* [86]	proposed
DSC	bladder	0.933 ± 0.015	**0.941 ± 0.017**
	prostate	0.824 ± 0.036	**0.835 ± 0.015**
ASSD (mm)	bladder	0.89 ± 0.26	**0.79 ± 0.28**
	prostate	**1.48 ± 0.47**	1.64 ± 0.28

method. The spiky error at the top of bladder in the mesh-based segmentation (third column in Fig.8.8) signal high risk of mesh folding, which has to be accounted for by special procedures discussed in Sec.8.1.1. In contrast, the proposed GVF shape prior segmentation gives accurate and smooth segmentation of bladder and prostate.

8.4 Conclusion

We propose the novel GVF shape prior which can be directly embedded in the voxel grid space, avoiding the mesh folding problem fundamentally. The flexible and powerful GVF shape prior can be incorporated in an efficient multi-object MRF formulation. The proposed shape prior segmentation method is validated on the application of brain tissue segmentation (two nesting/including objects), and the bladder/prostate segmentation (two excluding objects). The experiment results demonstrates superior or competitive segmentation accuracy compared to other state-of-art methods.

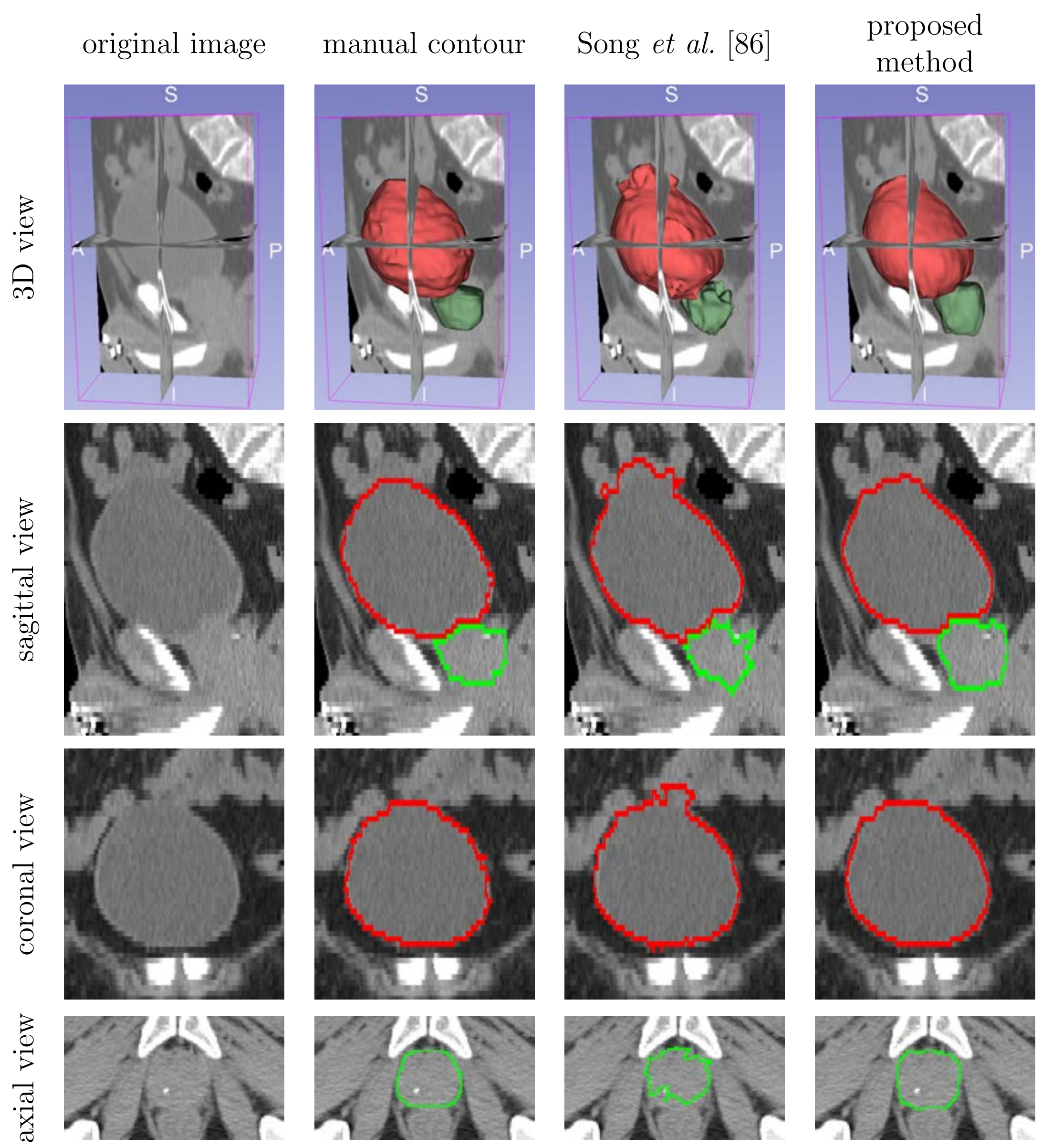


Figure 8.8: Illustrative example of segmentations by the proposed GVF shape prior segmentation method and the mesh-based LOGISMOS method [86]. Red and greeen contour are for bladder and prostate, respectively.

CHAPTER 9
CONCLUSION

In this dissertation, a series of useful labeling problem models incorporating various prior information are proposed and validated. All proposed models are motivated by practical problems. After judicious theoretical manipulations, all proposed formulations admit *efficient* and *globally optimal* solution. We first review the intricate structures of the original problems which inspired the formulations. Then we briefly discuss about why efficient and globally optimal solution is important for natural/medical image applications.

The labeling problem with ordering constraints is motivated by the natural scene labeling problem. Due to its multiple labeling nature, the constraints are complicated and hard to enforce in a graph-cut framework. However, since it is so complicated that it becomes restrictive, we can find its general layout. With this layout in mind, a dynamic programming method with only linear memory and $O(N^{1.5})$ time is devised and optimally solves the problem.

The fact that human makes mistakes while drawing scribbles inspired the robust segmentation project. To achieve the intuitive balance that the same percentage of ignored human scribble should incur at least the same percentage of segmentation energy drop, the ratio energy is proposed. Seemingly hard to optimize, the ratio energy function is transformed into a series of linear energy functions which can be efficiently solved by graph-cut.

