9-15-2011 12:00 AM

Advances in Graph-Cut Optimization: Multi-Surface Models, Label Costs, and Hierarchical Costs

Andrew T. Delong
University of Western Ontario

Supervisor
Yuri Boykov
The University of Western Ontario

Graduate Program in Computer Science
A thesis submitted in partial fulfillment of the requirements for the degree in Doctor of Philosophy
© Andrew T. Delong 2011

ADVANCES IN GRAPH-CUT OPTIMIZATION: MULTI-SURFACE MODELS, LABEL COSTS, AND HIERARCHICAL COSTS

(Spine title: Advances in Graph-Cut Optimization)
(Thesis format: Monograph)

by

Andrew Delong

Graduate Program in Computer Science

A thesis submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

The School of Graduate and Postdoctoral Studies
The University of Western Ontario
London, Ontario, Canada

THE UNIVERSITY OF WESTERN ONTARIO
School of Graduate and Postdoctoral Studies

CERTIFICATE OF EXAMINATION

Supervisor:

. . . .
Dr. Yuri Boykov

Examiners:

. . . .
Dr. Roberto Solis-Oba

. . . .
Dr. Éric Schost

. . . .
Dr. Hristo S. Sendov

. . . .
Dr. Brendan J. Frey

The thesis by

Andrew Thomas Delong

entitled:

Advances in Graph-Cut Optimization: Multi-Surface Models, Label Costs, and Hierarchical Costs

is accepted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

. . . .
Date

. . . .
Chair of Thesis Examination Board

Abstract

Computer vision is full of problems that are elegantly expressed in terms of mathematical optimization, or *energy minimization*. This is particularly true of “low-level” inference problems such as cleaning up noisy signals, clustering and classifying data, or estimating 3D points from images. Energies let us state each problem as a clear, precise objective function. Minimizing the correct energy would, hypothetically, yield a good solution to the corresponding problem. Unfortunately, even for low-level problems we are confronted by energies that are computationally hard—often NP-hard—to minimize. As a consequence, a rather large portion of computer vision research is dedicated to proposing better energies and better *algorithms* for energies. This dissertation presents work along the same line, specifically new energies and algorithms based on *graph cuts*.

We present three distinct contributions. First we consider biomedical segmentation where the object of interest comprises multiple distinct regions of uncertain shape (*e.g.* blood vessels, airways, bone tissue). We show that this common yet difficult scenario can be modeled as an energy over multiple interacting surfaces, and can be globally optimized by a single graph cut. Second, we introduce multi-label energies with *label costs* and provide algorithms to minimize them. We show how label costs are useful for clustering and robust estimation problems in vision. Third, we characterize a class of energies with *hierarchical costs* and propose a novel *hierarchical fusion* algorithm with improved approximation guarantees. Hierarchical costs are natural for modeling an array of difficult problems, *e.g.* segmentation with hierarchical context, simultaneous estimation of motions and homographies, or detecting hierarchies of patterns.

Keywords: Energy minimization, graph cuts, discrete optimization, metric labeling, minimum description length, segmentation, biomedical imaging, robust estimation, multi-view reconstruction.

Co-Authorship Statement

Chapters 1 and 2 are my own original work in summarizing the relevant background.

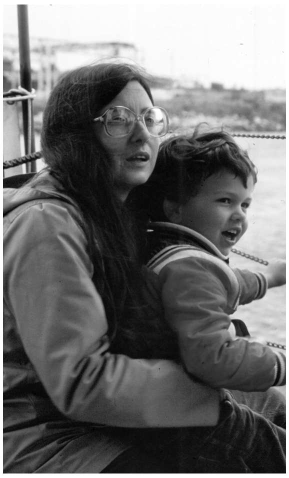
Chapter 3 was a collaborative effort with my advisor, Yuri Boykov. He recognized that the high-level ideas of Li *et al.* [102] could be applied in a more straight-forward manner. We then developed the technical ideas together. I went on to create the examples, the code, some extensions, and ultimately wrote the bulk of our paper [36].

Chapter 4 is about "label costs" and was a close collaboration with Anton Osokin, Hossam Isack, and Yuri Boykov. The homography detection and motion estimation applications were based on earlier work by Hossam Isack and Yuri Boykov on energy-based methods for multi-model fitting [68, 70]. I proposed the main technical idea that allows "label costs" and "label subset costs" to be optimized by the α-expansion algorithm. I then researched the related work on facility location and relevant high-order potentials; I am primary author of all related text. I programmed the C++/MATLAB library on which our work was based, starting from Olga Veksler's public α-expansion code. I collaborated with Anton Osokin on the approximation bound, and I designed the worst-case examples. Hossam Isack performed all the motion estimation and homography detection experiments specifically for this project; he and Yuri Boykov are primary authors of Section 4.7. Anton Osokin and Yuri Boykov are the primary authors of Sections 4.5.1 and 4.6.2—I merely helped to refine the presentation.

Chapter 5 is about energies with "hierarchical costs" and was developed for a pattern recognition project with Lena Gorelick, Olga Veksler, and Yuri Boykov. I proposed the *hierarchical fusion* algorithm, gave a formal characterizations of "hierarchical costs," and developed all the proofs as they appear here. Theorem 5.15 uses ideas from a proof that Olga Veksler developed for a special case. I worked with Lena Gorelick on a prototype implementation of hierarchical fusion. Though we performed some experiments using this approach, they are part of a very complex application-oriented work [59] that is not part of this dissertation.

Dedication

To my loving parents,
who took a chance all those years ago.



And in memory of Kyle.

Acknowledgements

I owe so much to my advisor, Yuri Boykov, and to my informal co-advisor, Olga Veksler. Yuri has had an extraordinary (extraordinarious!) influence on my interests, my work ethic, my sense of humour, and my opportunities in life. I'm lucky to have an advisor who works out of genuine curiosity and a passion for research, and who knows how to have a measure of fun in doing it. It is hard to find words to thank Olga for all the ways she has helped me, both remembered and forgotten. I will never forget the time and genuine care that Yuri and Olga invested in me as mentors, and as friends. I think, through them, I gained a better sense for what is important, and what is not. Wherever I go next it will not be the same. Спасибо.

I want to thank the many friends I made in the UWO vision group these last few years. I remember how surprised and happy I was to find out that Lena Gorelick was joining our group. Working with her has been great for me—lots of cool ideas bouncing around—and having fun with Lena and her husband Shachar has been even better. The last couple of years I've also enjoyed talking to Frank Schmidt each day, and learning how to express my complex feelings for Sarah Palin: a mix of pity, attraction, and Schadenfreude (Shaden = damage, Freude = joy). I particularly miss the time when Victor Lempitsky was visiting our group, and I am very lucky to have the hospitality and friendship that he and his wife (Victor's-Olga) have shown me. I likewise miss the time that Olivier Juan was here, and wish he and his wife Charlotte the best in their new life as parents. Other fun and interesting visitors I've gotten to know include Carl Olsson and Anton Osokin. It was also a great pleasure to know June (have fun at Google!) and Hossam (have fun with Yuri!). Good luck to the Iranian club, Paria, Taha and Vida. Thanks also Mark Brophy for rare indulgences into politics and philosophy.

I also enjoyed the warm hospitality of Daniel Cremers, his wife Lena, and their adorable daughter while I stayed in Germany—thank you. I'd like to thank Jing Yuan and his family for their hospitality as well.

When first I arrived in London it was not the best of times for me, until I met great friends like Mike & Jasna, Santoni-san, Dan Siemon, Micah, Gaston & Alex, Franzi & Carsten, Ryan looks-hot-as-a-woman Demopoulos, Beth, Nathan, Angela, Shayne, Freddie and many others along the way. We had lots of fun, especially in 240, and it's been bitter sweet to see everyone graduate and move on or get buried under their work.

I appreciate the advice and hard work of my thesis committee, professors Brendan Frey, Hristo Sendov, Éric Schost, and especially Roberto Solis-Oba who was meticulous and served on both my PhD and MSc committees. Also thanks to Cheryl for helping me out all those years and to Janice for laughing at my self-conscious jokes.

