

January 2018

# SOL: Segmentation with Overlapping Labels

Karin Ng

*The University of Western Ontario*

Supervisor

Boykov, Yuri Y.

*The University of Western Ontario*

Graduate Program in Computer Science

A thesis submitted in partial fulfillment of the requirements for the degree in Master of Science

© Karin Ng 2018

Follow this and additional works at: <https://ir.lib.uwo.ca/etd>

 Part of the [Computer Sciences Commons](#)

---

## Recommended Citation

Ng, Karin, "SOL: Segmentation with Overlapping Labels" (2018). *Electronic Thesis and Dissertation Repository*. 5185.  
<https://ir.lib.uwo.ca/etd/5185>

This Dissertation/Thesis is brought to you for free and open access by Scholarship@Western. It has been accepted for inclusion in Electronic Thesis and Dissertation Repository by an authorized administrator of Scholarship@Western. For more information, please contact [tadam@uwo.ca](mailto:tadam@uwo.ca).

# Abstract

Image segmentation is a fundamental problem in Computer Vision which involves segmenting an image into two or more segments. These segments usually correspond to objects of interest in the image, i.e. liver, kidney's etc. The classic approach to this problem segments the image into mutually exclusive segments. However, this approach is not well-suited when segmenting overlapping objects, e.g. cells, or when segmenting a single object into multiple parts that are not necessarily mutually exclusive. Moreover, we show that optimization methods for multi-part object segmentation with different priors/constraints may better avoid local minima in case of a relaxation allowing parts to overlap.

We propose a novel segmentation model, i.e. Segmentation with Overlapping Labels (**SOL**), which allows for the objects' multiple parts to overlap. This aids in overcoming the aforementioned issue of local minima with standard optimization approaches. We prove that SOL is an NP-hard problem, as well as introduce a novel move-making optimization framework to find an approximate solution to SOL. Our qualitative and quantitative results show that our proposed method outperforms state-of-the-art algorithms for multi-part segmentation.

**Keywords:** Computer vision, image segmentation, discrete optimization, graph cuts, shape priors

## Acknowledgements

This work was partially supported by NIH grants R01-EB004640, P50-CA174521, and R01-CA167632. We thank Drs. S. O’Dorisio and Y. Menda for providing the liver data (NIH grant U01-CA140206). This work was also supported by NSERC Discovery and RTI grants (Canada) for Y. Boykov and O. Veksler.

# Contents

<b>Abstract</b>	<b>ii</b>
<b>Acknowledgements</b>	<b>iii</b>
<b>List of Figures</b>	<b>vi</b>
<b>List of Tables</b>	<b>x</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Image Segmentation . . . . .	2
1.2 Binary Segmentation . . . . .	3
1.2.1 Energy . . . . .	3
1.2.2 Optimization via Graph Cuts . . . . .	5
1.2.3 Submodularity . . . . .	8
1.3 Multi-Label Segmentation . . . . .	8
1.3.1 Energy . . . . .	8
1.3.2 Optimization via $\alpha$ -Expansion . . . . .	9
1.3.3 Optimizable Energies via $\alpha$ -Expansion . . . . .	10
1.4 Second-Order Shape Priors . . . . .	11
1.4.1 Hedgehog Shape Prior . . . . .	12
1.5 Prior Work Limitations . . . . .	14
1.6 Thesis Contributions . . . . .	15
1.7 Thesis Outline . . . . .	15
<b>2 SOL: Segmentation with Overlapping Labels</b>	<b>16</b>
2.1 Overview and Motivation . . . . .	17
2.2 SOL Energy . . . . .	20
2.3 SOL Optimization . . . . .	22
2.4 Binary Optimization Moves . . . . .	23
2.4.1 $\alpha$ -Expansion Move: . . . . .	24

Submodularity . . . . .	25
2.4.2 $\alpha$ -Contraction Move: . . . . .	26
Submodularity . . . . .	28
2.5 NP-Hardness Proof . . . . .	29
<b>3 Experiments</b>	<b>31</b>
3.1 Synthetic Example . . . . .	32
3.2 Liver Segmentation . . . . .	34
<b>4 Future Work and Conclusion</b>	<b>45</b>
<b>Bibliography</b>	<b>47</b>
<b>Curriculum Vitae</b>	<b>49</b>

# List of Figures

1.1	Classical image segmentation. (a) depicts a binary segmentation of a tiger from its background and (b) depicts multiple objects segmented from the background. . . . .	2
1.2	Example of graph-cut segmentation. From left to right: (a) base image, (b) only using the data term, (c) data and smoothness term. . . . .	4
1.3	An example of a graph construction and segmentation. (a) shows the graph nodes. (b) and (c) show the edges corresponding to the unary and pairwise potentials of energy (1.1), respectively. The fully constructed graph $G$ is shown in (d). (e) shows a possible $s/t$ -cut and the severed edges by the $s/t$ -cut are shown in grey. (f) shows the corresponding pixel labeling of the $s/t$ -cut in (e). . . . .	6
1.4	Example of a graph with a path from the source $s$ to the sink $t$ . . . . .	7
1.5	Image restoration using $\alpha$ -expansion [4]. The noisy image (b) is taken in as the input, and $\alpha$ -expansion returns an estimated image (c) as the output. . . . .	10
1.6	(a) shows an example of a shape that is star-convex w.r.t. the center $c$ and (b) is an example of a shape that is not star-convex. Note that in (b), the pixel $q$ is on the line between $c$ and $p$ but does not lie inside the shape. . . . .	11
1.7	Hedgehog constraints [18] for segment $S$ . (a) user-seed defines a signed distance map $d$ . (b) surface normals $\bar{n}_p^S$ of $S$ are constrained by $\angle(\bar{n}_p^S, \nabla d_p) \leq \theta$ . . . . .	12
1.8	shows how to approximate <i>hedgehog constraint</i> at pixel $p$ . Cone $C_{\theta(p)}$ of the allowed surface normals (blue) is enforced by ensuring that all the neighbouring pixels in the corresponding polar cone $\widehat{C}_\theta(p)$ (red) lie inside $S$ if $p \in S$ . . . . .	13
1.9	(a) synthetic example with a complex seed (red). (b) non-smooth vector field that leads to conflicting hedgehog constraints (highlighted by the red lines). Note that due to over-constraints, no segmentation can pass through the red lines on (b) leading to (c) incorrect segmentation of [18]. . . . .	14

2.1	illustrates the negative effect of using complex seeds [18, 16]. (a) the image to be segmented and the foreground seeds (shown in red), (b) distance map of the seeds, (c) the non-smooth gradient of the distance map, and (d) segmentation result of [18]. The segmentation artifacts in (d) are due to conflicting hedgehog constraints caused by the non-smooth vector field (c). . . . .	17
2.2	top row shows how we split the complex seed in Fig. 2.1 into three simple ones. Middle row shows the distance map for each seed. Bottom row shows the field of gradients for the distance maps. Notice these distance maps and vector fields are smooth. . . . .	18
2.3	(b) shows the three different local minima found by $\alpha$ -expansion with the Hedgehog constraint when enforcing mutually exclusive foreground labels. Note the over-segmentation of the first expansion and under-segmentation of subsequent expansions. (d) shows the three different local minima of our proposed method (allowing foreground labels to overlap) with the Hedgehog constraint. . . . .	19
2.4	depicts a possible segmentation with three foreground labels ( $F_1, F_2, F_3$ ) and one background label $\mathcal{B}$ , represented by $f_p^k$ for each pixel $p$ . Furthermore, the hatched area corresponds to the foreground segment and the dotted area corresponds to the background, represented by $f_p$ . . . . .	20
2.5	(a) shows a sample of a current segmentation. (b) shows a feasible expansion for label $F_1$ . Notice how $F_1$ expands while $F_2$ and $F_3$ remain unaffected. . . . .	24
2.6	(a) shows a sample of a current segmentation. (b) shows a feasible contraction for label $F_1$ . Notice how $F_1$ loses pixel support while $F_2$ and $F_3$ remain unaffected. . . . .	26
2.7	(a) the hatched area shows the pixels that may change their data term in a contraction (i.e. switch from foreground to background). Pixel $p$ is a part of this set and pixel $q$ is not. (b) shows an example of a feasible contraction. Notice that for pixel $p$ , the data term changed to background while pixel $q$ remained foreground. . . . .	27
3.1	(a) depicts our original synthetic image while (b) displays the ground-truth segmentation. . . . .	32

3.2	compares the results and energy of a synthetic example segmentation. Each row corresponds to a different segmentation model/optimization framework. The first two rows correspond to classical segmentation while the last two correspond to SOL. Columns 2-4 corresponds to different local minima by varying the order of expanding labels, if applicable. Note that the first row only has one segmentation because binary segmentation can be globally optimized. As you can see, our approach EC-moves achieves the best results. . . . .	33
3.3	(a) is the input image. (b) shows the output of the Canny-edges algorithm [6], the set of edge pixels, $E$ , is shown in white. . . . .	35
3.4	Experiment results for subject 1. (a) depicts the seeds where the teal circles are background seeds and the lines correspond to the separate foreground seeds. (b) shows the results of $\alpha$ -expansion [A1], (c) shows the results of $\alpha$ -expansion using the hedgehog shape prior [A2], and (d) shows the results of our approach [A3]. Finally, (e) is the ground truth. . . . .	36
3.5	Experiment results for subject 2. (a) depicts the seeds where the teal circles are background seeds and the lines correspond to the separate foreground seeds. (b) shows the results of $\alpha$ -expansion [A1], (c) shows the results of $\alpha$ -expansion using the hedgehog shape prior [A2], and (d) shows the results of our approach [A3]. Finally, (e) is the ground truth. . . . .	37
3.6	Experiment results for subject 3. (a) depicts the seeds where the teal circles are background seeds and the lines correspond to the separate foreground seeds. (b) shows the results of $\alpha$ -expansion [A1], (c) shows the results of $\alpha$ -expansion using the hedgehog shape prior [A2], and (d) shows the results of our approach [A3]. Finally, (e) is the ground truth. . . . .	38
3.7	Experiment results for subject 4. (a) depicts the seeds where the teal circles are background seeds and the lines correspond to the separate foreground seeds. (b) shows the results of $\alpha$ -expansion [A1], (c) shows the results of $\alpha$ -expansion using the hedgehog shape prior [A2], and (d) shows the results of our approach [A3]. Finally, (e) is the ground truth. . . . .	39
3.8	Experiment results for subject 5. (a) depicts the seeds where the teal circles are background seeds and the lines correspond to the separate foreground seeds. (b) shows the results of $\alpha$ -expansion [A1], (c) shows the results of $\alpha$ -expansion using the hedgehog shape prior [A2], and (d) shows the results of our approach [A3]. Finally, (e) is the ground truth. . . . .	40



3.9	Experiments results for subject 6. (a) depicts the seeds where the teal circles are background seeds and the lines correspond to the separate foreground seeds. (b) shows the results of $\alpha$ -expansion [A1], (c) shows the results of $\alpha$ -expansion using the hedgehog shape prior [A2], and (d) shows the results of our approach [A3]. Finally, (e) is the ground truth. . . . .	41
3.10	Graphical representation of precision vs. recall values for the three approaches (A1 - A3). . . . .	43
4.1	(a) depicts an image of cells that may benefit from SOL over mutually exclusive labels. (b) shows an expected SOL segmentation. . . . .	46

# List of Tables

3.1	Precision and Recall values for the various subjects (subjects 1 - 6) and approaches (A1 - A3). . . . .	42
3.2	F <sub>1</sub> Score for the various subjects (subjects 1 - 6) and approaches (A1 - A3).	44

# Chapter 1

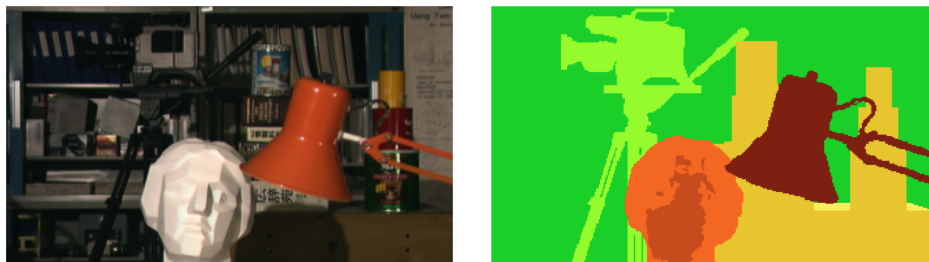
## Introduction

## 1.1 Image Segmentation

A fundamental problem in Computer Vision is image segmentation. Classical image segmentation is the process of taking in an image as an input and segmenting the image into two or more non-overlapping segments, see Fig. 1.1. Each segment/label corresponds to an object of interest in the image. The simplest form of image segmentation is binary segmentation. This involves segmenting the image into a foreground (usually representing some object) and a background (representing every other part of the image that is not the object).



(a) binary segmentation



(b) multi-label segmentation

Figure 1.1: Classical image segmentation. (a) depicts a binary segmentation of a tiger from its background and (b) depicts multiple objects segmented from the background.

There are many applications of binary segmentation. For example, segmenting body parts (e.g. kidney) in a Computed Tomography (CT) scan or a Magnetic Resonance Image (MRI) is of interest to the medical field [5]. Self-driving cars require the ability to segment road signs or obstacles from the background [20]. Image segmentation is also used extensively in commercial photo-editing tools [1].

While binary segmentation has its uses, it can only be used to segment one object of interest at a time. In practice, we often need to segment multiple objects from a single image simultaneously. For example in biomedical imaging, segmenting multiple organs is a problem with considerable clinical significance. Instead of segmenting each object individually via binary segmentation, it is possible to formulate the problem as a multi-

label segmentation problem. In multi-label segmentation the objective is to partition the image into  $n$  separate segments with unique labels. Each label corresponds to a different object of interest in the image. It is common to refer to segmentation as an image labeling problem.

There have been numerous approaches to solve the problem of image segmentation (binary or multi-label) utilizing different priors and optimization techniques. Priors refer to some prior knowledge used to guide the segmentation process. For example, by using the smoothness prior [4], segmentation with fewer discontinuities becomes preferable. There are a wide range of other priors that could be used in segmentation, e.g. shape prior [25], convexity prior [14] or structural-prior [8, 17]. In terms of optimization, it is possible to use continuous optimization methods such as convex relaxation [7] or TV-based methods [22], or discrete methods such as graph-cuts [4]. In this work we will focus on the latter approach.

We will start by discussing the simplest form of segmentation, i.e. binary segmentation, then cover multi-label segmentation and star shape priors [25, 18]. Afterwards, we will highlight some of the drawbacks of the classical segmentation model which assumes that segments (or labels) are mutually exclusive.

## 1.2 Binary Segmentation

As previously mentioned, the binary segmentation problem involves segmenting an image into a foreground and background. This can be solved by minimizing an energy.

### 1.2.1 Energy

The energy that is composed of (a) a term to account for the information (e.g. color/intensity) in the image (data term), and (b) a term to discourage discontinuities (smoothness term). Before formulating the binary segmentation problem as an energy function, we will first introduce some notation. Let  $\Omega$  denote the set of image pixels,  $\mathcal{N}$  denote some pixel neighbourhood<sup>1</sup>, and the set of labels  $\mathcal{L} = \{0, 1\}$  where 0 is the background and 1 is the foreground. Let  $f_p \in \mathcal{L}$  be the label variable of pixel  $p \in \Omega$  such that  $\mathbf{f} = \{f_p \mid \forall p \in \Omega\}$  is an image labeling. Given these variables, the binary segmentation problem [5, 4] can

---

<sup>1</sup>Two commonly used neighbourhoods in computer vision for 2D images are the 4-neighbourhood and 8-neighbourhood where a pixel is considered in the neighbourhood of its 4 (or 8) nearest pixels

be formulated as follows

$$E(\mathbf{f}) = \underbrace{\sum_{p \in \Omega} D_p^\gamma(f_p)}_{\text{Data Term}} + \lambda \underbrace{\sum_{p, q \in \mathcal{N}} V(f_p, f_q)}_{\text{Smoothness Term}} \quad (1.1)$$

where  $\lambda$  is a normalization term, and the “unary” data  $D$  and “pairwise” smoothness  $V$  functions will be explained shortly.

**Data Term :**  $D_p^\gamma(f_p)$  is a function that measures how well pixel  $p$  fits label  $f_p$ . One of the most commonly used data terms is the negative log likelihood of the probability of pixels’ features given a label’s feature model

$$D_p^\gamma(f_p) = -\ln \Pr(I_p | \gamma_{f_p}) \quad (1.2)$$

where  $I_p$  represents the information at pixel  $p$  and  $\{\gamma_i | i \in \mathcal{L}\}$  represents the parameter of the probability distribution  $\Pr(x | \gamma_i)$  for pixel features in different segments, e.g. intensity, colour, etc. Sometimes the parameter  $\gamma$  is already known and can simply be passed into the data term. However, many problems require the parameter to first be estimated. A common example of this is the *gaussian mixture model* (GMM) of the feature information (e.g. colour gaussian mixture model). These distributions may be estimated using the standard *expectation-maximization* (EM) procedure [2]. The input to calculate the models are often the initial seeds/scribbles provided by the user.

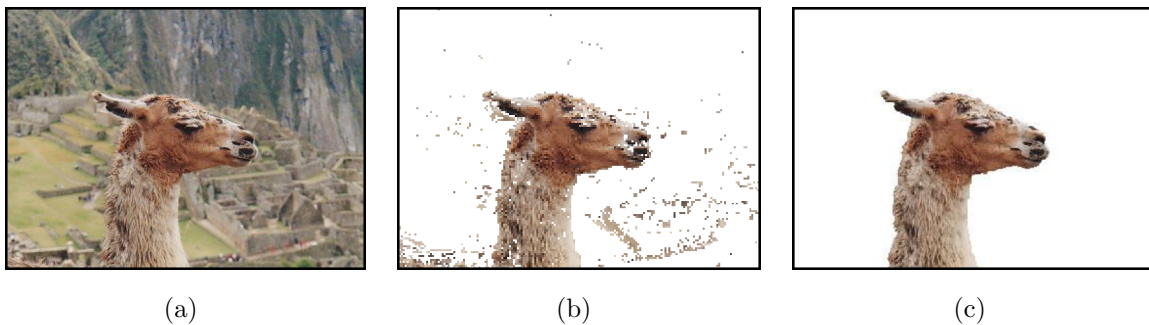


Figure 1.2: Example of graph-cut segmentation. From left to right: (a) base image, (b) only using the data term, (c) data and smoothness term.