The multi-scale technique applied on MVCBCT is inspired by the general observation that the larger the neighborhood is, the more accurate the segmentation is. However, the cost of naively increasing neighborhood size is prohibitive. Connecting every voxel with the over-segmented region it lies in effectively simulates a larger neighborhood while limiting the number of additional pairwise relationships.

Leveraging multi-modality information can be expected to boost the segmentation performance. We enforce the inter-modality agreement prior to make the 4DCT-PET co-segmentation achieve better accuracy than using CT alone. The observation that PET and CT tumor segmentation volumes agree at core and far away from the highly metabolic regions, but differ most around the tumor boundary, leads to the incorporation of location context prior into the overall PET-CT context prior design. The additional location prior improved the co-segmentation accuracy.

Both star-shape prior and gradient vector flow (GVF) shape prior are motivated by removing the resampling step in graph-search, which introduces not only extra processing steps, but also practice difficulties (column-intersection/mesh-folding). Choosing voxels, instead of resampled graph nodes, as the labeling problem variables bring us three fold benefits: 1) avoids resampling; 2) the problem is conceptually simplified from a multi-labeling graph-search problem to a binary-labeling graph-cut problem; 3) the direct correspondence of two subgraphs occupying the same voxel grid enable straightforward multi-object inclusion/exclusion interaction. The other very important factor is the use of consistent digital ray and GVF field to encode the shape prior information directly in the voxel grid space.

By judiciously exploiting the above intricate structures of the problem, we proposed and validated multiple formulations incorporating various prior information. One key feature of the models presented in this dissertation is that they all admit efficient and globally optimal solutions.

With efficient solutions the proposed models can be applied on large scale problems, such as 3D/4D medical image segmentations. All except the robust segmentation problem can be solved by either a single minimum *s*-*t* cut or dynamic programming. For the iterative robust segmentation problem, we proved a strong theoretical bound on the number of iterations, and validated that in practice it also converges in just a few iterations.