Last I want to thank my parents for being honest, hard-working, curious people. They make having integrity seem effortless. I've never figured out how they do that.

Graduate school has been a wonderful time, and a difficult time, in my life—but I would not trade these years for anything. I have been so fortunate, to have know all these persons, and to have had such freedom. I'll try to hold on to that thought for a while.

Contents

List of Figures

List of Algorithms

List of Tables

Chapter 1

Energy Minimization in Vision

Broadly speaking, this dissertation is about *energy minimization* in computer vision. In computer vision an *energy* is simply a mathematical objective function that we wish to extremize. For example, the energy $E(x) = (x - 5)^2 + (x - 3)^2$ has a minimum value of 2 at $x^* = 4$. The specific use of the word ‘energy’ suggests an objective function that has its origins in statistical physics—typically an unconstrained objective function where variables ‘interact’—but this connotation is not essential to our work. Rather, the important thing to understand is that a huge number of problems in vision are *inference problems* where the most likely explanation for the data can be found by minimizing a corresponding energy. For example, if we assume 5 and 3 are samples from a normal distribution, then the x^* that minimizes $E(x)$ is a maximum-likelihood estimate of the distribution’s *mean* parameter.

Of course, the inference problems in vision tend to be very complex and involve hundreds or even millions of inter-dependent variables. Some energies precisely model the desired inference problem, while others are merely a coarse approximation. Some energies are easy to optimize (*e.g.* convex functions) while others are known to be NP-hard. Once an accurate energy and a satisfying algorithm are both available, the associated inference problem is essentially solved. Researchers can then either improve the model or move on to other, more difficult problems.

Many of the most important developments in computer vision began with a proposal for a better energy, a better algorithm for an energy, or a combination of both. Good examples are [110, 132, 24, 83]. This dissertation is a small contribution in the same vein: we describe new energies that have useful interpretations, along with algorithms that are both effective in theory and fast in practice. We specifically focus on *discrete labeling* problems of the kind described in the following section.

1.1 Labeling Problems

A labeling problem is, roughly speaking, the task of assigning an explanatory ‘label’ to each element in a set of observations. Many classical clustering problems are also labeling problems because each data point is assigned a cluster label. To describe a labeling problem one needs a set of observations (the data) and a set of possible explanations (the labels). A discrete

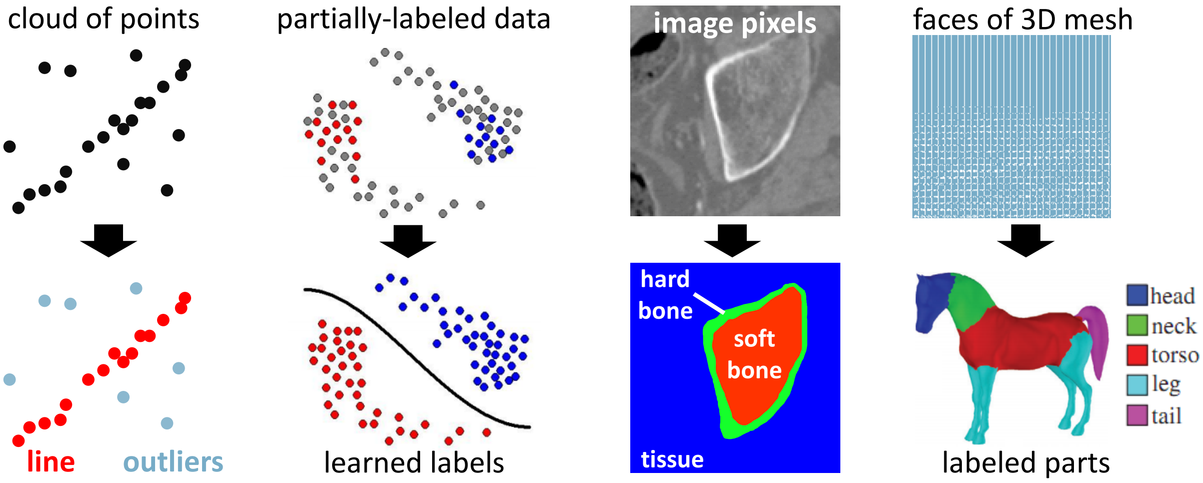


Figure 1.1: Example labeling problems. Given some input data, the goal is to assign an explanatory label to each input element. For example, if the data are 2D points then we may wish to classify them according to geometric models (points belonging to the same line). If the data comes partially-labeled, we can infer the remaining labels as in semi-supervised learning. (yinyang from [42], horse from [76].)

labeling problem associates one discrete variable with each datum, and the goal is to find the best overall assignment to these variables (a 'labeling') according to some criteria. In computer vision, the observations can be things like pixels in an image, salient points within an image, depth measurements from a range-scanner, or intensity measurements from CT/MRI. The labels are typically either semantic (*car*, *pedestrian*, *street*) or related to scene geometry (depth, orientation, shape, texture). Figure 1.1 depicts a few such possibilities.

We use the following notation for labeling problems throughout the dissertation. The set $\mathcal{P}$ indexes the observations, and the label set $\mathcal{L}$ indexes the explanations. The set of discrete variables is $\{f_p\}_{p \in \mathcal{P}}$ where each f_p is allowed to take one value from the set $\mathcal{L}$. A discrete *labeling* is the complete map $f : \mathcal{P} \to \mathcal{L}$ that assigns to each element $p \in \mathcal{P}$ a corresponding label f_p. For example, if $\mathcal{P} = \{p, q\}$ and $\mathcal{L} = \{\ell_1, \ell_2, \ell_3\}$, then labeling $f = (\ell_3, \ell_1)$ says that $f_p = \ell_3$ and $f_q = \ell_1$. If we instead let $\mathcal{P}$ index the pixels of a 100×100 image and there are two labels $\mathcal{L} = \{object, background\}$, then this is a standard "binary segmentation" scenario with $2^{10{,}000}$ possible labelings. In general we have $|\mathcal{L}|^{|\mathcal{P}|}$ possible labelings (configurations of f), and we prefer one labeling over another based on some application-specific criteria.

Data-driven criteria In computer vision we try to make sense of the input data. This means that every labeling problem must be formulated so that the data influences the outcome. For example, if our 100×100 image is an X-ray and a particular pixel $p \in \mathcal{P}$ is brightly coloured, then our labeling problem should prefer a labeling with $f_p = bone$ over one with $f_p = tissue$. If the image were of an outdoor scene instead, then we would expect blue pixels to prefer labels like *sky* or *water* and green pixels to prefer labels like *grass* or *leaves*. This is the most rudimentary kind of data-driven criterion possible, where each discrete variable f_p derives its preferences based solely on the data at observation p. By now it is common to derive these preferences from machine-learning techniques, but the output is fundamentally the same.