While the data term encodes the information as to whether or not a particular pixel is more likely to be foreground or background, it is not sufficient to acquire an accurate segmentation. This is demonstrated in Fig. 1.2(b), where only the first “unary”-term on (1.1) is minimized by the simple log likelihood test

$$f_p = \begin{cases} 1 & \text{if } -\ln \frac{\Pr(I_p | \gamma_1)}{\Pr(I_p | \gamma_0)} > 0 \\ 0 & \text{otherwise.} \end{cases}$$

This example shows the effect of having background pixels similar to the foreground pixels and how it results in a non-smooth segmentation. The smoothness term in the energy (1.1) is used to encourage spatially coherent segmentation. The improvement in the segmentation when using the smoothness term can be seen in Fig. 1.2(c).

**Smoothness Term :** The smoothness function  $V(f_p, f_q)$  in equation (1.1) is used to penalize discontinuities in the segmentation. A common smoothness function is

$$V(f_p, f_q) = \begin{cases} 1 & f_p \neq f_q \\ 0 & f_p = f_q \end{cases} \quad (1.3)$$

which is known as the ‘‘Ising model’’ [4] for binary problems and the ‘‘Potts model’’ for multi-label problems. This terminology comes from Markov Random Fields and statistical physics [23]. The smoothness parameter  $\lambda$  is a measure of how important the smoothness penalty is w.r.t. the data term. If  $\lambda$  is set to  $\infty$  then all pixels will be assigned to the same label. If  $\lambda$  is set to 0 the segmentation will be entirely dependent on the data term.

### 1.2.2 Optimization via Graph Cuts

Energy (1.1) can be globally optimized in polynomial time via graph cuts [4]. This can be done by constructing a graph with two special nodes, i.e. source  $s$  and sink  $t$ , and encoding the unary and pairwise potentials of (1.1) as edge weights. In the constructed graph,  $s$  and  $t$  represent the foreground and background labels, respectively. The optimal segmentation can be computed as a partition of the graph into two disjoint subsets<sup>2</sup> that minimizes the sum of edge weights between them. This partitioning problem is commonly referred to as the min-cut problem. Now we will briefly cover the graph construction and how to find its min-cut.

#### Graph Construction:

*Unary Potentials (t-links):* To construct the graph we begin by adding a node for each image pixel, see Fig. 1.3(a). Each node is then connected to both  $s$  and  $t$  via t-links as illustrated in Fig. 1.3(b). The t-link  $(s, p)$  encodes the penalty for assigning  $p$  to the

---

<sup>2</sup>One contains the source while the other contains the sink.

background, while t-link  $(p, t)$  encodes the penalty for assigning  $p$  to the foreground. The min-cut will sever one of the two t-links attached to  $p$  which defines whether  $p$  is connected to the  $s$  or  $t$  nodes, i.e.  $p$  is a foreground or background pixel. For example, if the t-link from the  $s$  is severed, the node will only be connected to  $t$  and its final labeling will be background.

*Pairwise Potentials (n-links):* To encode the discontinuity penalties, for each pair of neighbouring pixels/nodes  $p$  and  $q$  in  $\mathcal{N}$  we will add an undirected edge, Fig. 1.3(c). Those undirected edges are commonly referred to as *n-links*. The weight associated with each *n-link* is  $\lambda$ .

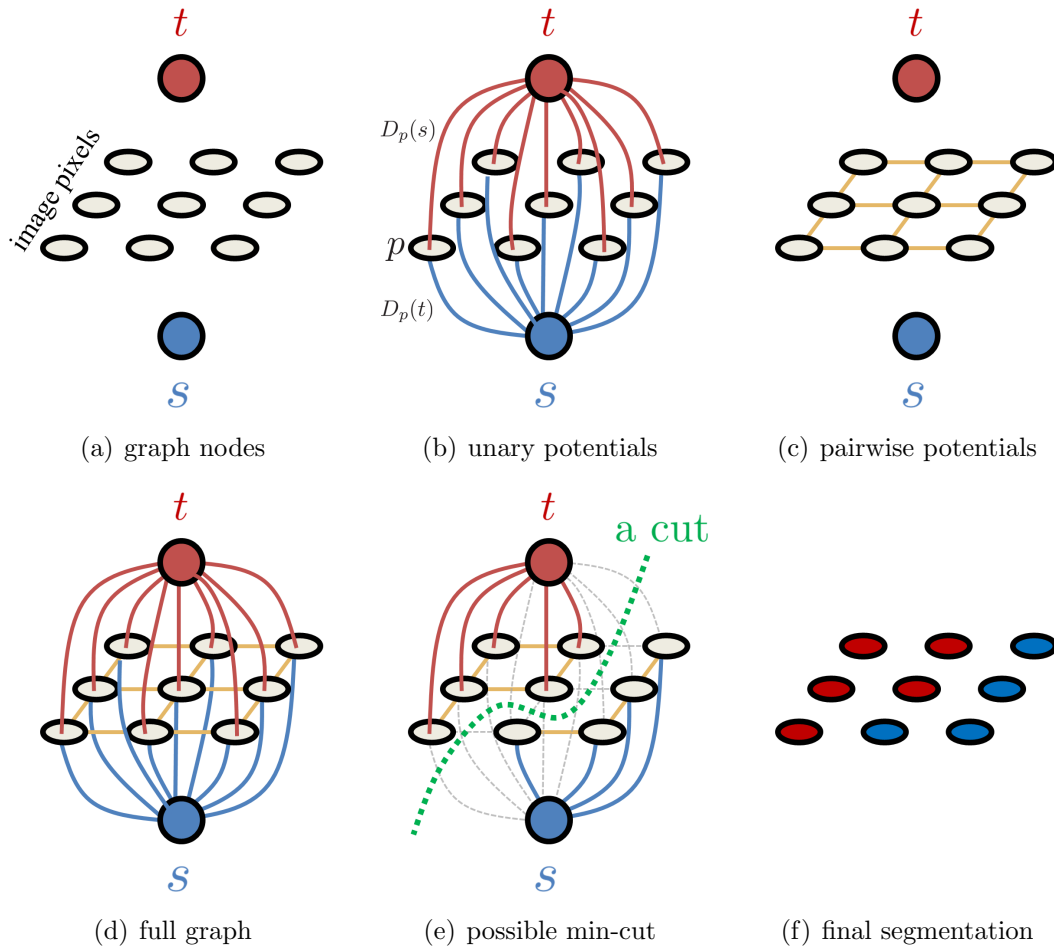


Figure 1.3: An example of a graph construction and segmentation. (a) shows the graph nodes. (b) and (c) show the edges corresponding to the unary and pairwise potentials of energy (1.1), respectively. The fully constructed graph  $G$  is shown in (d). (e) shows a possible  $s/t$ -cut and the severed edges by the  $s/t$ -cut are shown in grey. (f) shows the corresponding pixel labeling of the  $s/t$ -cut in (e).



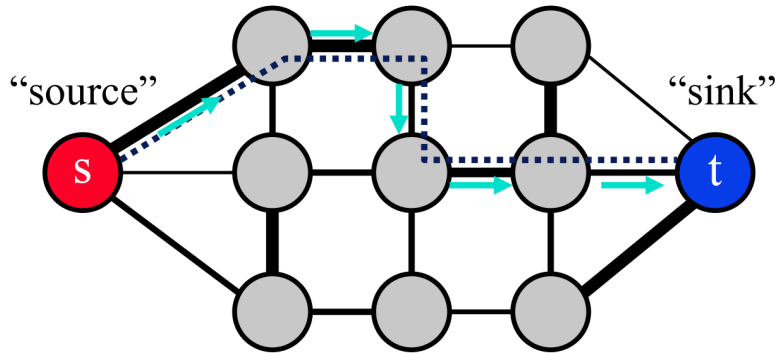


Figure 1.4: Example of a graph with a path from the source  $s$  to the sink  $t$ .

### Min-Cut:

Finding the total weight of the edges in a minimum cut is equivalent to finding the maximum flow in a flow network [10]. Therefore, any maximum flow algorithm can be used to help us solve our binary graph cut problem. In this Section we will explore various maximum flow algorithms.

One of the first maximum flow algorithms developed is *Ford-Fulkerson* [11]. It is a greedy algorithm that finds and saturates paths (i.e. no more flow can be pushed through the path) from  $s$  to  $t$ , see Fig. 1.4. The algorithm terminates once all paths are saturated. A downside of *Ford-Fulkerson* is its time complexity of  $O(mf^*)$ , where  $m$  is the number of edges in the graph, and  $f^*$  is the maximum flow when capacities are integers. Notice that the time complexity depends on the maximum flow value which could be arbitrarily large depending on the edge weights. Thus, a max-flow algorithm that depends only on the structure of the graph is more preferable than *Ford-Fulkerson*.

There have been a number of successful attempts to develop algorithms with better time bounds such as *Push-Relabel* [13], with a time complexity of  $O(n^2m)$  where  $n$  is the number of vertices and  $m$  is the number of edges. While there are standard choices for general problems with good theoretical bounds, in practice, it has been found that different algorithms perform better for specific applications. In Computer Vision, the *Boykov-Kolmogorov* (BK) algorithm [3] has been shown to outperform general-purpose methods. Despite the lack of a polynomial time bound, BK has been extensively used by the vision community. Recently, an extension of BK was developed that has polynomial time, named *Incremental Breadth First Search* (IBFS), which has been shown to be faster than BK in general [12].

Not every pairwise energy (e.g. smoothness term) can be globally optimized via graph

cuts. As a matter of fact, only a subset of pairwise energies can be represented as a graph for which we can compute the max-flow [21, 4]. This subset of energies is commonly referred to as submodular energies. We cover this subset in more detail in Section 1.2.3.

### 1.2.3 Submodularity

A pairwise energy function is considered *submodular* if it satisfies the submodularity condition

$$V(0, 0) + V(1, 1) \leq V(1, 0) + V(0, 1) \quad \forall (p, q) \in \mathcal{N}. \quad (1.4)$$

The submodularity condition is of significance because submodular pairwise energies can be globally optimized in polynomial time via graph cuts [21]. Moreover, it has been shown in [21] that optimizing non-submodular energies is an intractable problem for general graphs (e.g. on trees it is solvable via dynamic programming).

To prove that energy (1.1) is submodular, we need to show that condition (1.4) is satisfied. By substituting (1.3) into (1.4) we acquire the following inequality

$$0 + 0 \leq 1 + 1$$

which holds for all  $(p, q)$  in  $\mathcal{N}$ . It is well known that energy (1.1) is submodular for the Ising model (1.3).

## 1.3 Multi-Label Segmentation

As previously discussed, binary segmentation is limited by its ability to only segment a single object in an image. On the other hand, multi-label segmentation explores the concept of segmenting an image into more than two segments. More precisely, the label set for multi-label segmentation is  $\mathcal{L} = \{l_1, l_2, \dots, l_n\}$ , where  $n > 2$ .

### 1.3.1 Energy

Like binary segmentation, multi-label segmentation also involves minimizing an energy with a data term and a smoothness term. Unlike binary segmentation minimizing the energy for multi-label segmentation, even with Potts model (1.3), is NP-hard [4]. The energy for multi-label segmentation is

$$E(\mathbf{f}) = \underbrace{\sum_{p \in \Omega} D_p(f_p)}_{\text{Data Term}} + \lambda \underbrace{\sum_{p, q \in \mathcal{N}} V(f_p, f_q)}_{\text{Smoothness Term}} \quad (1.5)$$

where  $\mathbf{f} = \{f_p \in \mathcal{L} \mid \forall p \in \Omega\}$ .

The data term is similar to the one defined for binary segmentation (1.2). In terms of pairwise potentials, the greater number of labels corresponds to the possibility of comparing labels in an unordered or ordered fashion. That is, comparing ordered labels may have different costs depending on where the labels lie in the order. Potts model (1.3) is an example of an unordered label comparison. In Potts model the cost of assigning a pair of neighbouring pixels to *any* two different labels is the same. Potts model could be seen as the generalization of the Ising model to multi-labeling. In the case of ordered labels, e.g. for 1D labels  $\mathcal{L} \subset \mathbb{R}^1$ , there are pairwise potentials  $V(x, y) = g(x - y)$  such that energy (1.1) can be optimized for convex [19] functions  $g : \mathbb{R}^1 \rightarrow \mathbb{R}^1$ . For non-convex  $g$ , e.g. Potts model  $g(t) = [t \neq 0]$  [4] (where  $[ ]$  are the Iverson brackets) or truncated quadratic  $g(t) = \min(t^2, T)$  [26] (where  $T$  is some threshold), optimization of (1.1) becomes NP-hard on the multi-label cost.

As the multi-label segmentation problem is NP-hard, there are a number of algorithms that attempt to find an approximate solution. One such example is  $\alpha$ -expansion [4] which is covered in Section 1.3.2 below.

### 1.3.2 Optimization via $\alpha$ -Expansion

One of the most commonly used algorithms to find an approximate solution to multi-label energies in computer vision is  $\alpha$ -expansion [4], Alg. 1.

---

**Algorithm 1:**  $\alpha$ -expansion

---

```

1  $\hat{\mathbf{f}} \leftarrow$  arbitrary labeling
2 repeat
3   for any (randomly) chosen  $\alpha \in \mathcal{L}$ 
4      $\mathbf{f}^\alpha \leftarrow \arg \min_{\mathbf{f}} E(\mathbf{f})$  where  $\mathbf{f}$  is an  $\alpha$ -expansion of  $\hat{\mathbf{f}}$ 
5     if  $E(\mathbf{f}^\alpha) < E(\hat{\mathbf{f}})$ 
6        $\hat{\mathbf{f}} \leftarrow \mathbf{f}^\alpha$ 
7 until no expansion move reduces  $E$ 

```

---

The  $\alpha$ -expansion algorithm begins with an arbitrary labeling  $\hat{\mathbf{f}}$ . At each iteration, a label  $\alpha$  is chosen at random. On line 4, label  $\alpha$  is given the opportunity to expand its support region, i.e. every pixel is given the binary choice to either keep its current labeling or switch to  $\alpha$ . The expansion move on line 4 can be formulated as  $f_p = \alpha \cdot x_p + \hat{f}_p \cdot (1 - x_p)$  using the binary labeling  $\mathbf{x} = \{x_p \in \{0, 1\} \mid p \in \Omega\}$  that can be solved optimally for “metric” interaction potentials [4], see Section 1.3.3. Once the labeling  $\hat{\mathbf{f}}$  can no longer

be improved upon, the algorithm has converged.

There are a number of applications for multi-label segmentation. One such application is image restoration. The problem of image restoration consists of taking a noisy image as input and estimating the original noise-free image. Noise in an image is possible due to many factors, e.g. low light. By using all image intensities as labels, it is possible to “restore” the image using  $\alpha$ -expansion as illustrated in Fig. 1.5.

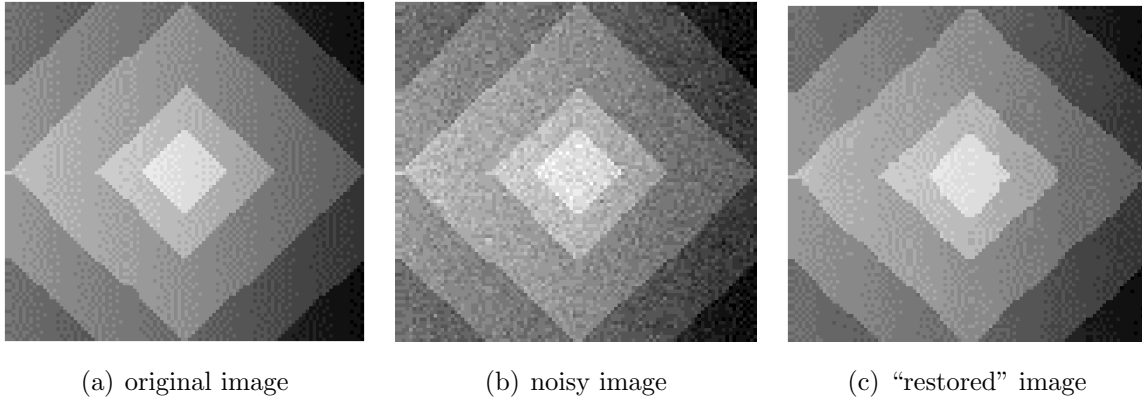


Figure 1.5: Image restoration using  $\alpha$ -expansion [4]. The noisy image (b) is taken in as the input, and  $\alpha$ -expansion returns an estimated image (c) as the output.

Similar to binary segmentation, not every multi-label pairwise energy can be optimized via graph cuts. In Section 1.3.3, we present the conditions under which a multi-label energy can be optimized via  $\alpha$ -expansion. In Section 1.4, we present some shape priors used in addition to the smoothness prior to improve segmentation accuracy.

### 1.3.3 Optimizable Energies via $\alpha$ -Expansion

In Section 1.2.3, the concept of submodularity was covered as an indicator as to whether a binary pairwise energy function could be optimized in polynomial time or not. With multi-label energy, we are exposed to more general smoothness functions (e.g. convex, truncated convex, etc.). As shown in [4], for such functions to be optimized via  $\alpha$ -expansion, the pairwise function must be metric. A function  $V(f_p, f_q)$  is metric on the set of labels  $\mathcal{L}$  if it satisfies three constraints:

$$V(\alpha, \beta) = 0 \Leftrightarrow \alpha = \beta \quad \forall \alpha, \beta \in \mathcal{L} \quad (1.6)$$

$$V(\alpha, \beta) = V(\beta, \alpha) \geq 0 \quad \forall \alpha, \beta \in \mathcal{L} \quad (1.7)$$

$$V(\alpha, \beta) \leq V(\alpha, \gamma) + V(\gamma, \beta) \quad \forall \alpha, \beta, \gamma \in \mathcal{L}. \quad (1.8)$$

These constraints guarantee that the binary expansion moves are submodular and aids in determining if utilizing a particular algorithm is feasible or not. For example, the  $\alpha$ -expansion algorithm is only viable for pairwise functions that are metric. Other algorithms, such as the  $\alpha$ - $\beta$ -swap algorithm [4], are more general and could be used for semi-metric energy functions. A semi-metric function is a function that only satisfies (1.6) and (1.7).

There are various shape priors that can be represented as metric pairwise potentials. Examples include star [25] and hedgehog [18] shape priors that will be examined in Section 1.4 below.

## 1.4 Second-Order Shape Priors

We have covered energy functions that use the image data and smoothness prior. However, users may have further knowledge about the shape of the object to be segmented. It is possible to incorporate that information into the energy function for more accurate results. A common shape prior is the star-shape prior [25] which allows the user to provide information about the shape of the object of interest via a single click identifying its center  $c$ . The star-shape prior guarantees that the segmentation will result in a star-convex shape w.r.t. the center  $c$ . A star-convex shape w.r.t.  $c$  means that for any pixel  $p$  in the shape, all the pixels along the line between  $c$  and  $p$  must also belong to the shape, see illustration in Fig. 1.6.