With globally optimal solutions, the proposed models become robust to local noise, and easier to debug if unexpected result is returned. More specifically, we can focus on improving the cost/energy term design, instead of facing the dilemma of wondering whether the cost is bad or the optimization algorithm is weak that we are trapped in a bad local minimum.

9.1 Limitations and Future Work

One limitation of the proposed work is that some formulations, such as 4DCT-PET co-segmentation and multi-scale segmentation, requires extra subgraph. For example, in the multi-scale segmentation, an over-segmentation regionwise layer is needed in addition to the original voxelwise layer. This increases running time and memory consumption. One possible future work on this direction is to define a searching region of interest (ROI) based on some initial results on lower resolution. For example, if we can get a rough segmentation in a downsampled-by-2 version of the image, then we can add only the voxels near to the rough segmentation boundary to the graph. All other far-from-boundary voxels can be assigned their lower resolution image labeling.

Another limitation of the proposed work is that even though the global optimality guarantee eliminates the need to worry about the strength of the optimization algorithm, the models still works only as good as the input cost/energy term designs are. How to design good cost term can be a challenging task. In the GVF shape prior project, random forest based machine learning method is used to generate data term cost. For other applications, how to design the cost term, or the system that generates the cost term, is an open and challenging question.

Recently deep learning has become a hot topic. The convolutional neural network (CNN) has been proven very effective at extracting useful abstract features from raw image. Some research has been done to combine CNN with conditional random field (CRF) [108, 114]. The idea is to use CNN for extracting image features and CRF for inferring. All proposed models in this dissertation are actually Markov random field (MRF), a special case of CRF. It would be exciting to combine the state-of-art feature extraction CNN with the also state-of-art inferring MRF.

All models proposed in this dissertation make use of unary and pairwise terms. For unary term, manual expert tracings can be straightforwardly used as training labels. But it is challenging to learn pairwise terms due to the lack of training label reference. How to effectively set pairwise term is another future work direction.

REFERENCES

[1] Internet Brain Segmentation Repository.

[2] a.C. Jalba, M.H.F. Wilkinson, and J.B.T.M. Roerdink. CPM: a deformable model for shape recovery and segmentation based on charged particles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10):1320–1335, 2004.

[3] Ayelet Akselrod-Ballin, Meirav Galun, John Moshe Gomori, Achi Brandt, and Ronen Basri. Prior knowledge driven multiscale segmentation of brain MRI. *Medical image computing and computer-assisted intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention*, 10(Pt 2):118–126, 2007.

[4] Xiaobo An and Fabio Pellacini. AppProp: all-pairs appearance-space edit propagation. *ACM Transactions on Graphics (TOG)*, 27:40, 2008.

[5] John Ashburner, Gareth Barnes, C Chen, Jean Daunizeau, Guillaume Flandin, Karl Friston, D Gitelman, S Kiebel, J Kilner, V Litvak, and Others. SPM8 manual. *Functional Imaging Laboratory, Institute of Neurology*, 2012.

[6] John Ashburner and Karl J. Friston. Unified segmentation. *NeuroImage*, 26(3):839–851, 2005.

[7] MS Aslan, Asem Ali, and Dongqing Chen. 3D vertebrae segmentation using graph cuts with shape prior constraints. In *International Conference on Image Processing (ICIP)*, pages 2193–2196, 2010.

[8] Suyash P. Awate, Tolga Tasdizen, Norman Foster, and Ross T. Whitaker. Adaptive Markov modeling for mutual-information-based, unsupervised MRI brain-tissue classification. *Medical Image Analysis*, 10(5):726–739, 2006.

[9] Ulas Bagci, Jayaram K. Udupa, Neil Mendhiratta, Brent Foster, Ziyue Xu, Jianhua Yao, Xinjian Chen, and Daniel J. Mollura. Joint segmentation of anatomical and functional images: Applications in quantification of lesions from PET, PET-CT, MRI-PET, and MRI-PET-CT images. *Medical Image Analysis*, 17(8):929–945, 2013.