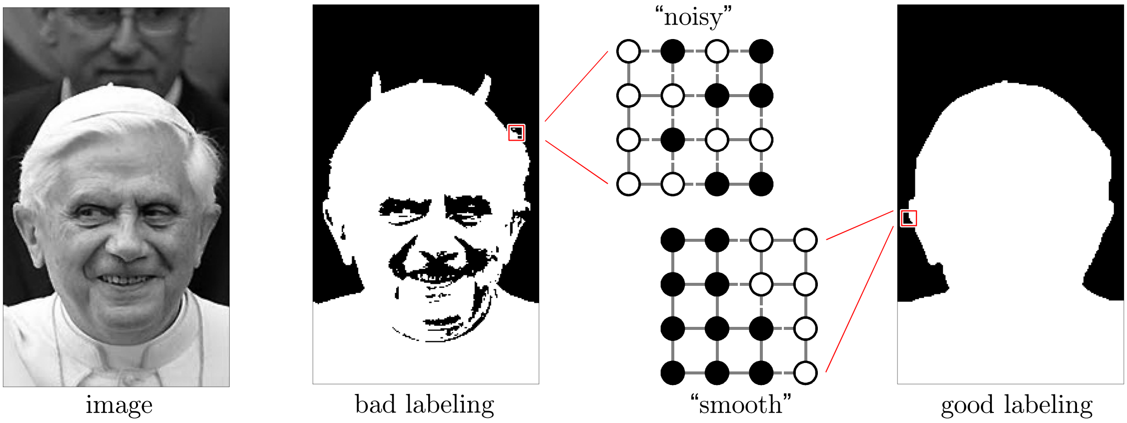
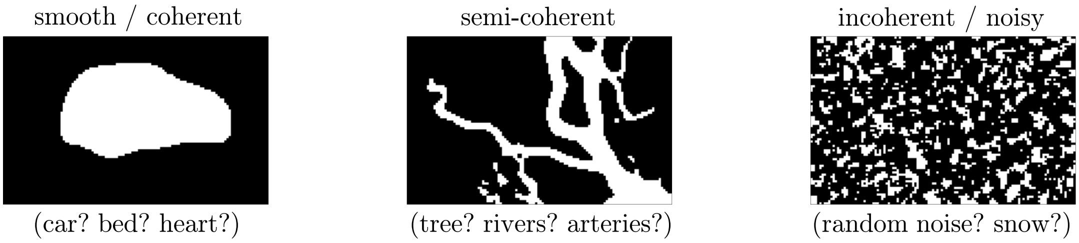


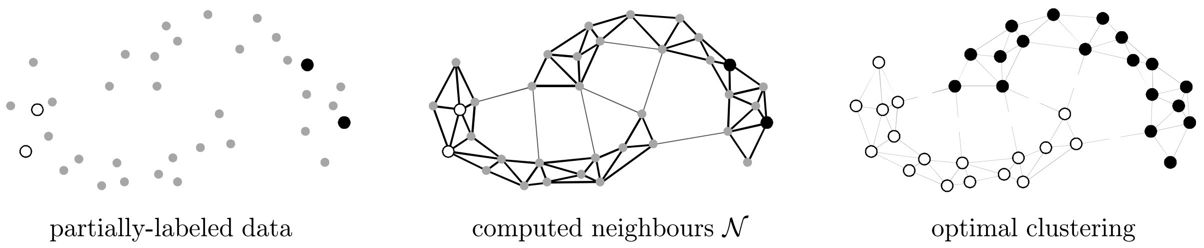
Figure 1.2: Suppose we want to isolate a bright object on a dark background. The simplest approach is to choose a labeling based on data-driven pixel preferences, *i.e.* bright pixels choose object, dark pixels choose background. However, the resulting contour will often be complex and noisy (left). When we know *a priori* that the object's shape should have a smooth, 'blobby' contour (cars, people, buildings) then we should prefer a labeling that satisfies this assumption (right). Each pixel in a smooth labeling is highly correlated with its nearest neighbours in the spatial grid, *e.g.* the magnified regions contain 15 transitions versus only 5 transitions.

Regularization criteria Some labelings are more likely to be correct *a priori*. When we explicitly prefer some kinds of labelings over others, irrespective of the data, these criteria are called *regularizers*. The most prominent example in computer vision is a preference for *spatially coherent* labelings. The idea is that, for most computer vision problems, coherent ("smooth") labelings are much more likely to be a correct explanation of the data than are incoherent ("noisy") labelings. For example, consider the binary image labelings below.



We know from experience that objects in photographs and in medical data correspond to coherent labelings more often than not; the truth of this claim varies from application to application, but in computer vision it has become a rule of thumb. It holds true because data in computer vision tends to be highly correlated in space. For example if $p, q \in \mathcal{P}$ are adjacent pixels in an X-ray image then one can expect $f_p = bone \Leftrightarrow f_q = bone$ with high probability, regardless of the data. The same cannot be said if p and q are very far apart in the image, because such pixels are not directly correlated in practice. It turns out that such *a priori* assumptions, or *priors*, are very important—often crucial—in many vision applications [54, 103].

Figure 1.2 shows a real image and two possible binary segmentations. The noisy labeling is based on individual pixel preferences, whereas in the smooth labeling some pixels sacrificed their individual preference so that the *spatial coherence* criterion is better satisfied. The



partially-labeled data

computed neighbours $\mathcal{N}$

optimal clustering

Figure 1.3: Given a set of points, some of which are labeled, *semi-supervised learning* asks us to choose the most probable label for each unlabeled point. Above we see 4 pre-labeled points (2 *white*, 2 *black*). To cluster the points, Blum & Chawla [17] compute a neighbour graph (*e.g.* nearest-neighbour, Delaunay triangulation) and then find a labeling that is smooth with respect to that connectivity. In other words, constrained clustering problems can be solved using the same criterion as for image segmentation [42].

close-up of the pixel grid also suggests a way to make the notion of smoothness more precise: smoothness implies fewer label transitions between neighbouring variables. This characterization of smoothness is standard in computer vision and we formalize it in the next section. Figure 1.3 explains how this same smoothness criterion can be used in semi-supervised learning [17, 140, 167]—an important labeling problem that, on the surface, seems entirely different from segmentation [42].

1.2 Labeling Problems as Energy Minimization

We now express some standard *data-driven* and *regularization* criteria as concrete *energy terms*. An energy term is an expression, dependent on labeling f, that is added linearly in the energy. Breaking an energy into terms means expressing it in the form

$$E(f) = \text{term1}(f) + \text{term2}(f) + \dots$$

Each energy term basically votes on how much it likes labeling f or some subset of its variables. If a particular term evaluates to a small numerical value, then this means f reasonably satisfies the corresponding criterion. Minimizing $E(f)$ thus finds a compromise among all the labeling criteria in the energy.

For example, suppose we have $\mathcal{P} = \{p, q, r\}$ and two possible labels $\mathcal{L} = \{\ell_1, \ell_2\}$. If we say $D_p(\ell)$ is the cost of assigning $f_p = \ell$ based on the data, then we insert expression $D_p(f_p)$ as a term in the energy. Assume our energy contains only the data terms shown below, where the table gives individual assignment costs based on the data.

$$E(f) \; = D_p(f_p) + D_q(f_q) + D_r(f_r)$$

data costs ↘	p	q	r
ℓ_1	0	6	0
ℓ_2	10	5	5

The minimum value for this binary energy is achieved at $f^* = (\ell_1, \ell_2, \ell_1)$ with $E(f^*) = 5$. Clearly such an energy is trivial to minimize in $\Theta(|\mathcal{P}||\mathcal{L}|)$ time since each term $D_p(f_p)$

can be minimized independently. (Minimizing such an energy is equivalent to "thresholding" techniques from the early days of image processing.)

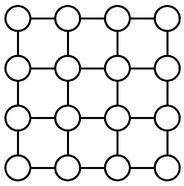
Now suppose we wish to incorporate the prior knowledge that some of the observations are directly correlated with each other; specifically we want p correlated with q and q correlated with r. We can incorporate this 'coupling' of variables by adding energy terms that explicitly encourage $f_p = f_q$ and $f_q = f_r$. We refer to these as *smooth terms* and denote them by V.

$$E(f) \;= D_p(f_p) + D_q(f_q) + D_r(f_r) + V(f_p, f_q) + V(f_q, f_r)$$

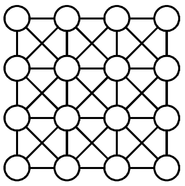
In the simplest case we use the delta function $V(\ell, \ell') = \delta(\ell \neq \ell')$ where δ is 1 if its condition is true, and 0 otherwise. The old optimum of $f = (\ell_1, \ell_2, \ell_1)$ now evaluates to $E(f) = 7$, whereas the new optimum $f^* = (\ell_1, \ell_1, \ell_1)$ evaluates to $E(f^*) = 6$. The smooth terms encouraged the labeling to be *coherent* (consistent) so that fluctuations caused by noisy data costs are smoothed out. Note that the the optimal label assignments can no longer be solved independently, and must somehow be minimized *jointly*. It is not entirely obvious how to minimize such energies in general. For problems with thousands of inter-dependent variables we will need fast, specialized algorithms to compute f^*, or at least an approximation thereof.