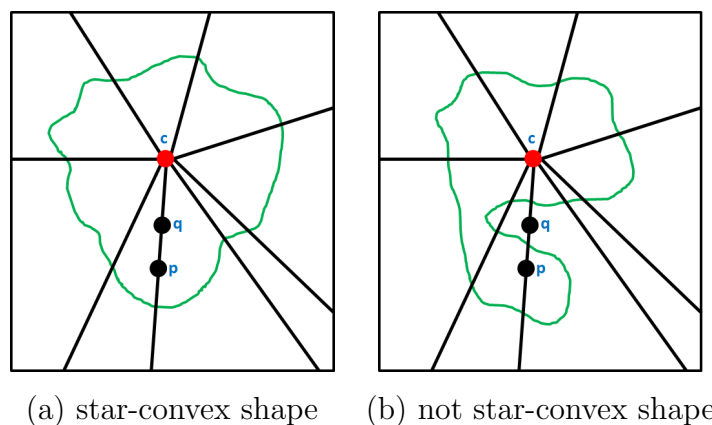
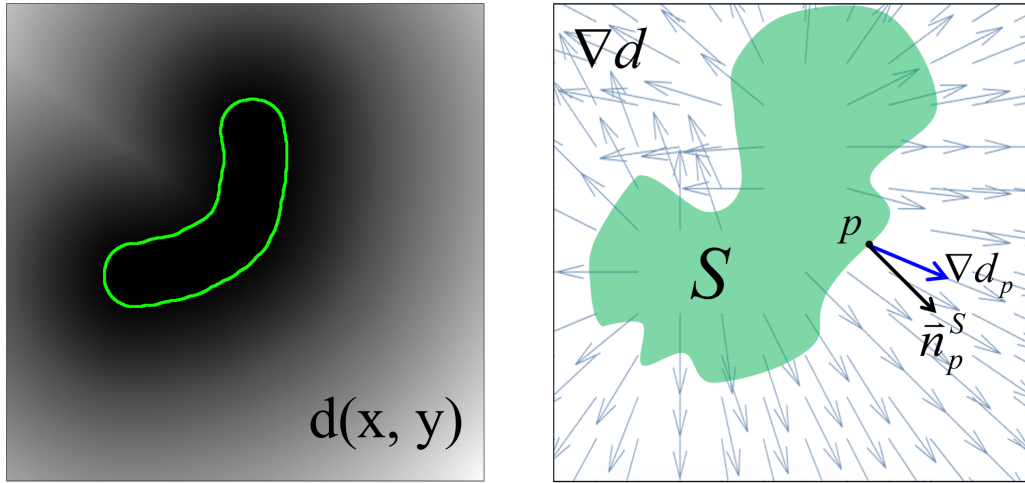


Figure 1.6: (a) shows an example of a shape that is star-convex w.r.t. the center  $c$  and (b) is an example of a shape that is not star-convex. Note that in (b), the pixel  $q$  is on the line between  $c$  and  $p$  but does not lie inside the shape.

Not all objects can be represented using the star-shape prior. Alternatively, the hedgehog shape prior [18] is capable of representing a larger set of shapes. We will give a brief overview of the hedgehog shape prior in Section 1.4.1, since our work extends it.

### 1.4.1 Hedgehog Shape Prior

The hedgehog shape prior locally constrains normals of points on the surface of the segment using a reference vector field. As explored in [18] there is more than one way to generate the reference vector field. For example in [18], the reference vector field was the gradient of the distance map of the user-scribble/seed. Unlike the star-shape prior, a hedgehog seed is not restricted to being a single point. In contrast to the star-shape prior, the hedgehog prior utilizes a shape tightness parameter  $\theta$  that gives the user control over the set of feasible shapes.



(a) user-seed & its distance map  $d$  (b) Hedgehog constraint  $\angle(\bar{n}_p^S, \nabla d_p) \leq \theta$

Figure 1.7: Hedgehog constraints [18] for segment  $S$ . (a) user-seed defines a signed distance map  $d$ . (b) surface normals  $\bar{n}_p^S$  of  $S$  are constrained by  $\angle(\bar{n}_p^S, \nabla d_p) \leq \theta$ .

Figure 1.7 illustrates the hedgehog constraint:

$$\angle(\bar{n}_p^S, \nabla d_p) \leq \theta \quad \forall p \in \partial S, \quad (1.9)$$

where  $\bar{n}_p^S$  is the normal of the surface of segment  $S$  at  $p$ ,  $\nabla d_p$  is the reference vector field at  $p$  and  $\partial S$  is the boundary of  $S$ . Now we will give a brief overview on how the hedgehog constraints (1.9) were formulated as pairwise potentials in [18].

Let us consider the area in which the surface normal may fall as the cone of allowed surface normals  $C_\theta(p)$ ,

$$C_\theta(p): \quad \angle(\bar{n}_p^S, \nabla d_p) \leq \theta \quad \forall p \in \partial S \quad (1.10)$$

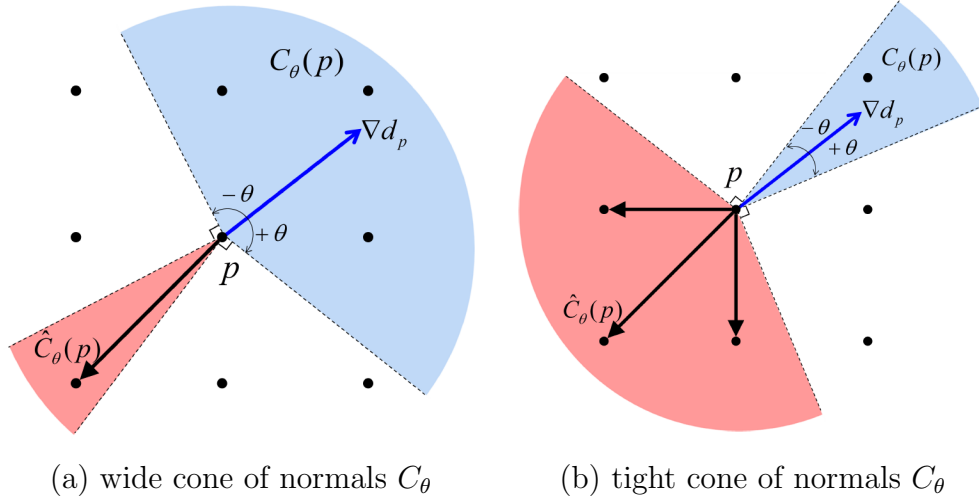
(a) wide cone of normals  $C_\theta$ (b) tight cone of normals  $C_\theta$ 

Figure 1.8: shows how to approximate *hedgehog constraint* at pixel  $p$ . Cone  $C_{\theta(p)}$  of the allowed surface normals (blue) is enforced by ensuring that all the neighbouring pixels in the corresponding polar cone  $\widehat{C}_\theta(p)$  (red) lie inside  $S$  if  $p \in S$ .

at some pixel  $p$ , see the illustration in Fig. 1.8 for two different values of  $\theta$ . It is easy to see that the boundary of segment  $S$  at  $p$  has normal  $\bar{n}_p^S \in C_\theta(p)$  iff the surface of  $S$  does not pass through the corresponding polar cone  $\widehat{C}_\theta(p)$ ,

$$\widehat{C}_\theta(p) := \{y \mid \langle (p, y), (p, z) \rangle \leq 0 \quad \forall z \in C_\theta(p)\}. \quad (1.11)$$

With this in mind, it is easy to approximate the hedgehog constraint (1.9) as pairwise potentials. Simply, the hedgehog constraint at  $p$  boils down to the following: if  $p$  belongs to segment  $S$  then all neighbouring points of  $p$  that lie in  $\widehat{C}_\theta(p)$  must also lie inside  $S$ . Let  $\mathcal{E}(\theta)$  be the set of hedgehog constraints edges,

$$\mathcal{E}(\theta) = \{(p, q) \mid (p, q) \in \widehat{C}_\theta(p) \quad \forall (p, q) \in \mathcal{N}\}. \quad (1.12)$$

The shape tightness parameter  $\theta$  is used to control how close the final segmentation is to the level sets of the distance map of the seed. When  $\theta$  is 0, the set of feasible shapes will be the levels sets of the distance map  $d$ . As  $\theta$  increases, the set of feasible shapes increases, see [18] for more details.

The hedgehog shape prior can be added to (1.5) as follows:

$$E(\mathbf{f}) = \underbrace{\sum_{p \in \mathcal{P}} D_p(f_p)}_{\text{Data Term}} + \lambda \underbrace{\sum_{p, q \in \mathcal{N}} V(f_p, f_q)}_{\text{Smoothness Term}} + \underbrace{H_\theta(\mathbf{f})}_{\text{Shape Prior Term}} \quad (1.13)$$

where

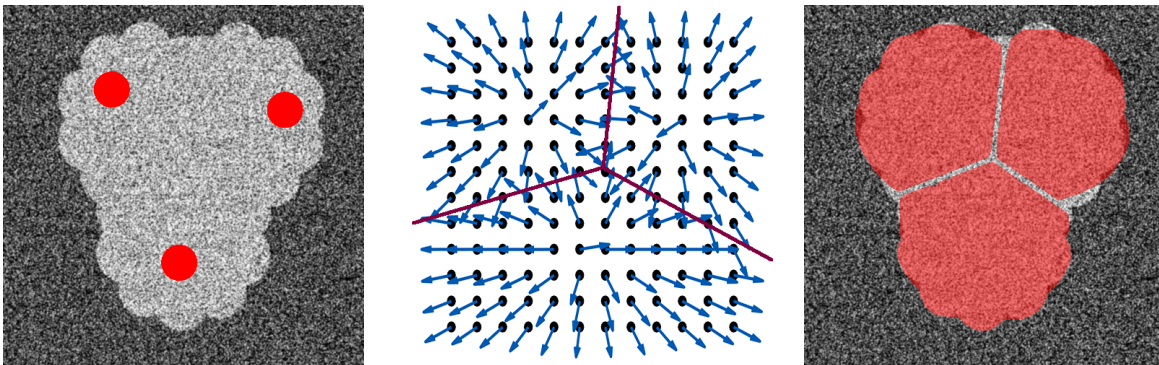
$$H_\theta(\mathbf{f}) = \sum_{k \in \mathcal{L}} \sum_{(p, q) \in \mathcal{E}_k(\theta)} w_\infty [f_p = k, f_q \neq k] \quad (1.14)$$

and  $[ \ ]$  are the Iverson brackets,  $w_\infty$  is a prohibitively expensive weight, and  $\mathcal{E}_k$  corresponds to the hedgehog shape prior for the boundary of label (object)  $k$ . If the proposition inside the Iverson bracket is true it returns 1 and 0 otherwise.

## 1.5 Prior Work Limitations

In previous sections, we explored the concepts of binary and multi-label segmentation. We now look at the drawbacks of the corresponding optimization algorithms in light of the hedgehog prior. For simplicity, we will focus on binary segmentation but the findings extend to multi-label segmentation as well.

In the context of binary segmentation with hedgehogs, a problem arises when the user seeds/scribbles are complex, e.g. a seed with multiple disconnected parts, see Fig. 1.9(a). As reported in [16], such complex seeds usually results in a non-smooth vector field which leads to segmentation errors, see Fig. 1.9(c), due to conflicting hedgehog constraints along the vector field discontinuities, see Fig. 1.9(b).



(a) original image with seeds (b) conflicting vector field (c) segmented image

Figure 1.9: (a) synthetic example with a complex seed (red). (b) non-smooth vector field that leads to conflicting hedgehog constraints (highlighted by the red lines). Note that due to over-constraints, no segmentation can pass through the red lines on (b) leading to (c) incorrect segmentation of [18].

To overcome the issue of conflicting constraints, we propose using *multiple smooth* vector fields to constrain the shape of the foreground segment, as presented in Section 2.1, instead of using a *single non-smooth* vector field [18].



## 1.6 Thesis Contributions

The main contributions of this thesis are:

- A new segmentation model that allows overlapping labels, namely **Segmentation with Overlapping Labels (SOL)**. This model is well-suited for segmenting multi-part foreground objects where each part has an independent shape prior. We show that solving image segmentation while allowing labels to overlap is an NP-hard problem.
- To approximately optimize SOL, we propose an optimization framework that uses our novel combinatorial moves, **Expansion-Contraction moves (EC-moves)**.
- We evaluate our approach on synthetic examples and real data, i.e. liver segmentation in CT-scans. We also compare our results to state of the art algorithms using Potts model ( $\alpha$ -expansion with and without the Hedgehog shape prior).

## 1.7 Thesis Outline

In **Chapter 2** we introduce SOL and its optimization framework that utilizes our novel EC-moves. In addition, we prove that SOL is NP-Hard. In **Chapter 3** we evaluate our approach on synthetic and real data. We qualitatively and quantitatively compare our approach to state of the art algorithms [18, 4]. **Chapter 4** concludes the thesis and offers possible future research directions.

## Chapter 2

# SOL: Segmentation with Overlapping Labels

## 2.1 Overview and Motivation

In Section 1.5, we exposed some issues with the classical segmentation model where segments are mutually exclusive. To summarize, using complex seeds (e.g. disjoint seeds) for a hedgehog shape prior [18] often results in a segmentation that does not accurately cover the object [18, 16] as illustrated in Fig. 2.1. More precisely, for complex seeds, e.g. multi-part seed shown in Fig. 2.1(a), the resulting distance map is not continuous as shown in Fig. 2.1(b). Thus, the gradient of the distance map is a non-smooth vector field, see Fig. 2.1(c). Using a non-smooth vector field to impose the hedgehog constraints (1.9) results in conflicting constraints along the distance map discontinuities, which in turn results in an incorrect segmentation, see Fig. 2.1(d).

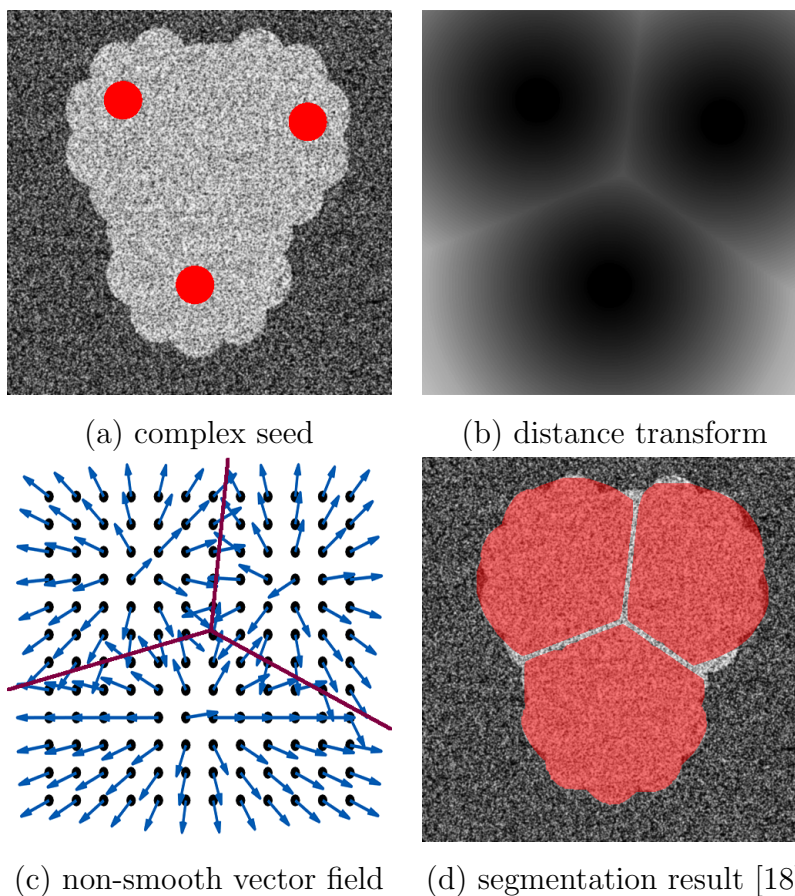


Figure 2.1: illustrates the negative effect of using complex seeds [18, 16]. (a) the image to be segmented and the foreground seeds (shown in red), (b) distance map of the seeds, (c) the non-smooth gradient of the distance map, and (d) segmentation result of [18]. The segmentation artifacts in (d) are due to conflicting hedgehog constraints caused by the non-smooth vector field (c).

In an attempt to avoid discontinuities in the distance map, we explored the idea of splitting complex seeds into multiple simple ones with different labels resulting in continuous distance maps. Instead of segmenting one foreground label using complex seeds, we propose separately segmenting  $n$  foreground parts/labels, each with a simple seed. For instance, the complex seed shown in Fig.2.1(a) could be split into three parts/labels, as shown in Fig. 2.2, top row. As you can see the resulting independent vector fields are smooth and do not lead to conflicting hedgehog constraints.