[10] Junjie Bai, Mohammad Saleh Miri, Yinxiao Liu, Punam Saha, Mona Garvin, and Xiaodong Wu. Graph-based optimal multi-surface segmentation with a star-shaped prior: Application to the segmentation of the optic disc and cup. In *2014 IEEE 11th International Symposium on Biomedical Imaging (ISBI)*, pages 525–528. IEEE, apr 2014.

[11] Junjie Bai, Qi Song, Sudershan K. Bhatia, and Xiaodong Wu. Globally optimal lung tumor co-segmentation of 4D CT and PET images. In Sebastien Ourselin and David R. Haynor, editors, *SPIE Medical Imaging*, volume 8669, page 86690W. International Society for Optics and Photonics, mar 2013.

[12] Y. Boykov and G. Funka-Lea. Graph cuts and efficient n-d image segmentation. *International Journal of Computer Vision*, 70(2):109–131, 2006.

[13] Y. Boykov and G. Funka-Lea. Graph cuts and efficient N-D image segmentation. *International Journal of Computer Vision*, 70(2):109–131, 2006.

[14] Y. Boykov and MP Jolly. Interactive organ segmentation using graph cuts. *Medical Image Computing and Computer-Assisted Intervention*, pages 147–175, 2000.

[15] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1222–1239, 2001.

[16] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on pattern analysis and machine intelligence*, pages 1222–1239, 2001.

[17] S. Bricq, Ch Collet, and J. P. Armspach. Unifying framework for multimodal brain MRI segmentation based on Hidden Markov Chains. *Medical Image Analysis*, 12(6):639–652, 2008.

[18] Tony F Chan and Luminita A Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, 2001.

[19] Jinhee Chun, Matias Korman, Martin Nöllenburg, and Takeshi Tokuyama. Consistent digital rays. *Discrete & computational geometry*, 42(3):359–378, 2009.

[20] D Comaniciu and P Meer. Mean shift: A robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence*, 2002.

[21] T Cour, F Benezit, and J Shi. Spectral segmentation with multiscale graph decomposition. *Computer Vision and Pattern Recognition*, 2005.

[22] Renske de Boer, Henri A. Vrooman, Fedde van der Lijn, Meike W. Vernooij, M. Arfan Ikram, Aad van der Lugt, Monique M B Breteler, and Wiro J. Niessen. White matter lesion extension to automatic brain tissue segmentation on MRI. *NeuroImage*, 45(4):1151–1161, 2009.

[23] A. Delong and Y. Boykov. Globally optimal segmentation of multi-region objects. *ICCV*, 2009.

[24] Andrew Delong and Yuri Boykov. Globally optimal segmentation of multi-region objects. In *Computer Vision and Pattern Recognition*, number October, 2009.

[25] Werner Dinkelbach. On Nonlinear Fractional Programming. *Management Science*, 13(7):492–498, 1967.

[26] Jana Dornheim, Heiko Seim, Bernhard Preim, Ilka Hertel, and Gero Strauss. Segmentation of Neck Lymph Nodes in CT Datasets with Stable 3D Mass-Spring Models. Segmentation of Neck Lymph Nodes. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2006.

[27] Jan Ehrhardt, Alexander Schmidt-Richberg, and Heinz Handels. Simultaneous segmentation and motion estimation in 4D-CT data using a variational approach. *Proceedings of SPIE*, 6914:691437–691437–10, 2008.

[28] P.F. Felzenszwalb and D.P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167181, 2004.

[29] PF Felzenszwalb and DP Huttenlocher. Efficient belief propagation for early vision. *International Journal of Computer Vision*, 2006.

[30] P.F. Felzenszwalb and O. Veksler. Tiered scene labeling with dynamic programming. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3097–3104. IEEE.

[31] G. Gallo, M. D Grigoriadis, and R. E Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, 18:30, 1989.

[32] G. Gallo, M. D Grigoriadis, and R. E Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, 18:30, 1989.

[33] Mona Kathryn Garvin, Michael David Abràmoff, Xiaodong Wu, Senior Member, Stephen R Russell, Trudy L Burns, and Milan Sonka. Automated 3-D Intraretinal Layer Segmentation of Macular Spectral-Domain Optical Coherence Tomography Images. *Search*, 28(9):1436–1447, 2009.