Let $\mathcal{N}$ denote the pairs of observations we know *a priori* to be correlated, for example in our 3-variable problem we used $\mathcal{N} = \{\{p, q\}, \{q, r\}\}$. We refer to $\mathcal{N}$ as the *neighbour set* throughout. Common neighbour sets in vision are depicted as graph edges below.

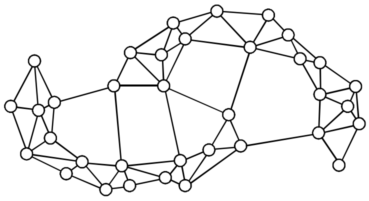
4-connected grid



8-connected grid



non-uniform, distance-based



Further assume that each unordered pair $pq \in \mathcal{N}$ interacts using its own smooth term where, for example, V_{pq} might be a different strength than V_{qr}. We can write this class of energies as

$$E(f) \;= \sum_{p \in \mathcal{P}} D_p(f_p) \;+ \sum_{pq \in \mathcal{N}} V_{pq}(f_p, f_q). \tag{1.1}$$

In the last decades, energies of the form (1.1) have proven indispensable for many problems in computer vision. They were first proposed for inference problems associated with *Markov random fields* (MRFs), a powerful class of statistical models in physics and pattern recognition [54, 103]. Despite the modeling power of MRFs, they saw limited use in computer vision until practical algorithms were finally introduced much later [23, 24].

As of this writing, energy (1.1) is the starting point for dozens of specialized formulations in vision [159, 80], machine learning [17], and even bioinformatics [161, 149, 131]. Minimizing (1.1) is known as the *MAP-MRF problem* [103, 158], owing to its statistical interpretation and ubiquity. This dissertation proposes useful generalizations of (1.1). For example, Chapter 4 introduces energies of the form

$$E(f) \;= \sum_{p \in \mathcal{P}} D_p(f_p) \;+ \sum_{pq \in \mathcal{N}} V_{pq}(f_p, f_q) \;+ |\mathcal{L}(f)| \tag{1.2}$$

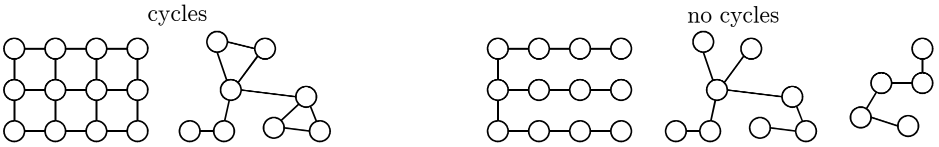
where $\mathcal{L}(f) \subseteq \mathcal{L}$ is the set of labels used by f. That is, we penalize the number of unique labels appearing in the solution. We also provide effective algorithms to minimize a more general class of energies, of which (1.2) is a special case.

1.3 Energy Minimization: Algorithms and Special Cases

Some energy minimization problems can be solved in polynomial-time, whereas many are known to be NP-complete or NP-hard and must be approximated (at best). Energy (1.1) is NP-hard to minimize in general [24] but there are many special cases to consider, some permitting specialized algorithms that run in polynomial time. We noted that energies of the form $E(f) = \sum_{p \in \mathcal{P}} D_p(f_p)$ are trivial to minimize in $\Theta(|\mathcal{P}||\mathcal{L}|)$ time. As one considers energies of increasing generality, the array of corresponding algorithms grows more diverse and sophisticated. Chapter 2 reviews algorithms relevant to this dissertation, but it helps to understand the situation more broadly. Here we give a high-level overview of important special cases of (1.1), their difficulty, and some applicable minimization techniques. There are two main factors to consider: *structural* restrictions (special neighbour sets $\mathcal{N}$), or *functional* restrictions (special cost functions D_p and V_{pq}). We review the some basic and well-known examples of each kind.

1.3.1 Tree-structured neighbour sets

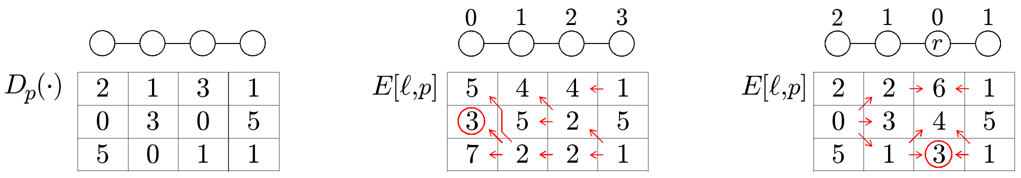
A simple but useful structural restriction is a neighbour set $\mathcal{N}$ that defines an acyclic graph, *i.e.* $\mathcal{N}$ defines a chain, a tree, or a forest structure.



Any energy of the form (1.1) with no cycles can be minimized in $\Theta(|\mathcal{P}||\mathcal{L}|^2)$ time via *dynamic programming* (DP) [12, 28] or, equivalently, message-passing algorithms [115, 162]. A 4-connected grid graph clearly has cycles, so image segmentation does not fall within this special case, but there are many tree-structured inference problems in computer vision [53, 44, 150] and in inference more broadly such as *hidden Markov models* (HMMs) [120] and *graphical models* [74].

The reason dynamic programming works in this case is because we can express a minimum of $E(f)$ recursively so as to take advantage of overlapping subproblems. The simplest case is a *chain*, where we can order the variables as $f_0, \ldots, f_n$ so that $\mathcal{N} = \{(p-1, p)\}_{p=1}^{n}$. Now, consider those terms of E involving only variables $f_p, \ldots, f_n$; we let $E[p, \ell]$ be the minimal possible sum of those terms when we force $f_p = \ell$. Clearly $E[\ell, n] = D_n(\ell)$ and, because $\mathcal{N}$ forms a chain, for any $p < n$ the value $E[p, \ell]$ can be expressed recursively as

$$E[p, \ell] \;=\; D_p(\ell) + \min_{\ell'} \Big(\, V_{p,p+1}(\ell, \ell') + E[p+1, \ell'] \, \Big). \tag{1.3}$$



$D_p(\cdot)$

2	1	3	1
0	3	0	5
5	0	1	1

data cost table

$E[\ell,p]$

0	1	2	3
5	4	4	1
3	5	2	5
7	2	2	1

DP on a *chain*

$E[\ell,p]$

2	1	0 r	1
2	2	6	1
0	3	4	5
5	1	3	1

DP on *tree* rooted at r

Figure 1.4: A 3-label energy of the form (1.1). $D_p(\cdot)$ is defined by the table at left, and $V(\ell, \ell')$ penalizes any $\ell \neq \ell'$ by cost 1. If we order the four variables from left to right, dynamic programming (DP) will compute columns $E[\cdot, p]$ from right-to-left (center). For arbitrary root r DP will compute, for example, columns $E[\cdot, p]$ at right. A red arrow $[q, \ell'] \to [p, \ell]$ indicates that $E[q, \ell']$ was directly used to compute $E[p, \ell]$ within (1.3) and (1.4).

The optimal energy is then $E(f^*) = \min_\ell E[0, \ell]$ and can be found by tabulating $E[p, \ell]$, starting at $p = n$ and applying (1.3) to work backwards. Figure 1.4 (center) shows a numerical example of dynamic programming on a chain-structured neighbour set $\mathcal{N}$.

Dynamic programming extends easily from chains to trees. Simply designate an arbitrary node to be the *root* $r \in \mathcal{P}$ and, for each node $p \in \mathcal{P}$, let $\mathcal{I}(p)$ denote its children with respect to that rooted tree. We now let $E[p, \ell]$ be the minimal possible sum of all energy terms involving only descendants of p when we force $f_p = \ell$. We can then write $E[p, \ell]$ more generally as

$$E[p, \ell] \;=\; D_p(\ell) + \sum_{q \in \mathcal{I}(p)} \Big(\min_{\ell'} V_{p,q}(\ell, \ell') + E[q, \ell'] \Big). \tag{1.4}$$

The optimal energy is now $E(f^*) = \min_\ell E[r, \ell]$ and can be found by tabulating $E[p, \ell]$, starting at the leaves of the tree and applying (1.4) as needed to work from the 'furthest' nodes inwards to the root. Figure 1.4 (right) repeats our numerical example for what is essentially a tree (root r has degree > 1). Notice that in both cases the optimal value is $E(f^*) = 3$ and we can recover labeling $f^* = (\ell_2, \ell_3, \ell_3, \ell_3)$ by simply remembering the red arrows when computing each $E[p, \ell]$ and tracing back our steps.