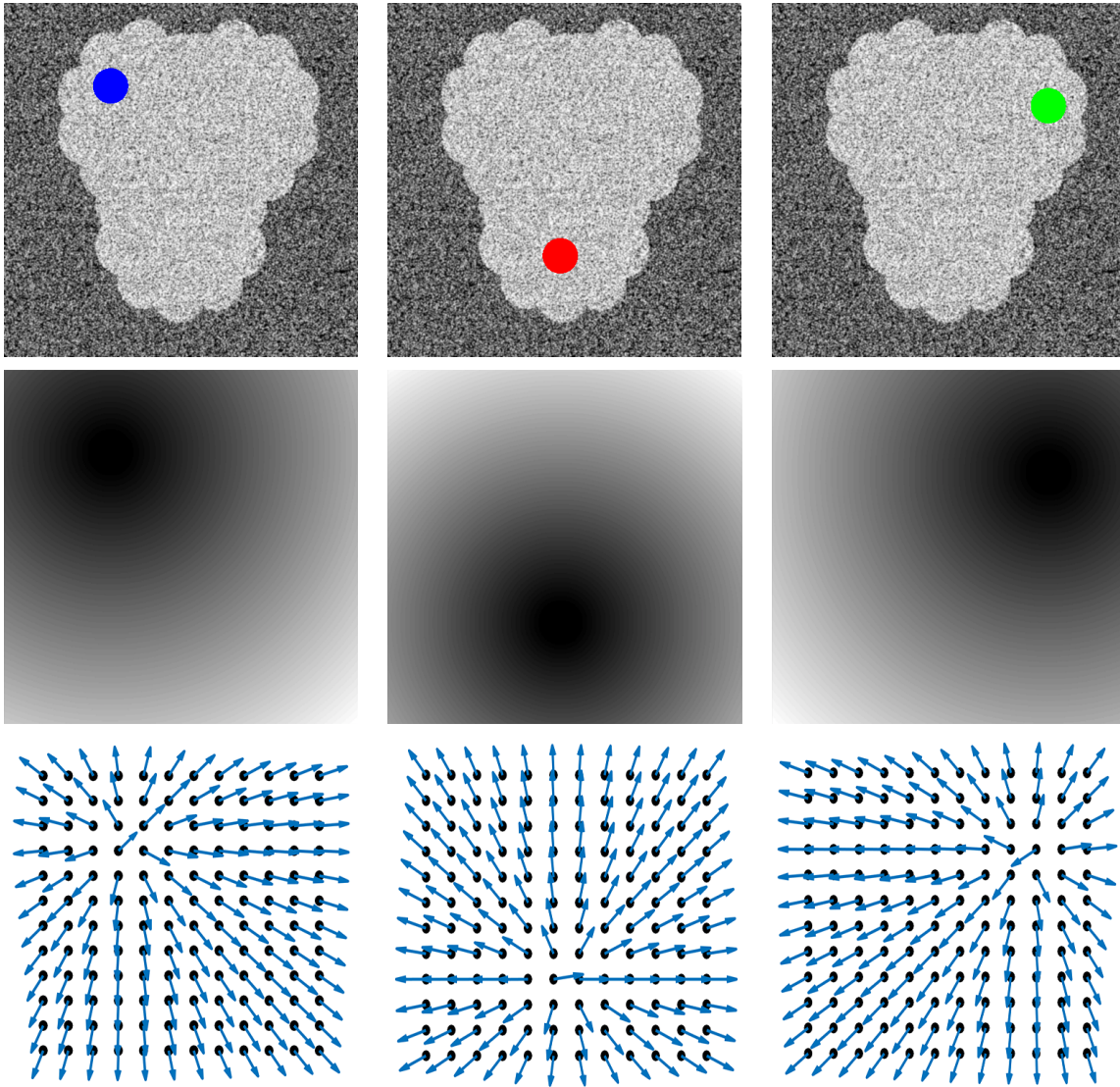


Figure 2.2: top row shows how we split the complex seed in Fig. 2.1 into three simple ones. Middle row shows the distance map for each seed. Bottom row shows the field of gradients for the distance maps. Notice these distance maps and vector fields are smooth.

On the one hand, splitting complex seeds into multiple simple ones eliminates the issue of conflicting hedgehog constraints. On the other hand, the globally optimizable binary segmentation problem transforms into an NP-hard multi-labeling problem, see Section 1.3. It is possible to use  $\alpha$ -expansion to obtain an approximate solution. However, in practice we found that  $\alpha$ -expansion usually converges to poor local minima, see Figure 2.3(b). We mainly attribute this poor performance to using the classical segmentation model that assumes independent mutually exclusive labels. In the context of our application, the independence assumption does not hold. This is due to the fact that our foreground labels/parts represent the same foreground object but each label has its own hedgehog constraints.

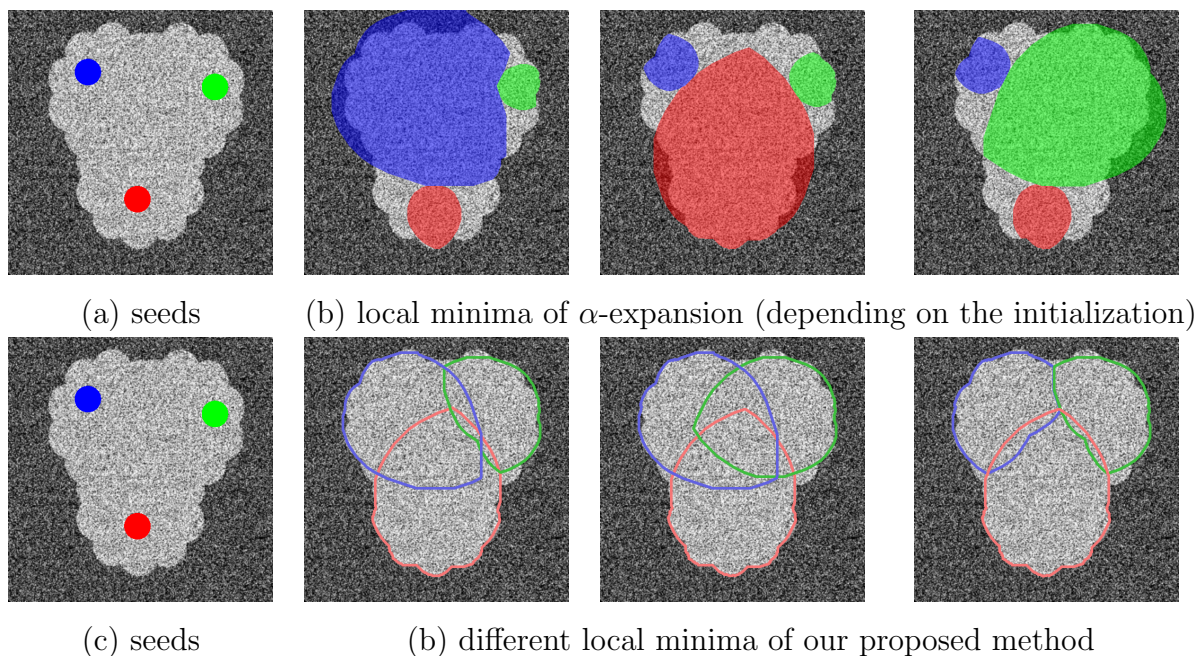


Figure 2.3: (b) shows the three different local minima found by  $\alpha$ -expansion with the Hedgehog constraint when enforcing mutually exclusive foreground labels. Note the over-segmentation of the first expansion and under-segmentation of subsequent expansions. (d) shows the three different local minima of our proposed method (allowing foreground labels to overlap) with the Hedgehog constraint.

In this work we propose a novel segmentation model which allows labels to overlap. We will refer to this model as **Segmentation with Overlapping Labels (SOL)**. SOL allows for a single object to be segmented using multiple labels representing different non-mutually-exclusive parts. Intuitively, such labels/parts are no longer independent. We will provide an energy formulation of the SOL problem and show that it is NP-Hard. We will also introduce an optimization framework for finding an approximate solution to SOL.

## 2.2 SOL Energy

We will start by redefining some notation. Let  $\Omega$  be the set of all image pixels, and  $\mathcal{L} = \{\mathcal{B}, F_1, \dots, F_n\}$  be set of all labels where  $\mathcal{B}$  and  $F_i$  are the background and the  $i^{\text{th}}$  foreground part, respectively. Let  $\gamma_{\mathcal{B}}$  and  $\gamma_F$  denote the Gaussian mixture color model of the background and foreground labels, respectively. In the context of our application, all foreground parts have the same color model but they could be different if needed.

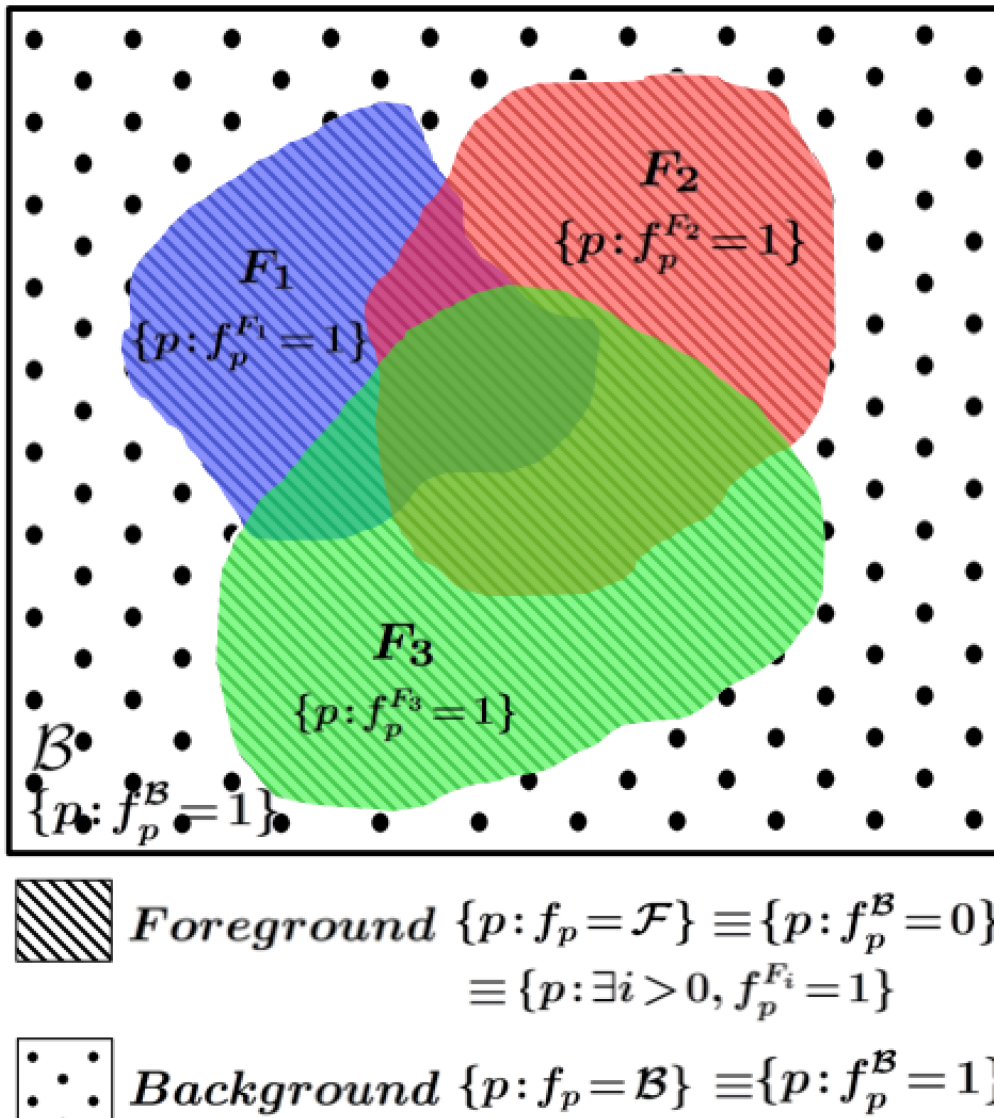


Figure 2.4: depicts a possible segmentation with three foreground labels ( $F_1, F_2, F_3$ ) and one background label  $\mathcal{B}$ , represented by  $f_p^k$  for each pixel  $p$ . Furthermore, the hatched area corresponds to the foreground segment and the dotted area corresponds to the background, represented by  $f_p$ .

Let  $\mathbf{f} = \{f_p^k \mid \forall p \in \Omega, k \in \mathcal{L}\}$  be a labeling of  $\Omega$  where  $f_p^k$  is a binary variable that indicates whether pixel  $p$  is assigned to label  $k$  or not, i.e.

$$f_p^k = \begin{cases} 1 & \text{pixel } p \text{ is assigned to } k \in \mathcal{L} \\ 0 & \text{otherwise.} \end{cases}$$

By definition,  $\mathbf{f}$  allows a pixel to be assigned to more than one label. However, in the context of our application we only allow a pixel to be assigned to either the background or at least one of the foreground labels, i.e.

$$f_p^{\mathcal{B}} = \prod_{i=1}^n (1 - f_p^{F_i}) \quad \forall p \in \Omega.$$

We will further introduce an indicator variable  $f_p \in \{\mathcal{B}, \mathcal{F}\}$  to indicate whether  $p$  is assigned to the background label or not, refer to Figure 2.4. More specifically, if a pixel is assigned to any number of foreground labels,  $f_p = \mathcal{F}$ , and if a pixel is assigned to the background label,  $f_p = \mathcal{B}$ . Technically  $f_p$  is equivalent to  $f_p^{\mathcal{B}}$  as  $f_p^{\mathcal{B}} = 0$  means that  $f_p = \mathcal{B}$ . However, we use the symbols  $\mathcal{B}$  and  $\mathcal{F}$  instead of 0 and 1 for notational clarity.

Our multi-part foreground segmentation energy of SOL is

$$E(\mathbf{f}, \gamma_{\mathcal{B}}, \gamma_{\mathcal{F}}) = \underbrace{\sum_{p \in \Omega} D_p^\gamma(f_p)}_{\text{data term}} + \lambda \underbrace{\sum_{p, q \in \mathcal{N}} w_{pq} [f_p \neq f_q]}_{\text{smoothness term}} + \underbrace{H_\theta(\mathbf{f})}_{\text{hedgehog constraints}}, \quad (2.1)$$

where  $\lambda$  is a normalization constant,  $\mathcal{N}$  is the pixels' neighbour system,  $w_{pq}$  is the discontinuity cost,  $[ \ ]$  are the Iverson brackets, and  $H_\theta(\mathbf{f})$  represents the hedgehog shape constraints. For SOL we define the hedgehog constraints as:

$$H_\theta(\mathbf{f}) = \sum_{k \in \mathcal{L} \setminus \mathcal{B}} \sum_{(p, q) \in \mathcal{E}_k(\theta)} w_\infty [f_p^k = 1, f_q^k \neq 1], \quad (2.2)$$

where  $w_\infty$  is a very large constant,  $\mathcal{E}_k(\theta)$  is the set of edges used to approximate the hedgehog shape prior constraint (1.12) for label  $k$ , and  $\theta$  is the tightness of the hedgehog constraints (1.9). The three terms of our energy are discussed in further details below.

The first term in our energy is usually referred to as the *data term* or the *regional term*.  $D_p^\gamma(f_p)$  is the cost of assigning pixel  $p$  to label  $f_p$  and usually computed as:

$$D_p^\gamma(f_p) = -\ln \Pr(I_p \mid \gamma_{f_p}), \quad (2.3)$$

where  $\Pr$  is the probability of the color intensity  $I_p$  at pixel  $p$  given the color model  $\gamma_{f_p}$ . Although, a pixel may be assigned to more than one foreground label in SOL, it will

only pay the cost of being assigned to the foreground once. It will use a color model derived from all the foreground seeds. Conceptually, a pixel with multiple foreground labels simply reduces to a pixel labeled foreground.

The *smoothness term* in (2.1) is used to discourage label discontinuities between neighbouring pixels. Our smoothness term is somewhat different from the commonly used Potts model [4] for multi-label segmentation. In Potts model, a pair of neighbouring pixels pays for a discontinuity if they are assigned to two different labels. In contrast, in our smoothness model, a pair of neighbouring pixels pays for a discontinuity only between background and non-background labels. The discontinuity cost  $w_{pq}$  between  $p$  and  $q$  is a non-negative weight that is application specific.

The final term in our energy imposes *hedgehog constraints* on each foreground label individually. Each foreground label has its surface normals locally constrained by a vector field<sup>1</sup>. This is in contrast to [18] where the foreground was restricted by a single vector field which occasionally resulted in segmentation errors, see Fig. 2.1.

## 2.3 SOL Optimization

We propose **Grabcut for Overlapping-Labels (GOL)**, Alg. 2, to find an approximate solution to (2.1). Initially, the user provides seeds denoting each individual foreground label and the background. Similar to [24], we start from a trivial solution<sup>2</sup> and optimize (2.1) in block-coordinate descent. Specifically, our framework alternates between (a) fixing current labeling  $\hat{\mathbf{f}}$  and optimizing color models  $\gamma_{\mathcal{B}}$  and  $\gamma_{\mathcal{F}}$ , and (b) fixing the color models and optimizing labeling  $\mathbf{f}$ . Our framework stops when both steps no longer decrease (2.1).

As shown in GOL step 4, given current labeling  $\hat{\mathbf{f}}$ , we use the EM-Algorithm [9] to re-estimate the foreground and background color models. In step 5, given  $\gamma_{\mathcal{B}}$  and  $\gamma_{\mathcal{F}}$ , optimizing (2.1) reduces to a multi-labeling problem that allows labels to overlap which is NP-hard, see proof in Section 2.5. One of the main contributions of this work and a key step in GOL is Alg. 3 **Expansion-Contraction Moves (EC-Moves)**. Given  $\gamma_{\mathcal{B}}$  and  $\gamma_{\mathcal{F}}$ , EC-Moves finds an approximate labeling to energy (2.1) via a series of binary moves.

---

<sup>1</sup>In our case, it is the gradient of the user-scribble’s distance map.

<sup>2</sup>All pixels are background except for the user seeds, they are assigned to their corresponding labels.



---

**Algorithm 2: Grabcut for Overlapping-Labels (GOL)**

---

```

1 Input:  $\hat{\mathbf{f}}$   $\leftarrow$  trivial initial labeling (user-seeds)
2  $E^* \leftarrow \infty$ 
3 repeat
4    $\gamma_{\mathcal{B}}, \gamma_{\mathcal{F}} \leftarrow$  Re-estimate color models given  $\hat{\mathbf{f}}$  using EM
5    $\mathbf{f}^t \leftarrow \arg \min_{\mathbf{f}} E(\mathbf{f}, \gamma_{\mathcal{B}}, \gamma_{\mathcal{F}})$  given  $\gamma_{\mathcal{B}}$  and  $\gamma_{\mathcal{F}}$  using EC-Moves (Alg. 3)
6   if  $E(\mathbf{f}^t, \gamma_{\mathcal{B}}, \gamma_{\mathcal{F}}) < E^*$ 
7      $\hat{\mathbf{f}} \leftarrow \mathbf{f}^t$ 
8      $E^* \leftarrow E(\mathbf{f}^t, \gamma_{\mathcal{B}}, \gamma_{\mathcal{F}})$ 
9 until converged (no decrease in  $E$ )
10 return  $\hat{\mathbf{f}}$ 

```

---



---

**Algorithm 3: Expansion-Contraction Moves (EC-Moves)**

---

```

1 Input:  $\hat{\mathbf{f}}$   $\leftarrow$  current labeling
2 repeat
3    $\alpha \leftarrow$  random label  $\in \mathcal{L} \setminus \mathcal{B} \equiv \{F_1, \dots, F_n\}$ 
4   if coin toss is heads
5      $\mathbf{f}^\alpha \leftarrow \arg \min_{\mathbf{f}} E(\mathbf{f})$  where  $\mathbf{f}$  is an  $\alpha$ -expansion of  $\hat{\mathbf{f}}$ 
6   else
7      $\mathbf{f}^\alpha \leftarrow \arg \min_{\mathbf{f}} E(\mathbf{f})$  where  $\mathbf{f}$  is an  $\alpha$ -contraction of  $\hat{\mathbf{f}}$ 
8   if  $\mathbf{f}^\alpha < \hat{\mathbf{f}}$ 
9      $\hat{\mathbf{f}} \leftarrow \mathbf{f}^\alpha$ 
10 until converged (no decrease for all  $\alpha \in \mathcal{L}$ )

```

---

The EC-Moves Alg. randomly traverses the foreground labels. At every iteration, an arbitrarily label  $\alpha$  randomly chooses to either expand or contract its support region. The current labeling is updated if the expansion (or contraction) results in a lower energy. EC-Moves terminates when no move leads to a new labeling with a lower energy. The expansion and contraction moves used in EC-Moves are elaborated on in Section 2.4.