[34] Mona Kathryn Garvin, Michael David Abràmoff, Xiaodong Wu, Stephen R Russell, Trudy L Burns, and Milan Sonka. Automated 3-D intraretinal layer segmentation of macular spectral-domain optical coherence tomography images. *IEEE transactions on medical imaging*, 28(9):1436–47, September 2009.

[35] M Gondran, M Minoux, and S Vajda. Graphs and algorithms. pages 636–641, 1984.

[36] Dongfeng Han, John Bayouth, Qi Song, Aakant Taurani, Milan Sonka, John Buatti, and Xiaodong Wu. Globally optimal tumor segmentation in PET-CT images: a graph-based co-segmentation method. In *Information Processing in Medical Imaging*, volume 22, pages 245–56, January 2011.

[37] D. Hoiem, A.A. Efros, and M. Hebert. Geometric context from a single image. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, page 654661, 2005.

[38] Hossam Isack, Yuri Boykov, and Olga Veksler. A-expansion for multiple "hedgehog" shapes. feb 2016.

[39] H. Ishikawa. Exact optimization for markov random fields with convex priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1333–1336, 2003.

[40] H. Ishikawa. Higher-order clique reduction in binary graph cut. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2993–3000, June 2009.

[41] H Ishikawa. Higher-Order Clique Reduction Without Auxiliary Variables. *Computer Vision and Pattern Recognition (CVPR)*, 2014.

[42] Dagmar Kainmueller, Hans Lamecker, Markus O Heller, Britta Weber, Hans-Christian Hege, and Stefan Zachow. Omnidirectional displacements for deformable surfaces. *Medical image analysis*, 17(4):429–41, may 2013.

[43] E Kim, H Li, and X Huang. A hierarchical image clustering cosegmentation framework. In *Computer Vision and Pattern Recognition*, 2012.

[44] TH Kim, KM Lee, and SU Lee. Nonparametric higher-order learning for interactive segmentation. In *Computer Vision and Pattern Recognition*, 2010.

[45] Y Kitamura, Y Li, W Ito, and H Ishikawa. Coronary Lumen and Plaque Segmentation from CTA Using Higher-Order Shape Prior. *Medical Image Computing and Computer-Assisted Intervention*, 2014.

[46] Tobias Klinder, Jorn Ostermann, Matthias Ehm, Astrid Franz, Reinhard Kneser, and Cristian Lorenz. Automated model-based vertebra detection, identification, and segmentation in CT images. *Medical Image Analysis*, 13(3):471–482, 2009.

[47] P. Kohli and P. H S Torr. Dynamic graph cuts for efficient inference in markov random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2079–2088, 2007.

[48] V Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2006.

[49] V Kolmogorov and C Rother. Minimizing nonsubmodular functions with graph cuts-a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1274–1279, 2007.

[50] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? *IEEE transactions on pattern analysis and machine intelligence*, 26(2):147–59, February 2004.

[51] Kyungmoo Lee, Meindert Niemeijer, Mona K Garvin, Young H Kwon, Milan Sonka, and Michael D Abramoff. Segmentation of the optic disc in 3-D OCT scans of the optic nerve head. *IEEE transactions on medical imaging*, 29(1):159–68, January 2010.

[52] Kang Li, Xiaodong Wu, Danny Z Chen, and Milan Sonka. Optimal surface segmentation in volumetric images–a graph-theoretic approach. *IEEE transactions on pattern analysis and machine intelligence*, 28(1):119–34, January 2006.

[53] Yin Li, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum. Lazy snapping. *ACM Trans. Graph.*, 23(3):303–308, August 2004.

[54] Yong Li, Tao Ju, and SM Hu. Instant propagation of sparse edits on images and videos. *Computer Graphics Forum*, 29:2049–2054, 2010.

[55] Jiangyu Liu, Jian Sun, and HY Shum. Paint selection. *ACM Transactions on Graphics (ToG)*, 28(3):69, 2009.