A cycle in the neighbour set means that such recursive expressions cannot work—a cycle creates a dependency that cannot be 'unwrapped' symbolically. Recently it was shown that neighbour sets defining *outer-planar graphs* can also be minimized efficiently [11]. Outer-planar graphs include trees as a special case, but they are still very far from general graphs; for example, the 4-connected grid is a basic planar graph used in many vision problems, but it is not outer-planar.

1.3.2 Binary energies with coherence

We describe a special case of energy (1.1) that has, in the last decade, exploded in popularity. If the energy is binary ($|\mathcal{L}| = 2$), *and* the smoothness terms encourage coherence (as opposed to explicitly discouraging it), then the energy can be minimized in polynomial time. We can see what this means, visually, by contrasting the characteristics of the two labelings below.

can explicitly encourage this

cannot explicitly encourage this

In other words, if we want to solve the problem in polynomial time, we can encourage smooth labelings, we can be indifferent to smoothness ($\mathcal{N} = \{\}$), but we cannot actively *discourage* smoothness. In terms of our energy (1.1), encouraging coherence means each $V_{pq}(f_p, f_q)$ term should assign lower cost to configurations with $f_p = f_q$ than to configurations with $f_p \neq f_q$. Encouraging *incoherent* labelings, *i.e.* preferring $f_p \neq f_q$, makes minimization NP-hard! The precise mathematical property that makes the former problems tractable is called *submodularity*, and is reviewed in Chapter 2.

This special case was first studied in combinatorial optimization [116, 32], in image restoration [62], and was finally popularized in computer vision through the early works of Boykov *et al.* [19, 20, 21, 22]. Note that cycles in $\mathcal{N}$ *are* permitted in this special case. By constraining V_{pq} we gain flexibility in $\mathcal{N}$ while still minimizing the energy in polynomial time. So how is this minimization carried out in practice? We obviously cannot use dynamic programming, so what is the algorithm? It turns out that this class of binary energies can be reformulated as a standard *s-t minimum cut* problem for which efficient algorithms have long existed [47, 41, 58, 114] and are still being developed for problems in vision [22, 35, 135]. A wide array of energy-minimization techniques now use s-t min-cut as the core subproblem; such techniques are referred to as *graph cut* methods. The algorithms in this dissertation are all based on graph cuts, and Chapter 2 reviews the relevant prior art in some detail.

1.3.3 Table of special cases and algorithms

Minimizing energy (1.1), also called the *MAP-MRF problem*, is known to be NP-hard to solve exactly [133, 24]. In fact, without any assumptions at all, MAP-MRF cannot be meaningfully approximated in polynomial time [1, 73]. The problem remains NP-hard to approximate within a constant factor for all but the most severe assumptions, such as V being "metric" (Section 2.2.2) or that $|\mathcal{L}| \leq 3$; even in this case the problem remains max-SNP-hard [31]. Table 1.1 provides a high-level overview of a number of tractable special cases, as well as some approximation algorithms that can be applied in the general case. Note that exact optimization is not always necessary. In fact Szeliski *et al.* [139] observed that, for many applications, the energy value of the human-selected solution is often higher than the globally optimal solution to our energy! In other words, seeking a global optimum is not always worth the computational effort, especially if our energy does not accurately model the problem.

Note that Table 1.1 is about energy (1.1) only, and that many generalizations of this energy to "high-order terms" have been proposed, *e.g.* [159, 80, 38]. The known approximation ratios for minimizing such energies are typically worse than for the pairwise case [60] or in many cases not even understood.

Table 1.1: Some special cases of energy (1.1) that result in minimization problems of varying difficulty. This table is incomplete and is intended to give a sense for the kinds of special cases that may make minimization easier. Not all of the "approximate methods" are approximation algorithms in the strict sense, *e.g.* [115, 83, 85, 158] provide no *a priori* approximation bounds whatsoever.

	special case	**algorithms**	**notes / applications**
EXACT METHODS	$\mathcal{N}$ acyclic graph	dyn. programming [28, 44], message passing [115, 90, 162]	template matching [53, 44], HMMs [120], stereo [150]; decomposition methods[157, 83]
	V submodular, $\lvert\mathcal{L}\rvert$=2	graph cut [20, 87]	segmentation [20], machine learning [17], multi-view reconstruction [88, 154, 99], move-making algorithms[24]
	V convex	transform [71] + graph cut	
	V permuted submodular	transform [127] + graph cut	
	D convex, V convex	parametric graph cut [65, 86]	denoising / image restoration
	D special, $\mathcal{N}$ planar	planar min-cut [108, 128]	D single source/sink; shape matching, segmentation
	$D = 0$, $\mathcal{N}$=planar, $\lvert\mathcal{L}\rvert$=2	max-weight matching [129]	
	$\mathcal{N}$ perfect graph	transform [72, 48] + message passing	
	$\mathcal{N}$ outerplanar graph	junction tree [11]	decomposition methods[11]
APPROXIMATE METHODS	V metric	α-expansion [24] and extensions [4], LP rounding [78], r-HST metrics [92]	approximation bounds; stereo, segmentation, model-fitting [70]
	V semi-metric	$\alpha\beta$-swap [24], r-HST metrics [92]	approximation bound [92]
	V truncated convex	range moves [151, 93]	approximation bound
	$\lvert\mathcal{L}\rvert$=2	QPBO [18, 85], QPBO-I [124], bipartite multi-cut [122]	approximation bound [122] $\propto$ log(#non-submodular terms); QPBO gives partial labelings
	arbitrary energy	mess. passing [115, 56, 142], decomposition methods [83] dual decomposition [89, 11], max-sum diffusion [158], local search [75], ...	NP-hard to approximate by constant factor [133, 1]

1.4 Chapter Outlines

The remainder of this dissertation can be summarized as follows.

Chapter 2 reviews graph cut methods that are essential to the development of this dissertation. This includes the binary graph cut reduction, submodular functions, and the iterative move-making algorithms "α-expansion" and "$\alpha\beta$-swap." The review of graph cuts and submodularity is essential to this entire dissertation, and the iterative algorithms are the heart of Chapters 4 and 5.

Chapter 3 presents a segmentation technique based on a special "multi-surface" graph cut construction. Our binary construction induces a multi-label segmentation where the interfaces between regions (boundaries/surfaces) have preferred distances from one another. Specifically, our construction has the following properties:

1. regions can have nesting constraints (*e.g.* soft bone must be surrounded by hard bone),
2. surfaces can have preferred distances (*e.g.* hard bone should be $\geq$ 5mm thick), and
3. the globally optimal multi-region segmentation can be computed by a single graph cut.

The content is based directly on my joint publication with Yuri Boykov at the 2009 *International Conference on Computer Vision* (ICCV) [36].

Chapter 4 extends the classic MAP-MRF energy to include "label costs" as a regularizer. In their simplest form, label costs penalize the number of unique labels used to explain the observations. In other words, why use 6 labels to explain the data if 5 will do just as well. In general we define a new class of energies with *label subset costs* and extend the α-expansion and $\alpha\beta$-swap algorithms to handle this regularizer. We also characterize the effect on algorithm's optimality guarantees with a tight bound, and establish connections to related problems in operations research and in computer vision. This work was initially published in the 2010 *Conference on Computer Vision and Pattern Recognition* (CVPR) [38] and subsequently expanded in the *International Journal of Computer Vision* (IJCV) [39].