## 2.4 Binary Optimization Moves

In this Section we explain the binary expansion and contraction moves used in EC-Moves. We only expand (or contract) a foreground label by casting the move as a binary move (i.e. pixels have the binary choice to gain/lose a label or stay the same). Our reasoning for eliminating background contraction is that it is not a binary move as the background has more than one foreground label it may switch to. This contrasts the binary foreground

contraction which may only switch to the single background label (or stay the same). Although background expansion is a binary move, utilizing it did not affect the results in practice.

### 2.4.1 $\alpha$ -Expansion Move:

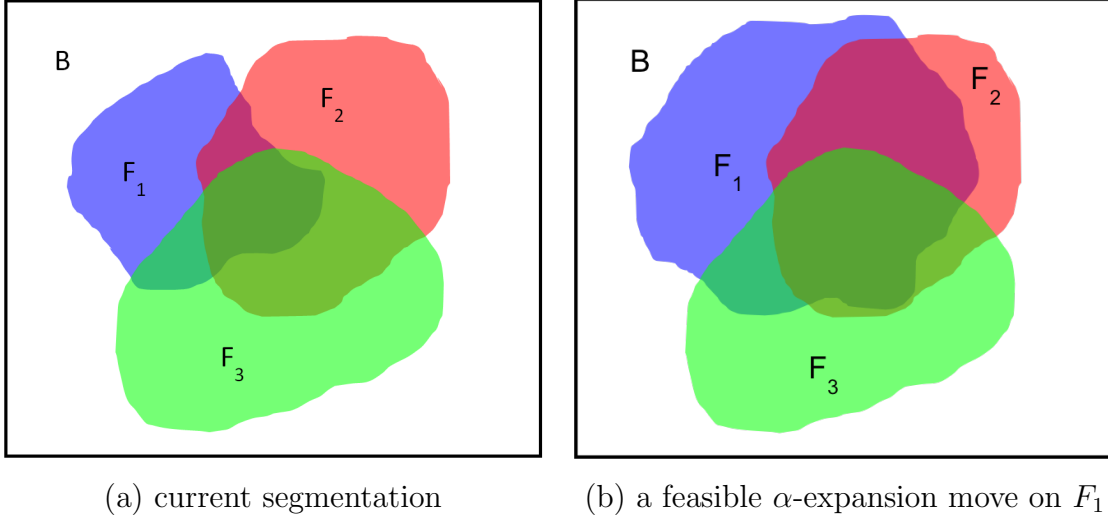


Figure 2.5: (a) shows a sample of a current segmentation. (b) shows a feasible expansion for label  $F_1$ . Notice how  $F_1$  expands while  $F_2$  and  $F_3$  remain unaffected.

In an expansion move on a foreground label  $\alpha \in \mathcal{L} \setminus \mathcal{B} \equiv \{F_1, \dots, F_n\}$ , a pixel is given a binary choice that depends on whether it is currently a background or foreground pixel. A currently labeled background pixel is given the choice to either stay the same or switch to  $\alpha$ . A currently labeled foreground pixel is given the choice to either stay the same or add  $\alpha$  to its set of foreground labels.

Figure 2.5 illustrates an example of a possible expansion on the label  $F_1$ . There are three things to note about pixels that were affected by the expansion. First, some background pixels decided to switch to  $F_1$ . Secondly, some foreground pixels that did not include  $F_1$  added  $F_1$  to their list of foreground labels. Lastly, no other foreground label, i.e.  $F_2$  and  $F_3$ , gained or lost any pixels.

Given the current labeling  $\hat{f}$  and color models  $\gamma_F$  and  $\gamma_B$ , an expansion move on  $\alpha \in \{F_1, \dots, F_n\}$  can be formulated as a binary energy as follows:

$$E_\alpha^e(\mathbf{x}) = \sum_{p \in \Omega} D_p^\gamma(\hat{f}_p) \bar{x}_p + D_p^\gamma(\alpha) x_p + \lambda \sum_{p,q \in \mathcal{N}} w_{pq} V(x_p, x_q) + \sum_{(p,q) \in \mathcal{E}_\alpha(\theta)} S_{pq}(x_p, x_q), \quad (2.4)$$

where  $\mathbf{x} = \{x_p \in \{0, 1\} \mid \forall p \in \Omega\}$ , if  $x_p$  is 1 then a pixel switches to  $\alpha$ ,

$$V(x_p, x_q) = \begin{cases} [\hat{f}_p \neq \hat{f}_q] & \text{if } x_p = 0, x_q = 0 \\ [\mathcal{B} = \hat{f}_q] & \text{if } x_p = 1, x_q = 0 \\ [\hat{f}_p = \mathcal{B}] & \text{if } x_p = 0, x_q = 1 \\ 0 & \text{if } x_p = 1, x_q = 1 \end{cases} \quad (2.5)$$

and

$$S(u, v) = \begin{cases} w_\infty & \text{if } u = 1, v = 0 \\ 0 & \text{otherwise.} \end{cases} \quad (2.6)$$

### Submodularity

To prove that (2.4) is submodular we need to prove that  $V(x_p, x_q)$  and  $S(u, v)$  are submodular for any  $\alpha \in \mathcal{L} \setminus \mathcal{B} \equiv \{F_1, \dots, F_n\}$ . We will first consider  $V(x_p, x_q)$  as defined in (2.5). For  $V$  to be submodular it must satisfy the submodularity condition (1.4), i.e.

$$V(0, 0) + V(1, 1) \leq V(0, 1) + V(1, 0) \quad (2.7)$$

$$[\hat{f}_p \neq \hat{f}_q] + 0 \leq [\hat{f}_p = \mathcal{B}] + [\hat{f}_q = \mathcal{B}] \quad \text{by substituting (2.5).} \quad (2.8)$$

Inequality (2.8) depends on the current labeling of pixels  $p$  and  $q$ , i.e.  $\hat{f}_p$  and  $\hat{f}_q$ . As such, to show that (2.8) always holds we will consider all possible variations of  $\hat{f}_p$  and  $\hat{f}_q$ . As a pixel's labeling can be either  $\mathcal{F}$  or  $\mathcal{B}$ , we have the following cases to consider

$$\begin{aligned} \text{if } \hat{f}_p = \mathcal{B}, \hat{f}_q = \mathcal{B} & \text{ then } 0 + 0 \leq 1 + 1 \\ \text{if } \hat{f}_p = \mathcal{F}, \hat{f}_q = \mathcal{B} & \text{ then } 1 + 0 \leq 0 + 1 \\ \text{if } \hat{f}_p = \mathcal{B}, \hat{f}_q = \mathcal{F} & \text{ then } 1 + 0 \leq 1 + 0 \\ \text{if } \hat{f}_p = \mathcal{F}, \hat{f}_q = \mathcal{F} & \text{ then } 0 + 0 \leq 0 + 0. \end{aligned}$$

For all possible cases, we can see that the submodularity constraint holds so we can claim that  $V(x_p, x_q)$  is submodular.

To prove that  $S(u, v)$  is submodular the submodularity constraint (2.8) must hold

$$S(0, 0) + S(1, 1) \leq S(0, 1) + S(1, 0) \quad (2.9)$$

$$0 + 0 \leq 0 + 1 \quad \text{by substituting (2.6).} \quad (2.10)$$

Therefore,  $S(u, v)$  is submodular. Since both  $V(x_p, x_q)$  and  $S(u, v)$  are submodular then their sum is also submodular and we can claim that (2.4) is submodular as well.

### 2.4.2 $\alpha$ -Contraction Move:

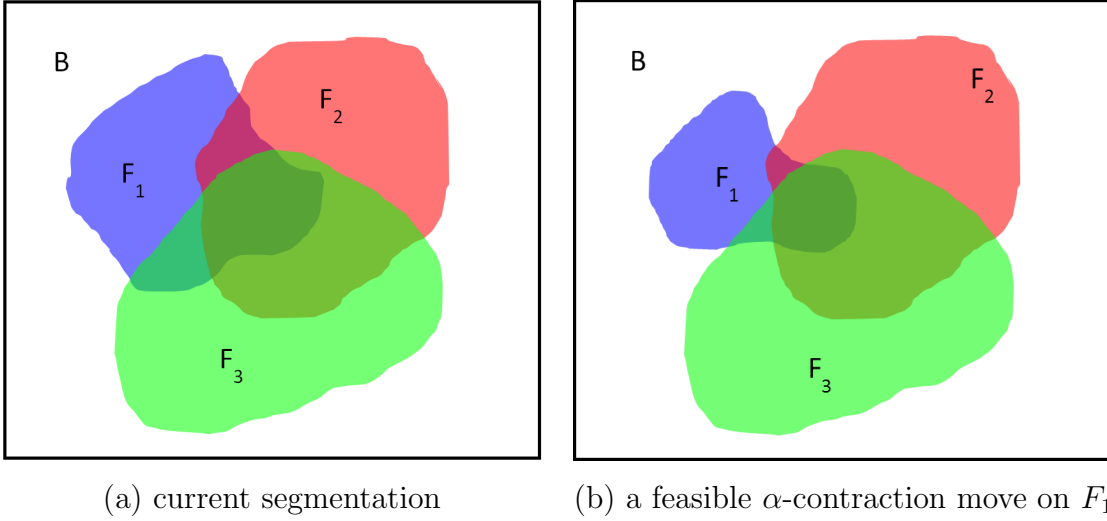


Figure 2.6: (a) shows a sample of a current segmentation. (b) shows a feasible contraction for label  $F_1$ . Notice how  $F_1$  loses pixel support while  $F_2$  and  $F_3$  remain unaffected.

In a contraction move on any foreground label  $\alpha \in \mathcal{L} \setminus \mathcal{B} \equiv \{F_1, \dots, F_n\}$ , every pixel has a binary choice depending on its current labelling. On the one hand, a pixel only labeled  $\alpha$  is given the choice to either stay the same or switch to the background label. On the other hand, any other pixel is given the choice to either stay the same or remove  $\alpha$  from its set of labels, if it was there.

Figure 2.6 illustrates an example of a possible contraction on the label  $F_1$ . There are three things to note about pixels that were affected by the contraction. First, some pixels that were *only* labeled  $F_1$  decided to switch to the background label. Secondly, some pixels that were assigned to multiple foreground labels including  $\alpha$  lost their  $\alpha$  label but maintained their other foreground labels. Lastly, no other foreground label gained or lost any pixels (i.e. their labeling stayed the same). It is worth mentioning that pixels with multiple foreground labels that did not include  $\alpha$  would not be affected by the contraction.

To simplify our notation, we use an indicator variable  $\delta_p$  that depends on the current labeling to indicate whether a pixel  $p$  is *only* labeled  $\alpha$  or not:

$$\delta_p = \begin{cases} 1 & \forall k \in \mathcal{L} \setminus \alpha, \quad \hat{f}_p^k = 0 \\ 0 & \exists k \in \mathcal{L} \setminus \alpha \text{ s.t. } \hat{f}_p^k = 1. \end{cases} \quad (2.11)$$

Given current labeling  $\hat{f}$ , a contraction move on  $\alpha$  can be formulated as:

$$E_\alpha^c(\mathbf{x}) = \underbrace{\sum_{\substack{p \in \Omega \\ \delta_p = 1}} D_p^\gamma(\hat{f}_p) \bar{x}_p + D_p^\gamma(\mathcal{B}) x_p}_{\text{pixels assigned only to } \alpha} + \underbrace{\sum_{\substack{p \in \Omega \\ \delta_p = 0}} D_p^\gamma(\hat{f}_p)}_{\text{other pixels}} + \lambda \sum_{p, q \in \mathcal{N}} w_{pq} V(x_p, x_q) + \sum_{(p, q) \in \mathcal{E}_\alpha(\theta)} S(x_q, x_p) \quad (2.12)$$

where  $S(u, v)$  is as previously defined in (2.6) and  $V(x_p, x_q)$  is a binary pairwise potential defined as

$$V(x_p, x_q) = \begin{cases} [\hat{f}_p \neq \hat{f}_q] & \text{if } x_p = 0, x_q = 0 \\ [\hat{f}_p \neq \mathcal{B}] \delta_q + [\hat{f}_p \neq \hat{f}_q] \bar{\delta}_q & \text{if } x_p = 0, x_q = 1 \\ [\mathcal{B} \neq \hat{f}_q] \delta_p + [\hat{f}_p \neq \hat{f}_q] \bar{\delta}_p & \text{if } x_p = 1, x_q = 0 \\ [\hat{f}_p \neq \hat{f}_q] \bar{\delta}_p \bar{\delta}_q + [\mathcal{B} \neq \hat{f}_q] \delta_p \bar{\delta}_q + [\hat{f}_p \neq \mathcal{B}] \bar{\delta}_p \delta_q & \text{if } x_p = 1, x_q = 1. \end{cases} \quad (2.13)$$

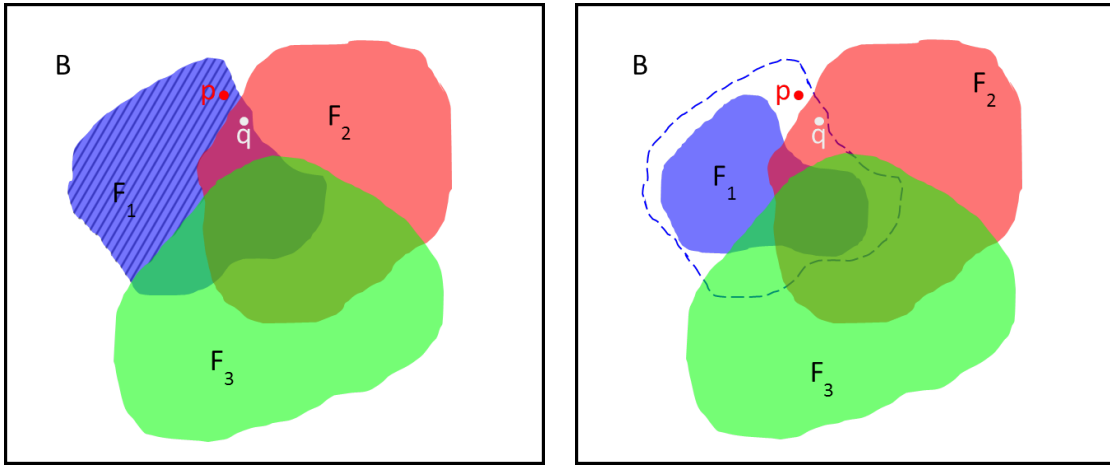


Figure 2.7: (a) the hatched area shows the pixels that may change their data term in a contraction (i.e. switch from foreground to background). Pixel  $p$  is a part of this set and pixel  $q$  is not. (b) shows an example of a feasible contraction. Notice that for pixel  $p$ , the data term changed to background while pixel  $q$  remained foreground.

Figure 2.7 illustrates the different choices given to pixels  $p$  and  $q$  during an  $\alpha = F_1$  contraction. Pixel  $p$  falls under the set of pixels that are only labeled  $F_1$ , i.e. the hatched area in Fig. 2.7(a). Pixels in this area have the option to switch from  $F_1$  to the background or to stay the same. Pixel  $q$  is not only labeled  $F_1$  and thus, it is given the option to either stay the same or lose the  $F_1$  label. Figure 2.7(b) shows a possible contraction in

which both pixel  $p$  and pixel  $q$  decided to lose their the  $F_1$  label. Pixel  $p$  switched to background while  $q$  remained foreground but  $F_1$  is no longer part of its foreground label set.

### Submodularity

To prove that (2.12) is submodular we need to prove that  $V(x_p, x_q)$  and  $S(u, v)$  are submodular. In order for  $V(x_p, x_q)$  to be submodular it must satisfy the submodularity condition (1.4), i.e.

$$V(0, 0) + V(1, 1) \leq V(0, 1) + V(1, 0). \quad (2.14)$$

By substituting (2.13) in (2.14)

$$\begin{aligned} [\hat{f}_p \neq \hat{f}_q] + [\hat{f}_p \neq \hat{f}_q] \bar{\delta}_p \bar{\delta}_q + [\mathcal{B} \neq \hat{f}_q] \delta_p \bar{\delta}_q + [\hat{f}_p \neq \mathcal{B}] \bar{\delta}_p \delta_q \leq \\ [\mathcal{B} \neq \hat{f}_q] \delta_p + [\hat{f}_p \neq \hat{f}_q] \bar{\delta}_p + [\hat{f}_p \neq \mathcal{B}] \delta_q + [\hat{f}_p \neq \hat{f}_q] \bar{\delta}_q. \end{aligned} \quad (2.15)$$

Inequality (2.15) depends on the current labeling of  $p$  and  $q$ , i.e.  $\hat{f}_p$  and  $\hat{f}_q$ . Furthermore, (2.15) also depends on the indicator variables  $\delta_p$  and  $\delta_q$ . Thus, to show that (2.15) always hold will consider all possibilities of  $\hat{f}_p$ ,  $\hat{f}_q$ ,  $\delta_p$  and  $\delta_q$ . There are only 9 cases to check as the other remaining cases will lead to invalid configuration, e.g.  $\hat{f}_p = \mathcal{B}$  while  $\delta_p = 1$ . The 9 valid cases are

$$\begin{aligned} \text{if } \hat{f}_p = \mathcal{B}, \hat{f}_q = \mathcal{B}, \delta_p = 0, \delta_q = 0 \quad \text{then } 0 + 0 + 0 + 0 &\leq 0 + 0 + 0 + 0 \\ \text{if } \hat{f}_p = \mathcal{F}, \hat{f}_q = \mathcal{B}, \delta_p = 1, \delta_q = 0 \quad \text{then } 1 + 0 + 0 + 0 &\leq 0 + 0 + 0 + 1 \\ \text{if } \hat{f}_p = \mathcal{F}, \hat{f}_q = \mathcal{B}, \delta_p = 0, \delta_q = 0 \quad \text{then } 1 + 1 + 0 + 0 &\leq 0 + 1 + 0 + 1 \\ \text{if } \hat{f}_p = \mathcal{B}, \hat{f}_q = \mathcal{F}, \delta_p = 0, \delta_q = 1 \quad \text{then } 1 + 0 + 0 + 0 &\leq 0 + 1 + 0 + 0 \\ \text{if } \hat{f}_p = \mathcal{B}, \hat{f}_q = \mathcal{F}, \delta_p = 0, \delta_q = 0 \quad \text{then } 1 + 1 + 0 + 0 &\leq 0 + 1 + 0 + 1 \\ \text{if } \hat{f}_p = \mathcal{F}, \hat{f}_q = \mathcal{F}, \delta_p = 1, \delta_q = 1 \quad \text{then } 0 + 0 + 0 + 0 &\leq 1 + 0 + 1 + 0 \\ \text{if } \hat{f}_p = \mathcal{F}, \hat{f}_q = \mathcal{F}, \delta_p = 1, \delta_q = 0 \quad \text{then } 0 + 0 + 1 + 0 &\leq 1 + 0 + 0 + 0 \\ \text{if } \hat{f}_p = \mathcal{F}, \hat{f}_q = \mathcal{F}, \delta_p = 0, \delta_q = 1 \quad \text{then } 0 + 0 + 0 + 1 &\leq 0 + 0 + 1 + 0 \\ \text{if } \hat{f}_p = \mathcal{F}, \hat{f}_q = \mathcal{F}, \delta_p = 0, \delta_q = 0 \quad \text{then } 0 + 0 + 0 + 0 &\leq 0 + 0 + 0 + 0. \end{aligned}$$

We can see that the submodularity condition (1.4) holds for all the possible valid cases. Thus, we conclude that  $V(u, v)$  is submodular. Since  $V(u, v)$  is submodular and  $S(u, v)$  is submodular, as previously shown in Section 2.4.1, then their sum is also submodular. Thus, energy (2.12) is submodular.