[56] X. Liu, O. Veksler, and J. Samarabandu. Order-Preserving moves for Graph-Cut-Based optimization. *IEEE transactions on pattern analysis and machine intelligence*, page 11821196, 2010.

[57] Xiaoqing Liu, O. Veksler, and J. Samarabandu. Graph cut with ordering constraints on labels and its applications. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pages 1–8. IEEE, June 2008.

[58] Y. Liu, G. Liang, and P. K Saha. A new multi-object image thresholding method based on correlation between object class uncertainty and intensity gradient. *Medical physics*, 2012.

[59] H Lombaert and Y Sun. A multilevel banded graph cuts method for fast image segmentation. In *IEEE International Conference on Computer Vision (ICCV)*, 2005.

[60] Sebastien Martin, Jocelyne Troccaz, and Vincent Daanen. Automated segmentation of the prostate in 3D MR images using a probabilistic atlas and a spatially constrained deformable model. *Medical Physics*, 37(4):1579, mar 2010.

[61] Tim McInerney and Demetri Terzopoulos. Deformable models in medical image analysis: a survey. *Medical Image Analysis*, 1(2):91–108, jun 1996.

[62] Eric N. Mortensen and William a. Barrett. Intelligent scissors for image composition. *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques - SIGGRAPH '95*, 84602(801):191–198, 1995.

[63] Ursula Nestle, Stephanie Kremp, and Anca Ligia Grosu. Practical integration of [18F]-FDG-PET and PET-CT in the planning of radiotherapy for non-small cell lung cancer (NSCLC): The technical basis, ICRU-target volumes, problems, perspectives. *Radiotherapy and Oncology*, 81(2):209–225, 2006.

[64] Ipek Oguz and Milan Sonka. LOGISMOS-B: layered optimal graph image segmentation of multiple objects and surfaces for the brain. *IEEE transactions on medical imaging*, 33(6):1220–35, June 2014.

[65] James B. Orlin. Max flows in O(nm) time, or better. In *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing - STOC '13*, page 765, New York, New York, USA, June 2013. ACM Press.

[66] Joo Young Park, Tim McInerney, Demetri Terzopoulos, and Myoung Hee Kim. A non-self-intersecting adaptive deformable surface for complex boundary extraction from volumetric images. *Computers and Graphics (Pergamon)*, 25(3):421–440, 2001.

[67] Jens Petersen, Mads Nielsen, Pechin Lo, Lars Haug Nordenmark, Jesper Holst Pedersen, Mathilde Marie Winkler Wille, Asger Dirksen, and Marleen de Bruijne. Optimal surface segmentation using flow lines to quantify airway abnormalities in chronic obstructive pulmonary disease. *Medical image analysis*, 18(3):531–41, apr 2014.

[68] Dzung L Pham. Robust Fuzzy Segmentation of Magnetic Resonance Images. In *IEEE Symposium on Computer-Based Medical Systems*, pages 127–131, 2001.

[69] Dzung L. Pham. Spatial Models for Fuzzy Clustering. *Computer Vision and Image Understanding*, 84(2):285–297, 2001.

[70] Jean Philippe Pons and Jean Daniel Boissonnat. Delaunay deformable models: Topology-adaptive meshes based on the restricted delaunay triangulation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007.

[71] E Rietzel and GTY Chen. Four-dimensional image-based treatment planning: Target volume segmentation and dose calculation in the presence of respiratory motion. *International Journal of Radiation Oncology Biology Physics*, 61(5):1535–1550, 2005.

[72] Carsten Rother, V Kolmogorov, and Andrew Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (TOG)*, 1(212):309–314, 2004.

[73] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 309–314. ACM, 2004.

[74] Siegfried Schaible. Fractional Programming . II , on Dinkelbach's Algorithm. *Management Science*, 22(8):868–873, 1976.

[75] O Sener, K Ugur, and AA Alatan. Error-tolerant interactive image segmentation using dynamic and iterated graph-cuts. *Proceedings of the 2nd ACM international workshop on Interactive multimedia on mobile and portable devices*, 2012.

[76] Leila Shafarenko, Maria Petrou, and Josef Kittler. Automatic watershed segmentation of randomly textured color images. *Image Processing, IEEE Transactions on*, 6(11):1530–1544, 1997.