Chapter 5 defines a new characteristic of energies, as having *hierarchical costs*, and describes a novel *hierarchical fusion* algorithm to minimize such energies. This fusion algorithm is a strict generalization of α-expansion yet provides significantly tighter approximation bounds in many useful cases. Hierarchical costs are natural for modeling an array of difficult problems, *e.g.* segmentation with hierarchical context, simultaneous estimation of motions and homographies, or detecting hierarchies of patterns. This work was submitted to the *International Journal of Computer Vision* (IJCV) as [37].

Chapter 2

Review: Energies and Algorithms

This chapter reviews well-known concepts upon which subsequent chapters are developed. All contributions in this dissertation are based on *graph cuts* and on related move-making algorithms.

Section 2.1 explains the basic idea of reducing a binary energy minimization problem to that of computing an s-t min-cut [62, 20, 87]; this reduction is the starting point for Chapter 3. Section 2.2 explains the popular move-making algorithms "α-expansion" and "$\alpha\beta$-swap" for minimizing multi-label energies [24]. These algorithms find local minima of NP-hard energies by constructing a particular sequence of graph cut subproblems. These move-making algorithms are essential to Chapters 4 and 5.

2.1 Binary Energies Reducible to a Graph Cut

The special case of "binary energies with coherence" (Section 1.3.2) has proven extremely valuable in computer vision because

a) it is a good model for a wide variety of binary labeling problems,
b) it is a powerful subproblem for local search in labeling problems (Section 2.2), and
c) there exist fast minimization algorithms for large-scale problems.

The key insight that lets us solve problems efficiently is reducing the *binary energy minimization* problem to the well-known s-t *min-cut* problem. Furthermore, empirical tests have shown [22, 57] that the fastest method to compute the an s-t min-cut is to solve the dual s-t *maximum flow* problem using specialized algorithms. The s-t *min-cut* and s-t *max-flow* problems have long been studied in operations research, and it took many insights by different individuals before reduction from binary energy minimization became well-known in computer vision. This reduction combines early work on the duality between min-cut and max-flow [47, 130], work on submodular functions [32, 52, 87], pseudo-boolean functions [18], and MAP-MRF formulations in computer vision [62, 20]. Efficient s-t max-flow algorithms include [58, 22, 35, 135], but we will not discuss min-cut / max-flow duality in detail. Instead we explain s-t min-cut and assume it is sufficiently instructive. Readers interested in max-flow may consult [130] for discussion of how it relates to min-cut.

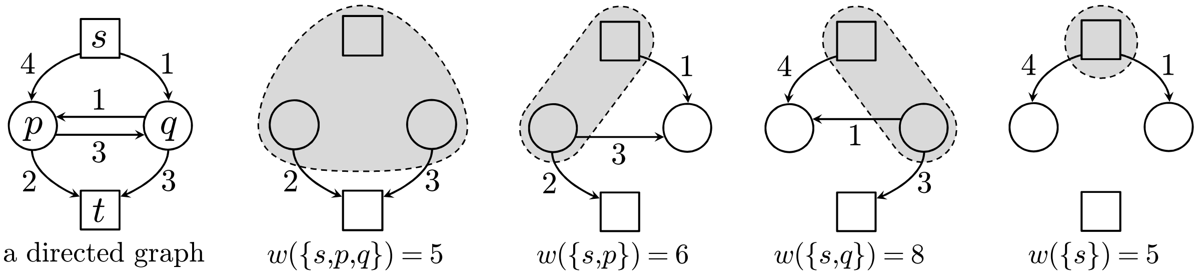


Figure 2.1: A simple s-t min-cut problem with six weighted arcs. There are four possible s-t cuts: $S = \{s,p,q\}, \{s,p\}, \{s,q\}$, or $\{s\}$. Since two of the cuts have minimal cost $w(S) = 5$ the optimum solution is not unique. General s-t min-cut problems can contain thousands or millions of vertices.

2.1.1 The s-t min-cut problem

To understand the *binary energy* $\rightarrow$ *s-t min-cut* reduction, one must first understand the basics of the s-t min-cut problem. Defining an instance of s-t min-cut is very simple. We require a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, a cost $w(u,v) \geq 0$ for arc each $(u,v) \in \mathcal{A}$, and two designated *terminals* $s, t \in \mathcal{V}$. The aim of s-t min-cut is to remove the cheapest subset of arcs so that there is no path from s to t in the graph. Rather than define a 'cut' directly in terms of arcs, the selected arcs are implied by definition based on vertices.

Definition 2.1. *A subset $S \subseteq \mathcal{V}$ such that $s \in S$ and $t \notin S$ is called an s-t* cut. *The cost $w(S)$ of an s-t cut S is defined as*

$$w(S) \overset{\text{def}}{=} \sum_{\substack{(u,v)\in\mathcal{A} \\ u\in S, v\notin S}} w(u,v)$$

In other words, the cost of an s-t cut S is the total cost of arcs leaving set S. Figure 2.1 shows a small min-cut problem instance.

The s-t min-cut problem is to find the s-t cut S^* of minimal total cost. An optimal cut can be computed in polynomial time by a number of classical s-t *maximum flow* algorithms [41, 58] but, in computer vision, more specialized algorithms are typically used, in particular the method of Boykov & Kolmogorov [22] and recent extensions, *e.g.* [82, 57]. The specialized algorithms are fast enough that, in practice, a non-negligible fraction of running time goes towards merely constructing the initial graph structure inside the computer.

2.1.2 Reduction of second-order energies

We now review how to transform the binary energy minimization problem into an s-t min-cut problem from Section 2.1.1. We will use the notation $\mathbf{x} = (x_1, \ldots, x_n)$ to denote an n-variable labeling for binary energy $E(\mathbf{x})$. Begin with a straight-forward observation.

Observation 2.2. *In a digraph with $\mathcal{V} = \{s, t, v_1, \ldots, v_n\}$ there are 2^n possible s-t cuts. There is thus a one-to-one correspondence between cuts and configurations of binary vector $\mathbf{x} \in \{0,1\}^n$.*

In this dissertation we arbitrarily define correspondence $v_i \in S \Leftrightarrow x_i = 0$. Let $S_{\mathbf{x}}$ denote the cut corresponding to binary vector $\mathbf{x}$. If we construct a digraph such that $w(S_{\mathbf{x}}) = E(\mathbf{x})$ for all configurations then we reduce minimizing $E(\mathbf{x})$ to the s-t min-cut problem.

Example 2.3. *Consider the binary energy function below with $\mathcal{P} = \{p, q\}$ and $\mathcal{L} = \{0, 1\}$.*

$$E(x_p, x_q) \;= D_p(x_q) + D_q(x_q) + V(x_p, x_q)$$

D	p	q
0	2	3
1	4	1

V	0	1
0	0	3
1	1	0

We can also define this energy by enumerating all its possible values

$$\begin{aligned} E(0,0) &= 2+3+0 = 5 \\ E(0,1) &= 2+1+3 = 6 \\ E(1,0) &= 4+3+1 = 8 \\ E(1,1) &= 4+1+0 = 5 \end{aligned}$$

Verify by inspection that minimizing $E(x_p, x_q)$ over $x_p, x_q \in \{0, 1\}$ is equivalent to the s-t min-cut problem shown in Figure 2.1.

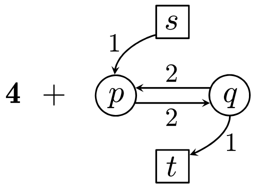
After looking at energy E from Example 2.3 it is informative to consider the following, equivalent binary energy:

$$E'(x_p, x_q) \;= \mathbf{4} + D_p(x_q) + D_q(x_q) + V(x_p, x_q)$$

D	p	q
0	0	1
1	1	0

V	0	1
0	0	2
1	2	0

Clearly $E(x_p, x_q) = E'(x_p, x_q)$ for all $x_p, x_q \in \{0, 1\}$. One can view E' as a *reparameterization* of energy E where 4 is an additive constant and therefore irrelevant to the minimization problem itself. We can alter the s-t min-cut problem from Figure 2.1 to correspond directly to the reparameterization E' using the graph below.