## 2.5 NP-Hardness Proof

In general, the SOL problem is NP-hard. To prove this we will reduce the Set Cover problem instance to SOL in polynomial time, i.e. Set Cover  $\leq_P$  SOL. In a Set Cover problem we are given a set of  $n$  elements that forms universe  $U = \{u_1, u_2, \dots, u_n\}$ , and a set of  $m$  subsets  $S = \{S_i \subseteq U \mid \forall i \in [1, m]\}$ . The aim is to find the least number of subsets such that their union covers the universe  $U$ .

A SOL problem (2.1) is defined by the pixel set  $\Omega$ , the label set  $\mathcal{L}$ , the data term  $D_p^\gamma(f_p)$ , the neighbourhood system  $\mathcal{N}$ , and the set of hedgehog constraints  $\mathcal{E}$ . Given  $U$  and  $S$  of a Set Cover problem we can construct the corresponding SOL problem as follows:

**Pixel set:**  $\Omega := \{U \cup A\}$  where  $A = \{a_i \mid \forall i \in [1, m]\}$ .  $A$  is a set of auxiliary pixels/nodes. In this section will refer to pixels as nodes. For each set  $S_i$ , we introduce an auxiliary node  $a_i$  that will be used as an indicator of whether  $S_i$  is one of the selected sets to cover  $U$  or not.

**Label set:**  $\mathcal{L} := \{\mathcal{B} \cup \{F_i \mid \forall i \in [1, m]\}\}$  where the label  $F_i$  corresponds to subset  $S_i$  and  $\mathcal{B}$  is the background label. In the context of the Set Cover problem the background label serves no purpose but it was kept for ease of juxtaposition to the SOL problem.

**Data term:** We define the data term as follows

$$D_u^\gamma(\ell) = \begin{cases} \infty & \forall u \in \Omega, \ell = \mathcal{B} & (2.16) \\ 0 & \forall u \in S_i, \ell = F_i \quad \forall i \in [1, m] & (2.17) \\ \infty & \forall u \notin S_i, \ell = F_i \quad \forall i \in [1, m] & (2.18) \\ 1 & u = a_i, \ell = F_i \quad \forall i \in [1, m] & (2.19) \\ 0 & u = a_i, \ell = F_j \quad \forall i, j \in [1, m], i \neq j. & (2.20) \end{cases}$$

In (2.16) the data term prohibits any node to be assigned to the background label  $\mathcal{B}$ . In (2.17) and (2.18) the data term prohibits a node  $u \notin S_i$  to be given the label  $F_i$ . In (2.19) and (2.20) the data term ensures that our energy increases by 1 if  $a_i$  is give the label  $F_i$ .

**Nodes' neighbour system:** In Set Cover there no notion of neighbourhood between the nodes, thus  $\mathcal{N} := \phi$ .

**Hedgehog constraints:** For a given set  $S_i$  we define the corresponding hedgehog constraints  $\mathcal{E}_{F_i}$  of label  $F_i$  as follows:

$$\mathcal{E}_{F_i} := \mathcal{C}_{F_i} \cup \mathcal{I}_{F_i}$$

, where the set of *connectedness* constraints is  $\mathcal{C}_{F_i} := \{(u, v) \mid \forall u, v \in S_i, u \neq v\}$  and the set of *indicator* constraints is  $\mathcal{I}_{F_i} := \{(u, a_i) \mid \forall u \in S_i, a_i \in A\}$ . The edges<sup>3</sup> in  $\mathcal{C}_{F_i}$  ensure that if a node  $u \in S_i$  is given the label  $F_i$  then every other node in the set  $S_i$  will also be given label  $F_i$ . The edges in  $\mathcal{I}_{F_i}$  ensures that if any node  $u \in S_i$  is given the label  $F_i$  then the corresponding auxiliary node  $a_i$  of  $S_i$  is given the label  $F_i$  as well.

**Objective:** The objective in Set Cover is to minimize the number of selected subsets to cover  $U$ , which is the energy of the corresponding SOL problem. Notice that if a node  $u \in S_i$  decides to be given the label  $F_i$  then  $a_i$  will be will also be given the label  $F_i$ . And, since  $D_{a_i}^\gamma(F_i) = 1$  by definition then we can conclude that our energy counts the number of subsets appearing in the final solution. Recall that there is no smoothness term and that the hedgehog term is 0 or  $\infty$ .

---

<sup>3</sup>We refer to an ordered pair nodes  $(u, v)$  as an edge.



# Chapter 3

# Experiments

Our experiments were conducted on both synthetic examples as well as real data (CT scans of the liver). We used these examples to compare  $\alpha$ -expansion [4] and its various modifications (i.e. adding a shape prior [18], allowing overlapping labels) with our approach. The same parameters were used for all approaches to ensure an accurate comparison. Further, this guaranteed that our final energy would be comparable.

### 3.1 Synthetic Example

We compare our approach for segmenting the synthetic example shown in Fig. 3.1 to [4, 18]. For all methods we utilized the same parameters, namely,  $\theta = 20$ ,  $\mathcal{N}$  was the 4-neighbourhood system, and  $\lambda = 2$ . The initial foreground and background color models were calculated using user-seeds. The pairwise discontinuity penalty between neighbouring pixels  $p$  and  $q$ , i.e.  $w_{pq}$ , was set to 1.

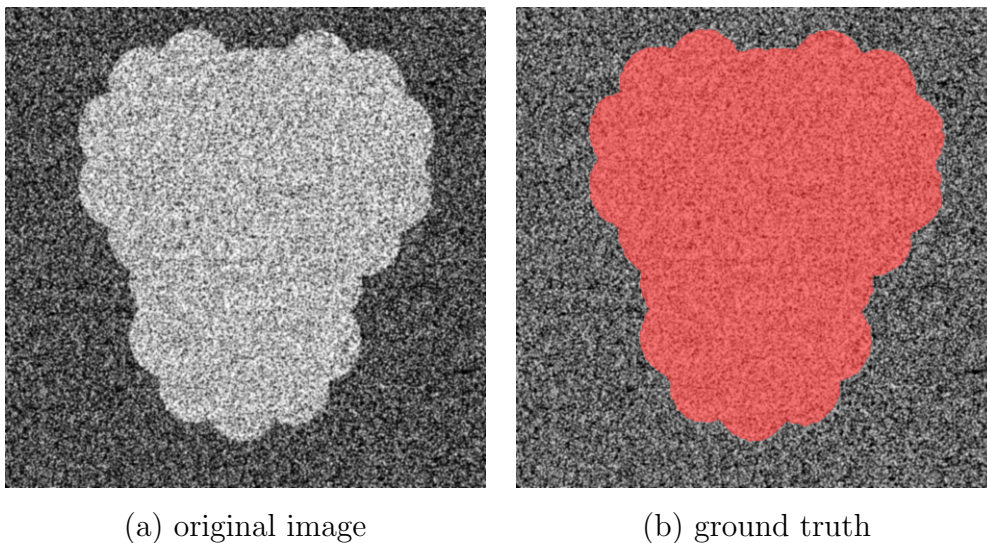


Figure 3.1: (a) depicts our original synthetic image while (b) displays the ground-truth segmentation.

In Figure 3.2, we display the segmentation results of the various approaches. The first row of Figure 3.2 corresponds to the classical binary segmentation using a complex seed [5, 18]. The first column depicts the multi-part seed. The second column illustrates the erroneous segmentation due to conflicting shape constraints as illustrated in Section 1.5. In this case, we only show one solution because binary segmentation can be solved optimally in polynomial time, see Section 1.2.3.

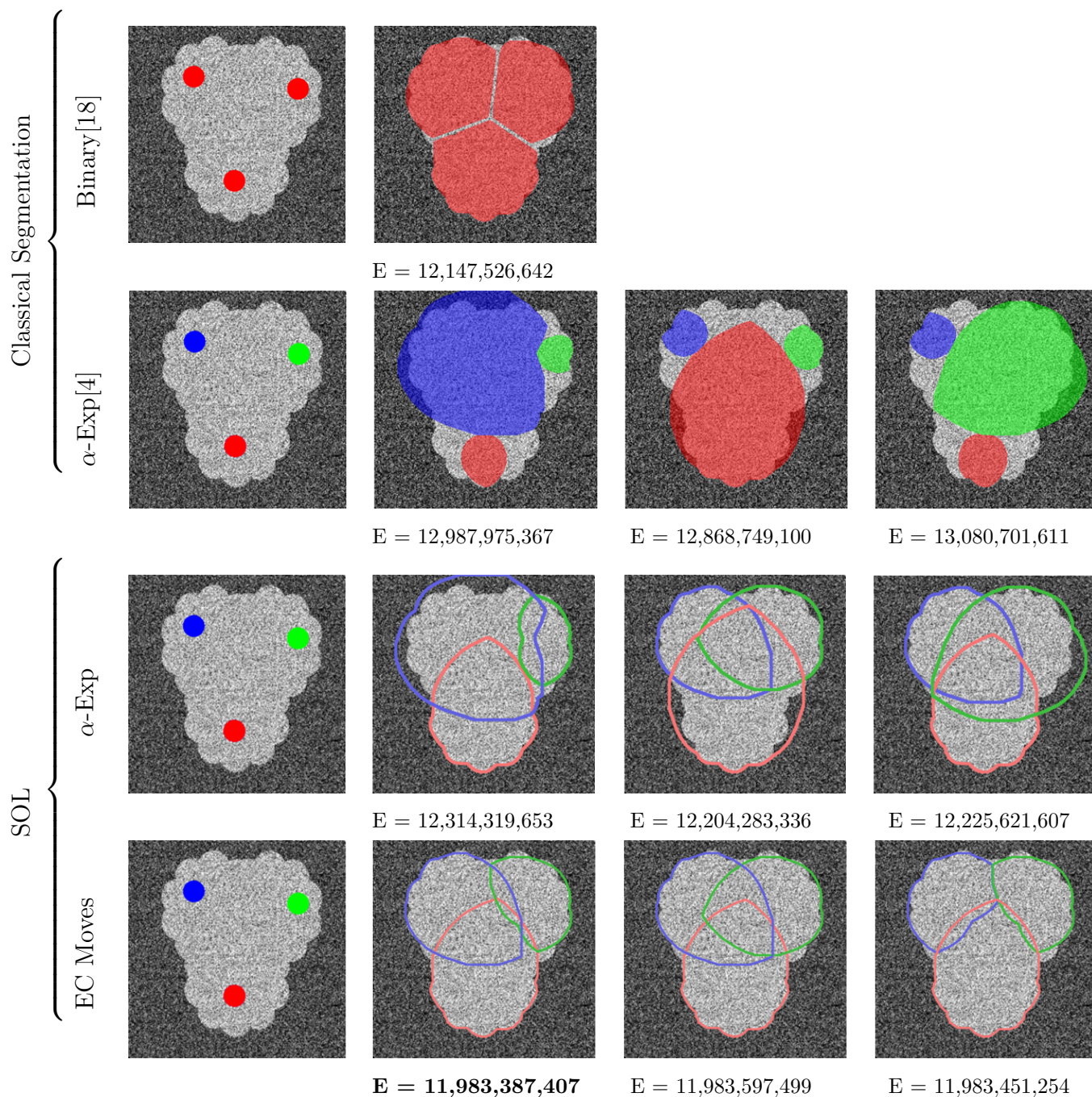


Figure 3.2: compares the results and energy of a synthetic example segmentation. Each row corresponds to a different segmentation model/optimization framework. The first two rows correspond to classical segmentation while the last two correspond to SOL. Columns 2-4 corresponds to different local minima by varying the order of expanding labels, if applicable. Note that the first row only has one segmentation because binary segmentation can be globally optimized. As you can see, our approach EC-moves achieves the best results.

The second row in Figure 3.2 corresponds to the classical  $\alpha$ -expansion [4]. The first column depicts the initial seeds. In contrast to the binary segmentation, the seeds are assigned to separate labels. The following columns show three different local minimas depending on the order of the expanding labels. It can be seen that this algorithm suffers from local minima in which the first expanding label attempts to cover the entire object, preventing the other labels from expanding.

The third row in Figure 3.2 corresponds to a modification of  $\alpha$ -expansion in which foreground labels are given the freedom to overlap. As the idea of overlapping labels has not been previously explored, we used our own implementation for this experiment and cover the results for the sake of comparison. It can be seen that allowing labels to overlap alleviates the local minima issue by allowing subsequent labels to expand further but the segmentation is still sensitive w.r.t. the order of expanding labels. This is due to a limitation in the background expansion, i.e. it does not allow pixels with multiple foreground labels to lose one of those foreground labels. The contraction moves we propose overcomes this problem by replacing the background expansion by a series of foreground labels contractions.

The final row in Figure 3.2 corresponds to our approach that uses the novel contraction moves. It can be seen that this provides a segmentation that is more robust towards the order of the expansion sequence. It is visually clear that our approach results in a better segmentation overall as labels are not expanding further than the edge of the object. As all experiments were run with the same parameters, the final energy is comparable across the different segmentation models/optimization frameworks and it can be seen that our approach finds the solution with lowest energy amongst them.

## 3.2 Liver Segmentation

Given a 2D CT slice, we compare the segmentation of the liver from the rest of the organs. We examined 6 separate subjects, comparing three different approaches:

- Approach 1 (A1):  $\alpha$ -expansion (without overlapping labels) [4]
- Approach 2 (A2):  $\alpha$ -expansion (without overlapping labels) with a shape prior [4, 18]
- Approach 3 (A3): our approach

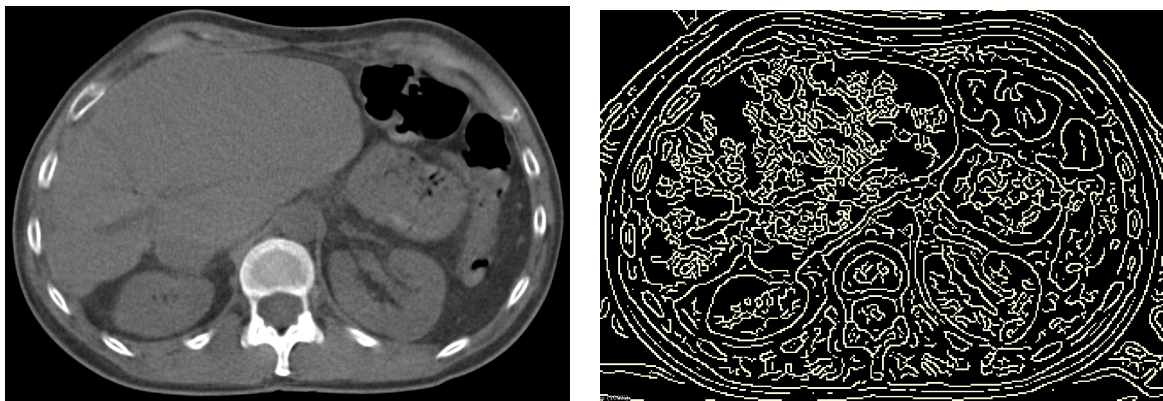
To ensure that the comparison focuses on the performance of the clustering aspect of Alg. 3, we used the ground truth segmentation to obtain the gaussian mixture color

models. For the shape prior we used hedgehog with  $\theta = 30$ . Finally, the smoothness term  $\lambda$  was set to 5 and we used canny edges to derive the discontinuity cost  $w_{pq}$ .

Canny-edges [6] is an edge detection operator, see Fig. 3.3. Canny-edges provides valuable information as to where the object boundary should occur, which may be incorporated into the pairwise discontinuity cost. The idea is to have a smaller discontinuity penalty along canny edges in comparison to discontinuities elsewhere. That is, if  $E$  is the set of edge pixels detected by Canny-edges then

$$w_{pq} = \begin{cases} 0.25 & (p \in E, q \notin E) \vee (q \in E, p \notin E) \\ 1 & (p \in E, q \in E) \vee (p \notin E, q \notin E) \end{cases} \quad (3.1)$$

where the 0.25 penalty is heuristically chosen. Decreasing the heuristically chosen weight encourages discontinuities to occur along the Canny-edges.



(a) original image

(b) set of Canny-edges pixels show in white

Figure 3.3: (a) is the input image. (b) shows the output of the Canny-edges algorithm [6], the set of edge pixels,  $E$ , is shown in white.

For qualitative comparison, we show the segmentation results of the various approaches. Figures 3.4 – 3.9 correspond to each of our 6 subjects, where each figure provides the seeds, the segmentation results of the three approaches, and the ground truth. For all subjects, it can be seen that our approach (A3) returns the most accurate segmentations, avoiding the over-segmenting issues of A1 and the under-segmenting issues of A2. This is further evidenced by how A3 consistently returns the lowest energy, corresponding with a segmentation that adheres to the colour models and stays within the liver edges.