[77] E Sharon, A Brandt, and R Basri. Fast multiscale image segmentation. In *Computer Vision and Pattern Recognition*, 2000.

[78] D W Shattuck, S R Sandor-Leahy, K a Schaper, D a Rottenberg, and R M Leahy. Magnetic resonance image tissue classification using a partial volume model. *NeuroImage*, 13(5):856–876, 2001.

[79] J. Shi and J. Malik. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE transactions on*, 22(8):888–905, 2000.

[80] Q Song, J Bai, M Garvin, M Sonka, J Buatti, and X Wu. Optimal Multiple Surface Segmentation With Shape and Context Priors. *IEEE transactions on medical imaging*, (99), November 2012.

[81] Q. Song, X. Wu, and Y. Liu. Simultaneous searching of globally optimal interacting surfaces with shape priors. In *Computer Vision and Pattern Recognition*, pages 2879–2886, 2010.

[82] Q. Song, X. Wu, Y. Liu, M. Sonka, and M. Garvin. Simultaneous searching of globally optimal interacting surfaces with shape priors. In *CVPR*, 2010.

[83] Qi Song, Junjie Bai, Dongfeng Han, Sudershan Bhatia, Wenqing Sun, William Rockey, John E Bayouth, John M Buatti, and Xiaodong Wu. Optimal co-segmentation of tumor in PET-CT images with context information. *IEEE transactions on medical imaging*, 32(9):1685–97, September 2013.

[84] Qi Song, Mingqing Chen, Junjie Bai, Milan Sonka, and Xiaodong Wu. Surface-region context in optimal multi-object graph-based segmentation: robust delineation of pulmonary tumors. *Information processing in medical imaging : proceedings of the ... conference*, 22:61–72, January 2011.

[85] Qi Song, Mingqing Chen, Junjie Bai, Milan Sonka, and Xiaodong Wu. Surface-region context in optimal multi-object graph-based segmentation: robust delineation of pulmonary tumors. *In Information Processing in Medical Imaging, pp. 61-72. Springer Berlin/Heidelberg*, 22:61–72, January 2011.

[86] Qi Song, Yunlong Liu, Yunlong Liu, and PK Saha. Graph search with appearance and shape information for 3-D prostate and bladder segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, volume 13, pages 172–180, 2010.

[87] Qi Song, Xiaodong Wu, Yunlong Liu, Mark Smith, John Buatti, and Milan Sonka. Optimal graph search segmentation using arc-weighted graph for simultaneous surface detection of bladder and prostate. In *Medical Image Computing and Computer-Assisted Intervention*, 2009.

[88] E. Strekalovskiy and D. Cremers. Generalized ordering constraints for multilabel optimization. In *IEEE International Conference on Computer Vision (ICCV)*. 2011.

[89] Kartic Subr, Sylvain Paris, Cyril Soler, and Jan Kautz. Accurate Binary Image Selection from Inaccurate User Input. *Computer Graphics Forum*, 32, 2013.

[90] M. Teschner, S. Kimmerle, B. Heidelberger, G. Zachmann, L. Raghupathi, A. Fuhrmann, M. P. Cani, F. Faure, N. Magnenat-Thalmann, W. Strasser, and P. Volino. Collision detection for deformable objects. *Computer Graphics Forum*, 24(1):61–81, 2005.

[91] Dan Tian and Fan Linan. A brain MR images segmentation method based on SOM neural network. *Engineering, 2007. ICBBE 2007. The 1st*, (20052001):686–689, 2007.

[92] J. Tohka, E. Krestyannikov, I. D. Dinov, A. M. Graham, D. W. Shattuck, U. Ruotsalainen, and A. W. Toga. Genetic algorithms for finite mixture model based tissue classification in brain MRI. *IEEE transactions on medical imaging*, 26(5):696–711, 2007.

[93] Jussi Tohka, Ivo D. Dinov, David W. Shattuck, and Arthur W. Toga. Brain MRI tissue classification based on local Markov random fields. *Magnetic Resonance Imaging*, 28(4):557–573, 2010.