The examples so far involved only two variables p and q. However, since the energy terms are added linearly, by the additivity property of this reduction [87] we can reduce each energy term one-by-one, and simply superimpose all the arcs. This is particularly trivial for the standard MAP-MRF energy (1.1) because it is of the *second-order*, *i.e.* each individual $D_p(\cdot)$ and $V_{pq}(\cdot, \cdot)$ term is a function of at most two variables (of 'degree' at most two). A complete sequence of steps for reducing each D_p and V_{pq} term was given in [87].

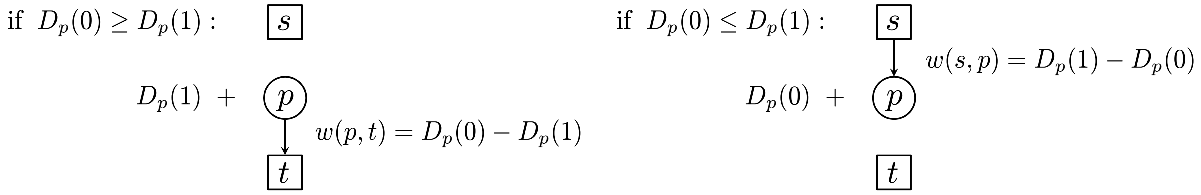
2.1.3 Which energies can be reduced to graph cut? (submodularity)

There is an important question regarding the *binary energy* → *s-t min-cut* reduction. The MAP-MRF energy (1.1) is NP-hard to minimize even in the binary case [133], and yet we know from example that reduction to s-t min-cut is sometimes feasible. Clearly there must be some property that distinguishes 'easy' binary energies from hard ones. It is thus natural to ask: *precisely which binary energy functions are reducible to a graph cut?*

In Section 1.3.2 we claimed, vaguely, that a binary energy must "encourage coherence" in order to be tractable. Notice in Example 2.3 that positive arc weights $w(p,q)$ and $w(q,p)$ encourage p and q to belong to the same side of the cut, *i.e.* configurations with $x_p = x_q$ are cheaper than configurations with $x_p \neq x_q$. If these arc weights were negative, they would have the *opposite* effect. However, in the s-t min-cut problem, the arc weights *cannot* be negative due to the assumption that $w(u,v) \geq 0$. If we were to omit this restriction from the definition of s-t min-cut, it would be NP-complete via reduction to/from the MAX-CUT problem!

Again, a binary energy is representable by s-t cuts if there exists a digraph with $w(u,v) \geq 0$ such that $w(S_{\mathbf{x}}) = E(\mathbf{x})$ for each $\mathbf{x}$. By the additivity property [87] we need only ask whether each individual $D_p(\cdot)$ term and individual $V_{pq}(\cdot,\cdot)$ term can be represented by weighted arcs.

Each $D_p(\cdot)$ is trivial to represent by breaking it into the two cases shown below:


if $D_p(0) \geq D_p(1)$:
s
$D_p(1)$ +
p
$w(p,t) = D_p(0) - D_p(1)$
t
if $D_p(0) \leq D_p(1)$:
s
$w(s,p) = D_p(1) - D_p(0)$
$D_p(0)$ +
p
t

Though $D_p(1)$ might be negative, if $D_p(0) \geq D_p(1)$ then we treat it as an additive constant and so the arc weight $w(p,t)$ is guaranteed to be non-negative. Likewise for the opposite case. Since we can represent arbitrary $D_p(\cdot)$ in the digraph, then these terms do not affect the 'hardness' of the binary energy—there will *always* exist a reduction for such terms.

Much more interesting are the second-order terms $V_{pq}(\cdot,\cdot)$, or simply V for brevity. Each term is defined by the four constants $V(0,0), V(0,1), V(1,0)$, and $V(1,1)$. For a digraph to represent V *and* be a valid s-t min-cut problem instance, the arc weights $w(u,v)$ must satisfy the following linear constraints:

$$\begin{aligned} &K + w(\{s,p,q\}) = V(0,0)\\ &K + w(\{s,p\}) \quad = V(0,1)\\ &K + w(\{s,q\}) \quad = V(1,0)\\ &K + w(\{s\}) \qquad = V(1,1)\\ &w(u,v) \geq 0 \;\; \forall\,(u,v) \in \mathcal{A} \end{aligned} \tag{2.1}$$

where $K \in \mathbb{R}$ is needed to account for any additive constant irrelevant to the minimization. The question "which V are representable as an s-t cut?" can now be stated as "for which V is system (2.1) feasible in K and w?" We can answer this question either by careful proof [87] or by automatic quantifier elimination [34] (eliminate $\exists K, \exists w$) using a symbolic algebra package. Either way, we find system (2.1) is feasible in K and w if and only if V satisfies

$$V(0,0) + V(1,1) \;\leq\; V(0,1) + V(1,0). \tag{2.2}$$

Theorem 2.4 ([87])**.** *A second-order potential $V_{pq}(\cdot,\cdot)$ is representable as an s-t cut if and only if it satisfies inequality* (2.2)*.*

In other words, the average cost of taking different labels must be at least the average cost of taking the same label. If a binary energy E satisfies (2.2) for every V_{pq} term, then $E(\mathbf{x})$ is said to be *submodular* or a *submodular function*.

Theorem 2.5 ([32, 87])**.** *Minimizing a second-order binary energy E is reducible to an s-t min-cut if and only if $E(\mathbf{x})$ is a submodular function.*

This result completely characterizes the class of second-order binary energies reducible to a graph cut, and therefore answers our original question. If an energy is submodular, it is fundamentally easier to minimize, much as convex functions are. In fact submodular functions are often referred to as “a discrete analog of convex functions” and are actively studied in mathematical optimization [52, 156], machine learning [8], and computer vision [87, 49, 113].

2.2 Local Search for Multi-Label Energies

An energy is considered ‘multi-label’ if its label set has cardinality $|\mathcal{L}| \geq 3$. The s-t cut reduction in Section 2.1 is inherently binary because there are two terminals s and t. So then, how can we minimize a multi-label energy? For some special V_{pq} it is possible to reduce the minimization problem to a *multi-terminal min-cut* [24] and simply apply a known algorithm [33]. However, it turns out that one can do much better, in general, by designing special *local search* algorithms for direct energy minimization, also called *move-making* algorithms in the computer vision literature. There are many strong approaches besides local search, *e.g.* LP-relaxation or message passing algorithms, but they are outside the scope of this dissertation; see Table 1.1 on page 9 for an overview.

Local search is the most basic kind of iterative improvement. Given a current solution $\hat{f}$, we are permitted to move to a better solution f if it belongs to some set of neighbouring[1] solutions $\mathcal{M}(\hat{f})$. The set of labelings $\mathcal{M}(\hat{f})$ can be thought of as the available *moves* from $\hat{f}$. The high-level local search algorithm is given below.

```
LOCALSEARCH($E, \mathcal{M}$)
$\hat{f}$ := arbitrary labeling
while exists $f \in \mathcal{M}(\hat{f})$ such that $E(f) < E(\hat{f})$
    $\hat{f} := f$
return $\hat{f}$
```

[1] Two labelings are ‘neighbours’ if they are similar according to $\mathcal{M}$; note that this use of the word ‘neighbour’ has nothing to do with neighbouring variables defined by $\mathcal{N}$.

The key to effective local search is a good class of moves $\mathcal{M}$. If $\mathcal{M}$ is too broad, then finding the *best* move f can become NP-hard. If $\mathcal{M}$ is too restrictive, then $\hat{f}$ will get stuck at poor local minima. For example, the simplest class of moves is to allow one variable to change at a time,

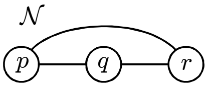
$$\mathcal{M}(\hat{f}) = \{\, f \;:\; f_{\mathcal{P}\setminus\{p\}} = \hat{f}_{\mathcal{P}\setminus\{p\}} \text{ for some } p \in \mathcal{P}\,\}. \tag{2.3}$$

The quality of each move can be evaluated by scanning each $\hat{f}_p$ over all labels while holding the other variables fixed.