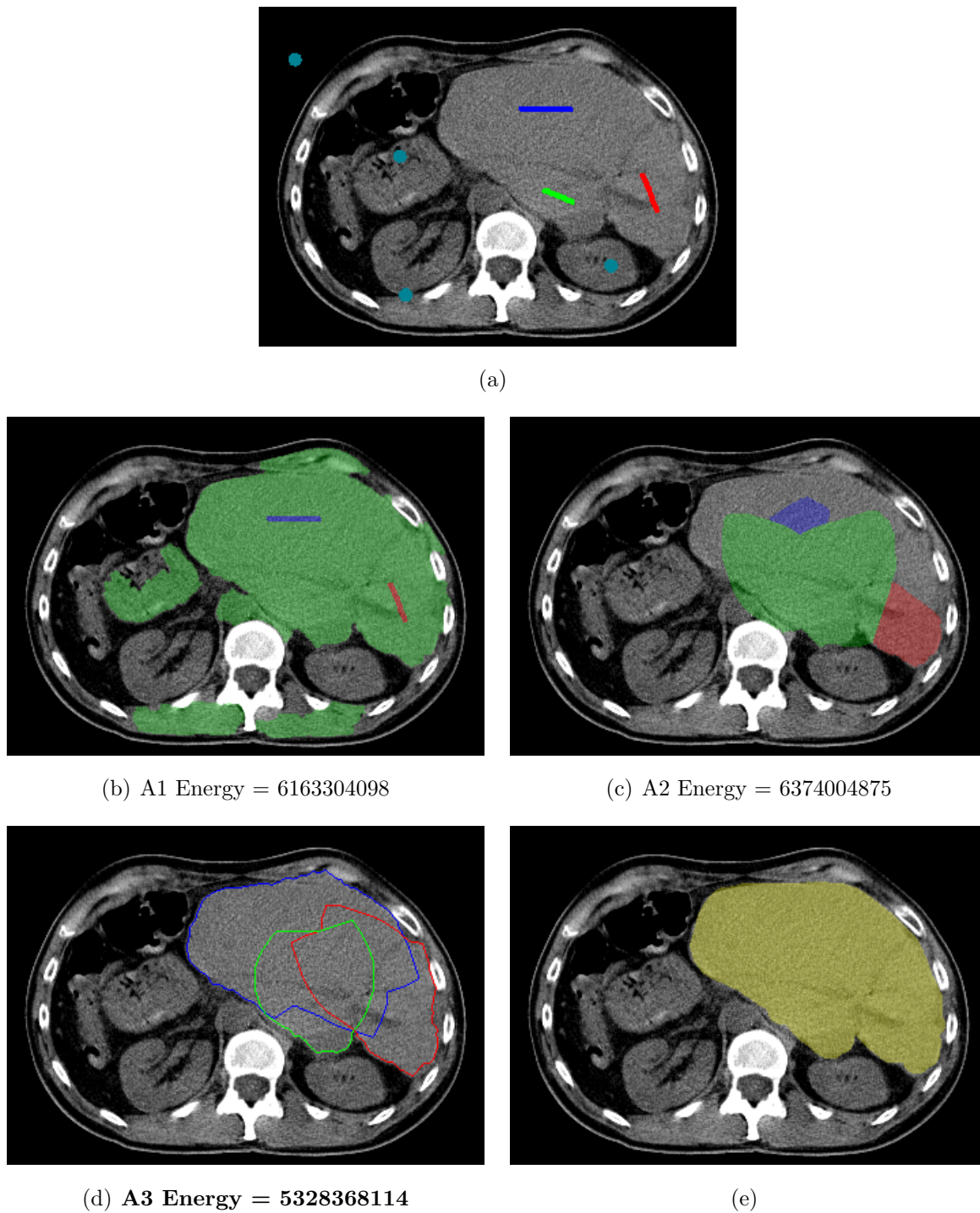
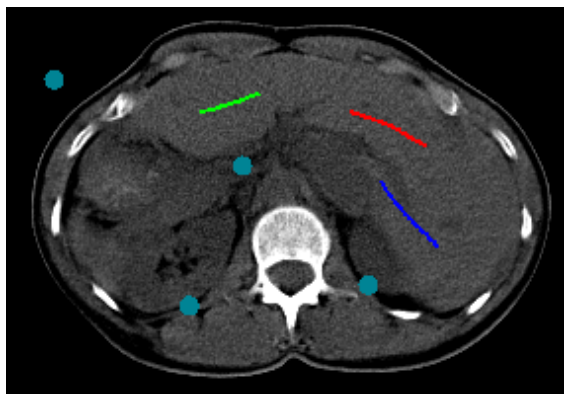
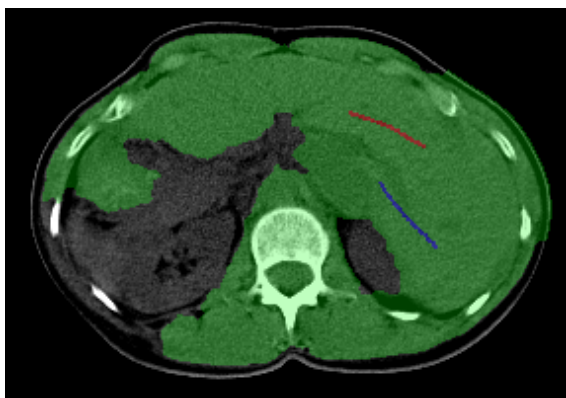


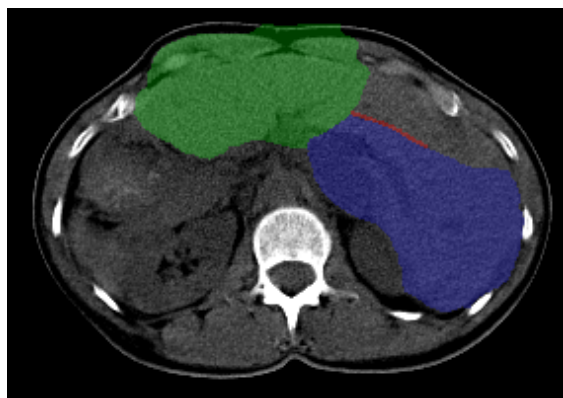
Figure 3.4: Experiment results for subject 1. (a) depicts the seeds where the teal circles are background seeds and the lines correspond to the separate foreground seeds. (b) shows the results of  $\alpha$ -expansion [A1], (c) shows the results of  $\alpha$ -expansion using the hedgehog shape prior [A2], and (d) shows the results of our approach [A3]. Finally, (e) is the ground truth.



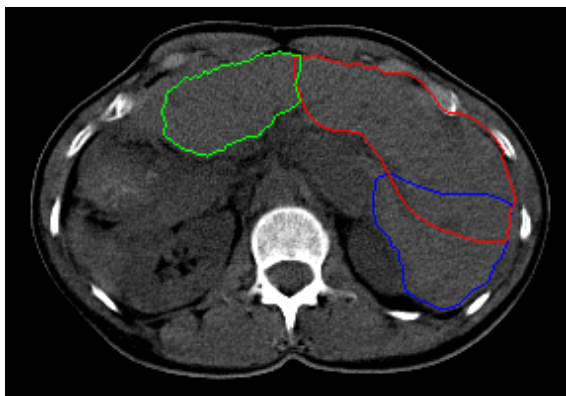
(a)



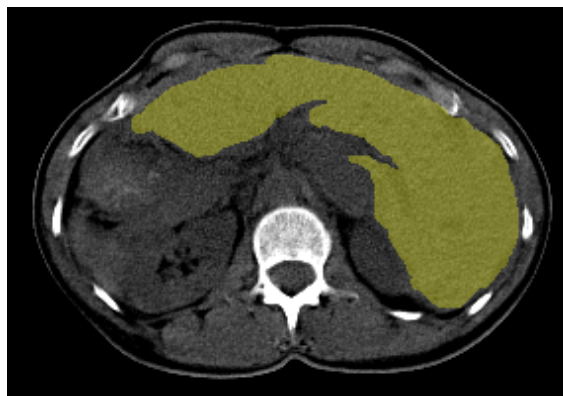
(b) A1 Energy = 3863877887



(c) A2 Energy = 3952234448



(d) A3 Energy = 3360526210



(e)

Figure 3.5: Experiment results for subject 2. (a) depicts the seeds where the teal circles are background seeds and the lines correspond to the separate foreground seeds. (b) shows the results of  $\alpha$ -expansion [A1], (c) shows the results of  $\alpha$ -expansion using the hedgehog shape prior [A2], and (d) shows the results of our approach [A3]. Finally, (e) is the ground truth.

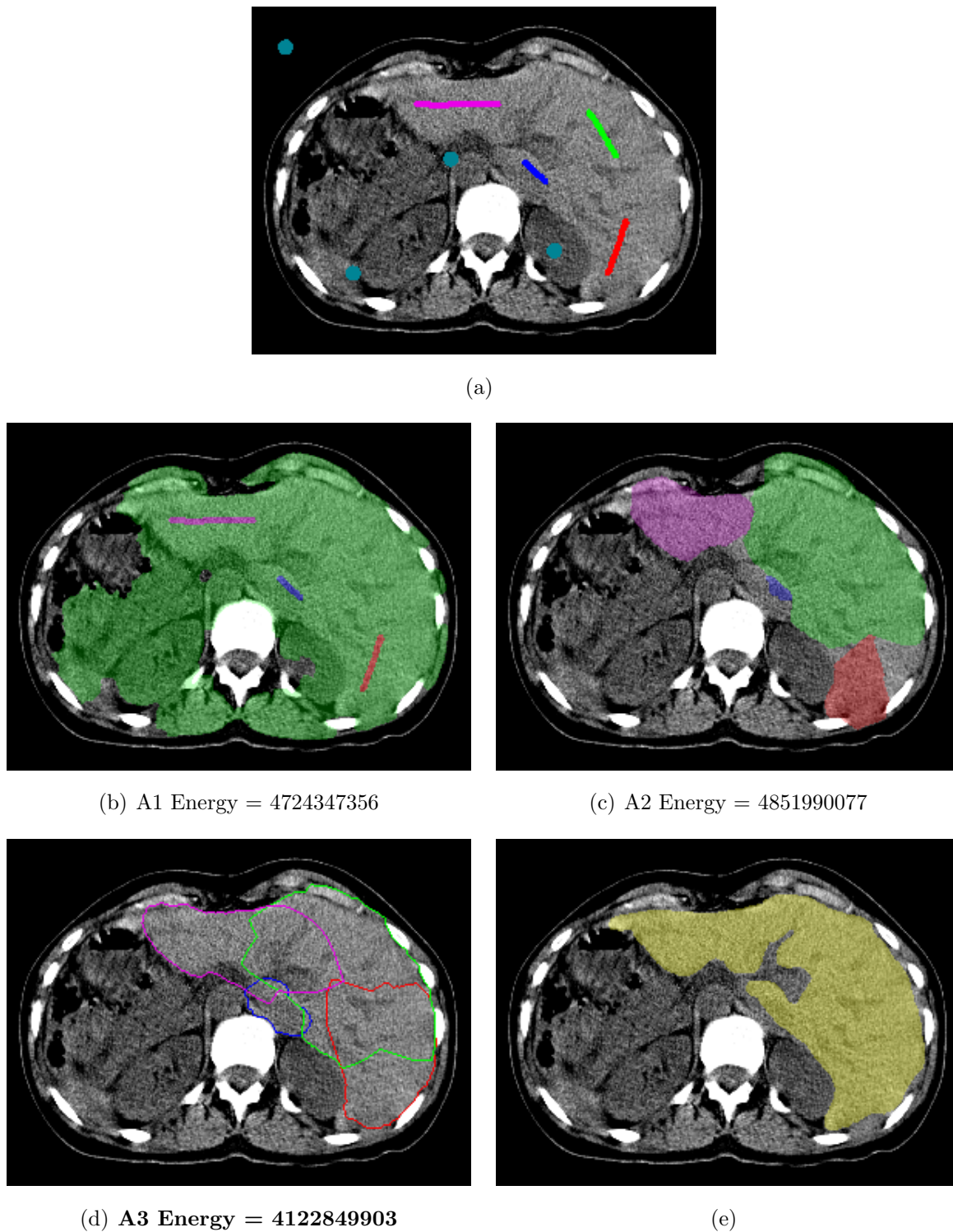
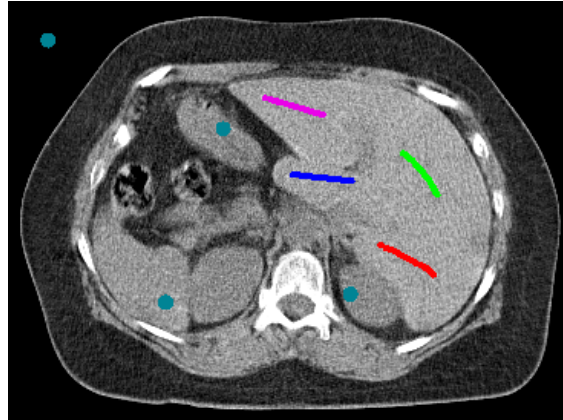
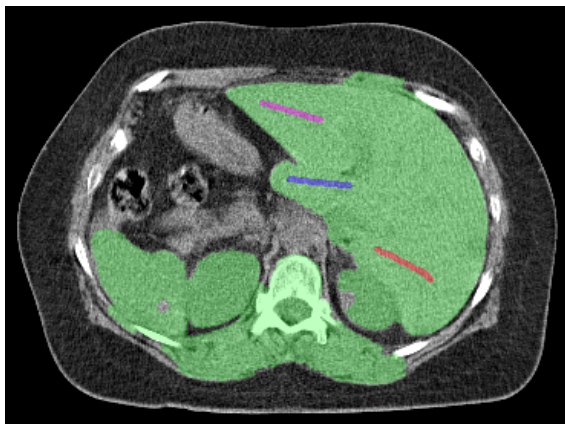


Figure 3.6: Experiment results for subject 3. (a) depicts the seeds where the teal circles are background seeds and the lines correspond to the separate foreground seeds. (b) shows the results of  $\alpha$ -expansion [A1], (c) shows the results of  $\alpha$ -expansion using the hedgehog shape prior [A2], and (d) shows the results of our approach [A3]. Finally, (e) is the ground truth.

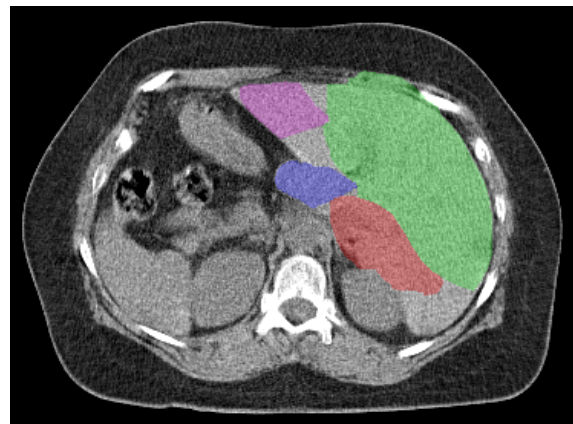




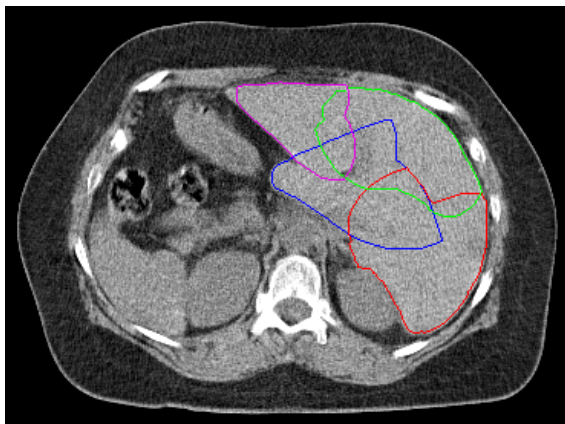
(a)



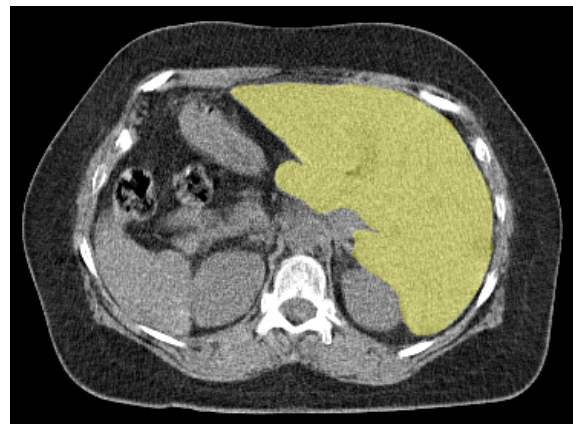
(b) A1 Energy = 7140357521



(c) A2 Energy = 7294224267



(d) A3 Energy = 6149069898



(e)

Figure 3.7: Experiment results for subject 4. (a) depicts the seeds where the teal circles are background seeds and the lines correspond to the separate foreground seeds. (b) shows the results of  $\alpha$ -expansion [A1], (c) shows the results of  $\alpha$ -expansion using the hedgehog shape prior [A2], and (d) shows the results of our approach [A3]. Finally, (e) is the ground truth.