[94] Hanghang Tong, Christos Faloutsos, and Jia-yu Pan. Fast Random Walk with Restart and Its Applications. In *Sixth International Conference on Data Mining (ICDM'06)*, pages 613–622. IEEE, December 2006.

[95] Sergi Valverde, Arnau Oliver, Mariano Cabezas, Eloy Roura, and Xavier Lladó. Comparison of 10 brain tissue segmentation methods using revisited IBSR annotations. *Journal of Magnetic Resonance Imaging*, 41(1):93–101, 2015.

[96] K Van Leemput, F Maes, D Vandermeulen, and P Suetens. Automated model-based bias field correction of MR images of the brain. *IEEE transactions on medical imaging*, 18(10):885–896, 1999.

[97] O. Veksler. Star shape prior for graph-cut image segmentation. *ECCV*, 2008.

[98] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001.*, volume 1, pages I–511. IEEE, 2001.

[99] Nhat Vu and B.S. Manjunath. Shape prior segmentation of multiple objects with graph cuts. *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, jun 2008.

[100] H Wang, N Ray, and H Zhang. Graph-cut optimization of the ratio of functions and its application to image segmentation. *ICIP*, 2008.

[101] Hui Wang, Hong Zhang, and Nilanjan Ray. Adaptive shape prior in graph cut image segmentation. *Pattern Recognition*, 46(5):1409–1414, may 2013.

[102] J Weese, M Kaus, and C Lorenz. Shape constrained deformable models for 3D medical image segmentation. *Information Processing in Medical Imaging*, 2001.

[103] Elisabeth Weiss, Krishni Wijesooriya, S Vaughn Dill, and Paul J Keall. Tumor and normal tissue motion in the thorax during respiration: Analysis of volumetric and positional variations using 4D CT. *International journal of radiation oncology, biology, physics*, 67(1):296–307, January 2007.

[104] Michael Wels, Yefeng Zheng, Martin Huber, Joachim Hornegger, and Dorin Comaniciu. A discriminative model-constrained EM approach to 3D MRI brain tissue classification and intensity non-uniformity correction. *Physics in medicine and biology*, 56(11):3269–300, 2011.

[105] X. Wu and D. Chen. Optimal net surface problems with applications. *Automata, Languages and Programming*, pages 775–775, 2002.

[106] Xiaodong Wu and DZ Chen. Optimal net surface problems with applications. *Automata, Languages and Programming*, pages 1029–1042, 2002.

[107] C Xu and JL Prince. Snakes, shapes, and gradient vector flow. *Image Processing, IEEE Transactions on*, 7(3):359–369, 1998.

[108] Puyang Xu and R. Sarikaya. Convolutional neural network based triangular crf for joint intent detection and slot filling. In *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pages 78–83, Dec 2013.

[109] Yin Yin, Qi Song, and Milan Sonka. Electric field theory motivated graph construction for optimal medical image segmentation. *Graph-Based Representations in Pattern Recognition*, 5534:334–342, 2009.

[110] Yin Yin, Xiangmin Zhang, R. Williams, Xiaodong Wu, D.D. Anderson, and M. Sonka. LOGISMOS–Layered Optimal Graph Image Segmentation of Multiple Objects and Surfaces: Cartilage Segmentation in the Knee Joint. *Medical Imaging, IEEE Transactions on*, 29(12):2023 –2037, dec 2010.

[111] Sheng You, Esra Ataer-cansizoglu, and Deniz Erdogmus. A Novel Application of Principal Surfaces to Segmentation in 4D-CT for Radiation Treatment Planning. *Direct*.

[112] Eun Young Kim and Hans J Johnson. Robust multi-site MR data processing: iterative optimization of bias correction, tissue classification, and registration. *Frontiers in neuroinformatics*, 7(November):29, 2013.

[113] Y. Zhang, M. Brady, and S Smith. Segmentation of brain MR images through a hidden Markov random field model and the expectation-maximization algorithm. *IEEE Trans Med Imag*, 20(1):45–57, 2001.

[114] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip H. S. Torr. Conditional random fields as recurrent neural networks. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.