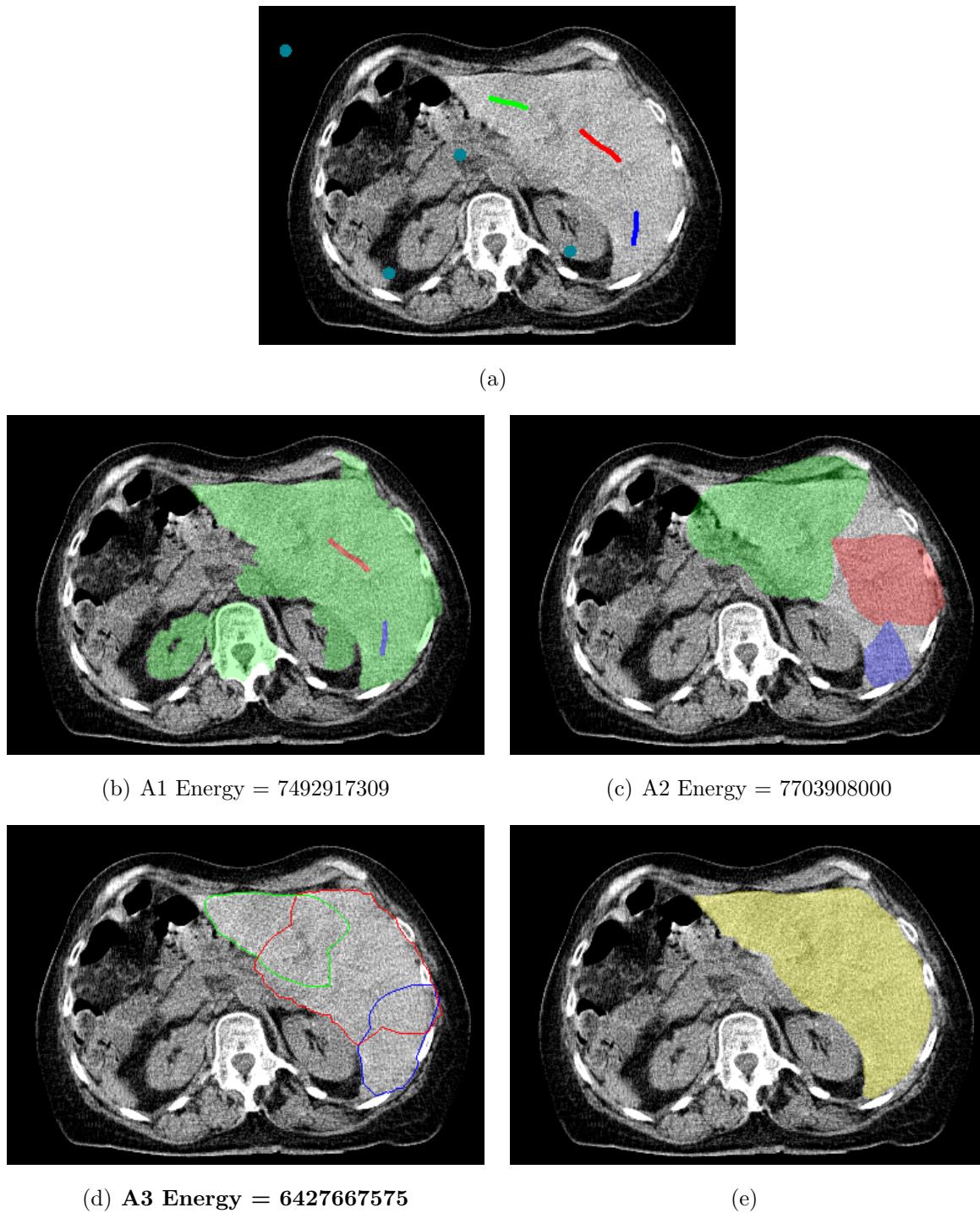
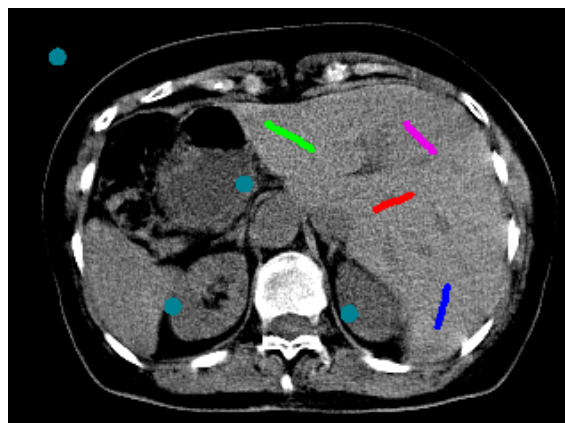
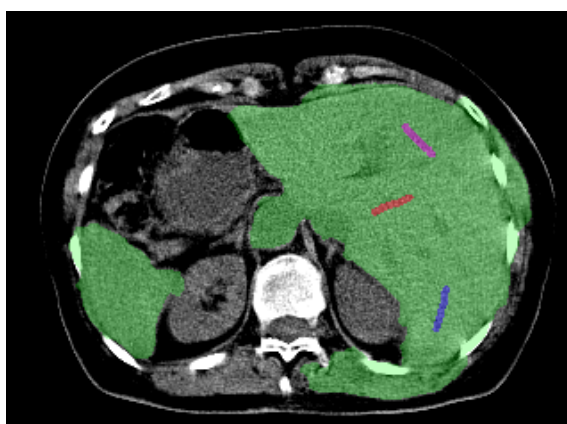


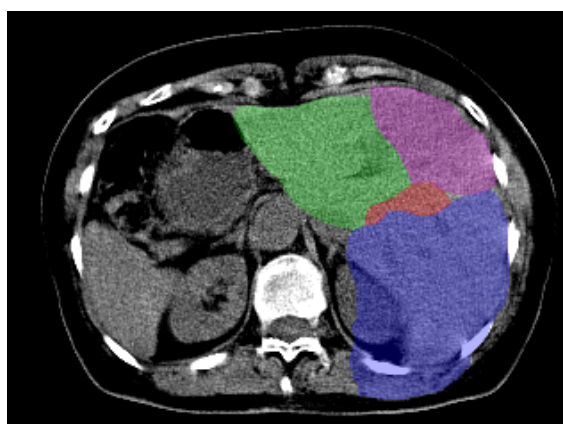
Figure 3.8: Experiment results for subject 5. (a) depicts the seeds where the teal circles are background seeds and the lines correspond to the separate foreground seeds. (b) shows the results of  $\alpha$ -expansion [A1], (c) shows the results of  $\alpha$ -expansion using the hedgehog shape prior [A2], and (d) shows the results of our approach [A3]. Finally, (e) is the ground truth.



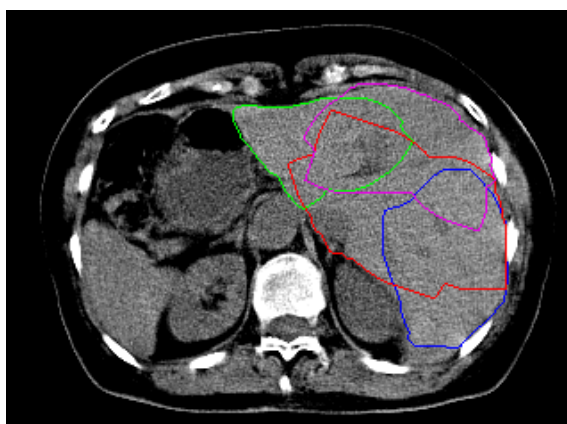
(a)



(b) A1 Energy = 5375392212



(c) A2 Energy = 5468661120



(d) A3 Energy = 4654115540



(e)

Figure 3.9: Experiments results for subject 6. (a) depicts the seeds where the teal circles are background seeds and the lines correspond to the separate foreground seeds. (b) shows the results of  $\alpha$ -expansion [A1], (c) shows the results of  $\alpha$ -expansion using the hedgehog shape prior [A2], and (d) shows the results of our approach [A3]. Finally, (e) is the ground truth.

For quantitative comparison, we calculate the precision and recall for each segmentation which are reported in Table 3.1. We calculate these values in relation to a manually segmented ground truth. Precision calculates the ratio between the number of true positives versus the total number of positives, i.e

$$\text{Precision} = \frac{\text{true positive}}{\text{true positive} + \text{false positive.}}$$

Recall calculates the ratio between the number of true positives versus the total number of labeled pixels, i.e.

$$\text{Recall} = \frac{\text{true positive}}{\text{true positive} + \text{false negative.}}$$

Subject	Metric	classic segmentation		SOL
		A1 [4]	A2 [4, 18]	A3 (Ours)
Subject 1	Precision	0.690	0.935	<b>0.971</b>
	Recall	<b>0.989</b>	0.660	0.973
Subject 2	Precision	0.383	0.672	<b>0.929</b>
	Recall	<b>0.998</b>	0.844	0.941
Subject 3	Precision	0.438	0.806	<b>0.832</b>
	Recall	<b>0.999</b>	0.893	0.970
Subject 4	Precision	0.535	0.928	<b>0.975</b>
	Recall	<b>0.991</b>	0.863	0.977
Subject 5	Precision	0.647	0.728	<b>0.911</b>
	Recall	<b>0.991</b>	0.821	0.976
Subject 6	Precision	0.628	0.792	<b>0.916</b>
	Recall	<b>0.995</b>	0.981	0.965

Table 3.1: Precision and Recall values for the various subjects (subjects 1 - 6) and approaches (A1 - A3).

In Table 3.1, it can be seen that A1 consistently has the highest recall values while maintaining consistently low precision values. This is compatible with what can be seen in the figures where A1 over-segments the image such that much of the liver is properly labeled, but many portions of the scan are incorrectly labeled as liver. It should be noted that it is possible to fine-tune  $\lambda$  to acquire better results for A1 but we used the same  $\lambda$  for all our liver experiments.

While the segmentation for A2 improves the precision values, the recall values decrease. It can be seen that this is due to the local minima issues that were explained in Section 3.1, causing the liver to be under-segmented.

Finally, A3 not only has the highest precision values, but also maintains recall values which are comparable to A1. As with A2, A3 benefits from the shape prior, resulting in high precision values. However, as it allows overlaps between the foreground labels, it avoids weak local minima. For ease of comparison, Figure 3.10 displays a graphical representation of the precision vs. recall for each algorithm and shows how A3 consistently outperforms both A1 and A2.

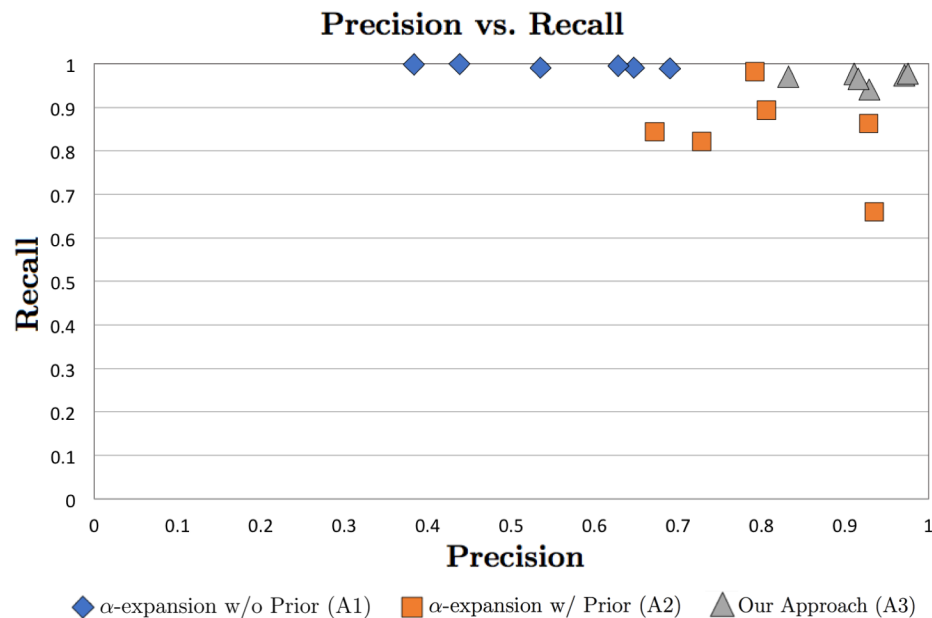


Figure 3.10: Graphical representation of precision vs. recall values for the three approaches (A1 - A3).

Lastly, Table 3.2 displays the  $F_1$  score. The  $F_1$  Score is a common metric to quantify the closeness of the segmentation to the ground truth. It combines the precision and recall values into a single value where an exact correspondence to the ground truth is 1. It is calculated as

$$F_1 \text{ Score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

	<b>Sub. 1</b>	<b>Sub. 2</b>	<b>Sub. 3</b>	<b>Sub. 4</b>	<b>Sub. 5</b>	<b>Sub. 6</b>
<b>A1</b> [4]	0.812	0.554	0.609	0.695	0.783	0.770
<b>A2</b> [4, 18]	0.774	0.749	0.847	0.894	0.772	0.877
<b>A3</b> (Ours)	<b>0.972</b>	<b>0.935</b>	<b>0.896</b>	<b>0.976</b>	<b>0.942</b>	<b>0.940</b>

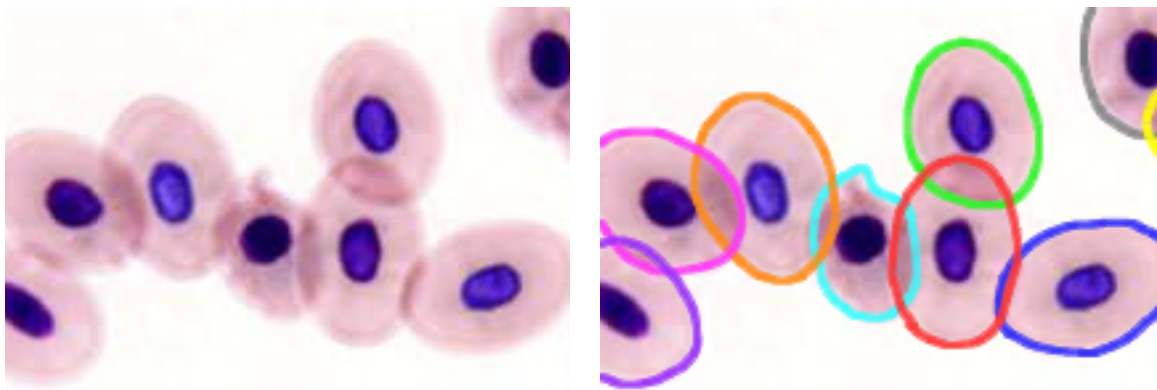
Table 3.2:  $F_1$  Score for the various subjects (subjects 1 - 6) and approaches (A1 - A3).

## **Chapter 4**

### **Future Work and Conclusion**

In this work, we have introduced the concept of segmentation with overlapping labels, a segmentation model that excels at segmenting translucent objects, occluded objects, and objects that require multiple seeds. Specifically, we have shown that SOL is an effective segmentation model for segmenting complex objects with multiple parts. Further, we have proved that SOL is an NP-hard problem and provided an iterative algorithm to find an approximate solution.

Our approach was inspired by the  $\alpha$ -expansion [4] and grab-cut algorithms [24]. However, our optimization framework utilizes the novel expansion and contraction moves. To test our algorithm, we ran several experiments on both synthetic images as well as medical images. The results of these experiments display the advantage of using our approach in comparison to the state-of-the-art methods.



(a) source image

(b) expected SOL segmentation

Figure 4.1: (a) depicts an image of cells that may benefit from SOL over mutually exclusive labels. (b) shows an expected SOL segmentation.

Cell segmentation is a possible application of SOL. The concept of overlapping labels is particularly useful in medical imaging where objects are translucent, see in Fig. 4.1. This property makes it difficult, if not impossible, for the classic segmentation model to properly segment each cell as a separate label.

Convexity [15] is an emerging prior and it has been recently extended to segment a non-convex object as a set of multiple convex parts. A possible future direction of our work would be to apply SOL to segment non-convex shapes consisting of multiple convex parts. Allowing the convex shape parts to overlap could potentially avoid local minima.

Lastly, our current application of SOL only takes into account one object with multiple parts. However, there are often times in which an image may have multiple complex objects, each composed of multiple parts. Another possible extension of SOL would be to handle such a problem. This would involve segmenting multiple foreground objects, each with its own set of overlapping labels.



# Bibliography

- [1] A. Berman, A. Dadourian, and P. Vlahos. Method for removing from an image the background surrounding a selected object, October 17 2000. US Patent 6,134,346.
- [2] Christopher M Bishop. Pattern recognition and machine learning (information science and statistics) springer-verlag new york. *Inc. Secaucus, NJ, USA*, 2006.
- [3] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE transactions on pattern analysis and machine intelligence*, 26(9):1124–1137, 2004.
- [4] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:1222–1239, November 2001.
- [5] Yuri Y Boykov and M-P Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 105–112. IEEE, 2001.
- [6] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.
- [7] D. Cremers, T. Pock, K. Kolev, and A. Chambolle. Convex relaxation techniques for segmentation, stereo and multiview reconstruction. In *Markov Random Fields for Vision and Image Processing*. MIT Press, 2011.
- [8] Andrew Delong and Yuri Boykov. Globally optimal segmentation of multi-region objects. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 285–292. IEEE, 2009.

- [9] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.
- [10] Lester R Ford and Delbert R Fulkerson. Maximal flow through a network. *Canadian journal of Mathematics*, 8(3):399–404, 1956.
- [11] Lester Randolph Ford Jr and Delbert Ray Fulkerson. *Flows in networks*. Princeton university press, 2015.
- [12] Andrew V Goldberg, Sagi Hed, Haim Kaplan, Pushmeet Kohli, Robert E Tarjan, and Renato F Werneck. Faster and more dynamic maximum flow by incremental breadth-first search. In *Algorithms-ESA 2015*, pages 619–630. Springer, 2015.
- [13] Andrew V Goldberg and Robert E Tarjan. A new approach to the maximum-flow problem. *Journal of the ACM (JACM)*, 35(4):921–940, 1988.
- [14] Lena Gorelick, Olga Veksler, Yuri Boykov, and Claudia Nieuwenhuis. *Convexity Shape Prior for Segmentation*, pages 675–690. Springer International Publishing, Cham, 2014.
- [15] Lena Gorelick, Olga Veksler, Yuri Boykov, and Claudia Nieuwenhuis. Convexity shape prior for binary segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(2):258–271, 2017.
- [16] Hossam Isack, Yuri Boykov, and Olga Veksler. A-expansion for multiple” hedgehog” shapes. Technical report, 2016.
- [17] Hossam Isack, Olga Veksler, Ipek Oguz, Milan Sonka, and Yuri Boykov. Efficient optimization for hierarchically-structured interacting segments (HINTS). In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [18] Hossam Isack, Olga Veksler, Milan Sonka, and Yuri Boykov. Hedgehog shape priors for multi-object segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.
- [19] Hiroshi Ishikawa. Exact optimization for markov random fields with convex priors. *IEEE transactions on pattern analysis and machine intelligence*, 25(10):1333–1336, 2003.

- [20] Jesmin F Khan, Sharif MA Bhuiyan, and Reza R Adhami. Image segmentation and shape analysis for road-sign detection. *IEEE Transactions on Intelligent Transportation Systems*, 12(1):83–96, 2011.
- [21] Vladimir Kolmogorov and Ramin Zabini. What energy functions can be minimized via graph cuts? *IEEE transactions on pattern analysis and machine intelligence*, 26(2):147–159, 2004.
- [22] Jan Lellmann, Jörg Kappes, Jing Yuan, Florian Becker, and Christoph Schnörr. *Convex Multi-class Image Labeling by Simplex-Constrained Total Variation*, pages 150–162. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
- [23] Stan Z Li. *Markov random field modeling in image analysis*. Springer Science & Business Media, 2009.
- [24] 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.
- [25] Olga Veksler. Star shape prior for graph-cut image segmentation. *Computer Vision—ECCV 2008*, pages 454–467, 2008.
- [26] Olga Veksler. Multi-label moves for mrf with truncated convex priors. *International journal of computer vision*, 98(1):1–14, 2012.

# Curriculum Vitae

**Name:** Karin Ng

**Post-Secondary Education and Degrees:** University of Western Ontario  
London, ON  
2016 - Present M.Sc.

The University of Toronto  
Toronto, ON  
2010 - 2015 B.Sc.

**Honours and Awards:** Faculty of Science Graduate Student Teaching Award,  
The University of Western Ontario, 2017

The Regents In-Course Scholarship,  
University of Toronto, 2014-2015

**Related Work Experience:** Teaching Assistant & Research Assistant,  
The University of Western Ontario  
2016 - 2017