The simple search neighbourhood (2.3) corresponds to the classic *iterated conditional modes* (ICM) [103, 13] algorithm for energy minimization. The ICM algorithm is essentially coordinate descent for the MAP-MRF problem, with one variable allowed to vary while the remaining variables stay fixed. ICM is wholly inadequate for the kind of energies we are interested in. To see why, consider a 3-variable, 3-label energy defined by the parameters below.

D	p	q	r
ℓ_1	2	2	2
ℓ_2	1	1	1
ℓ_3	0	0	0

V	ℓ_1	ℓ_2	ℓ_3
ℓ_1	0	1	2
ℓ_2	1	0	1
ℓ_3	2	1	0



The globally optimal labeling is clearly $f^* = (\ell_3, \ell_3, \ell_3)$ with $E(f^*) = 0$. However, if our initial labeling is $\hat{f} = (\ell_1, \ell_1, \ell_1)$ then the possible moves are

$$\begin{aligned}\mathcal{M}(\hat{f}) = \big\{&(\ell_1,\ell_1,\ell_1),\\ &(\ell_2,\ell_1,\ell_1),(\ell_1,\ell_2,\ell_1),(\ell_1,\ell_1,\ell_2),\\ &(\ell_3,\ell_1,\ell_1),(\ell_1,\ell_3,\ell_1),(\ell_1,\ell_1,\ell_3)\big\}\end{aligned} \tag{2.4}$$

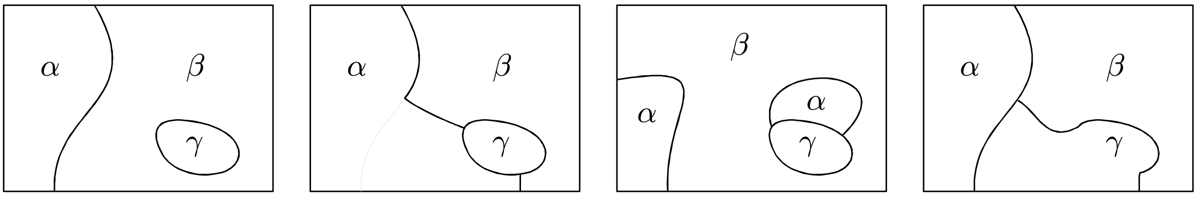
Evaluating the energy on each of these moves gives $E = 6, 7, 7, 7, 8, 8, 8$ respectively, and so $E(\hat{f}) = 6$ is a local minimum with respect to this class of moves. Even if we expand the move space $\mathcal{M}$ to change *two* variables at a time, no neighbouring solution in $\mathcal{M}(\hat{f})$ has energy lower than 6 and so $\hat{f}$ would still be a local minimum.

ICM-style local moves are straight-forward, but require polynomial time to explore only a polynomial number of alternative labelings. It turns out that for some important special cases one can do better—much better in fact. By careful choice of move space $\mathcal{M}$ one can explore an *exponential* number of alternative labelings in only polynomial time. Such local search algorithms are called *very-large search neighbourhood* (VLSN) techniques [2].

We explain two VLSN techniques where the local moves are computed by a graph cut: the $\alpha\beta$-swap algorithm and the α-expansion algorithm [24]. The $\alpha\beta$-swap algorithm is applicable to a slightly wider class of energies but the α-expansion algorithm, when applicable, is more effective both in theory [24] and in practice [139].

2.2.1 $\alpha\beta$-swap for semi-metrics

The $\alpha\beta$-swap algorithm [24] performs local search on multi-label energies. Given current labeling $\hat{f}$, the idea of a *swap move* is as follows. Choose any two labels $\alpha, \beta \in \mathcal{L}$ and allow all variables with $\hat{f}_p \in \{\alpha, \beta\}$ to simultaneously choose a new label in $\{\alpha, \beta\}$. Figure 2.2 shows some examples of swap moves. Intuitively, each variable currently labeled either α or β



current labeling $\hat{f}$ an $\alpha\beta$-swap other $\alpha\beta$-swap a $\gamma\beta$-swap

Figure 2.2: Examples swap moves, all with respect to the current 2D labeling $\hat{f}$ is shown at left. An $\alpha\beta$-swap move is made from binary choices: each f_p involved can only choose either α or β.

is allowed to either keep its current label or 'swap' to the other label. If there are k variables with a current label in $\{\alpha, \beta\}$, then there are 2^k possible $\alpha\beta$-swap moves available.

We can define the full search neighbourhood of $\alpha\beta$-swap as the set of all possible swap moves with respect to current labeling $\hat{f}$:

$$\mathcal{M}(\hat{f}) = \bigcup_{\alpha,\beta\in\mathcal{L}} \mathcal{M}^{\alpha\beta}(\hat{f}) \quad \text{where} \quad \mathcal{M}^{\alpha\beta}(\hat{f}) = \big\{\, f \;:\; f_p \neq \hat{f}_p \;\Rightarrow\; f_p, \hat{f}_p \in \{\alpha,\beta\} \,\big\}. \tag{2.5}$$

The local search algorithm using swap moves can still get stuck at a local minimum, but the move space $\mathcal{M}$ is exponential in size (a VLSN).

If a swap move $f \in \mathcal{M}(\hat{f})$ such that $E(f) < E(\hat{f})$ exists, then we must find it in polynomial time. Fortunately, for any particular $\alpha, \beta \in \mathcal{L}$ an optimal $\alpha\beta$-swap can be computed efficiently by a single binary *graph cut*. This reduction is straight-forward because an $\alpha\beta$-swap move is fundamentally binary: each variable can only choose either α or β. This binary energy will take the standard form

$$E'(\mathbf{x}) \;=\; \sum_{p\in\mathcal{P}} D'_p(x_p) \;+\; \sum_{pq\in\mathcal{N}} V'_{pq}(x_p, x_q) \tag{2.6}$$

where each configuration $\mathbf{x}$ corresponds to an $\alpha\beta$-swap move f via the relation

$$f_p = \begin{cases} \alpha & \text{if } x_p = 0 \\ \beta & \text{otherwise.} \end{cases}$$

The specific costs for data terms D'_p and smooth terms V'_{pq} are determined by the D_p and V_{pq} of the original multi-label energy $E(f)$. Specifically, we set

$$\begin{array}{ll} D'_p(0) := D_p(\alpha) \qquad & V'_{pq}(0,0) := V_{pq}(\alpha,\alpha) \\ D'_p(1) := D_p(\beta) \qquad & V'_{pq}(0,1) := V_{pq}(\alpha,\beta) \\ & V'_{pq}(1,0) := V_{pq}(\beta,\alpha) \\ & V'_{pq}(1,1) := V_{pq}(\beta,\beta) \end{array} \tag{2.7}$$

Minimizing $E'(\mathbf{x})$ implicitly solves the problem $\operatorname{argmin}_{f\in\mathcal{M}^{\alpha\beta}(\hat{f})} E(f)$, thereby finding the best move from among an exponential number of possibilities. The $\alpha\beta$-swap algorithm is generally implemented using the pseudocode below.

```
αβ-SWAP(E) — local search using αβ-swap moves
1 $\hat{f}$ := arbitrary labeling
2 repeat
3   for each $\alpha, \beta \in \mathcal{L}$
4     $f := \arg\min_{f \in \mathcal{M}^{\alpha\beta}(\hat{f})} E(f)$
5     if $E(f) < E(\hat{f})$
6       $\hat{f} := f$
7 until converged          // stop if energy cannot decrease for any $\{\alpha, \beta\}$
8 return $\hat{f}$
```

The key step of $\alpha\beta$-swap is minimizing binary energy E' efficiently (line **4**). This subproblem can be reduced to a single s-t min-cut if and only if E' is *submodular* (Section 2.1.3). For a second-order swap term V'_{pq} to be submodular, the multi-label term V_{pq} must satisfy a corresponding condition:

$$\begin{aligned} & V'_{pq}(0,0) \; + V'_{pq}(1,1) \;\le\; V'_{pq}(0,1) \; + V'_{pq}(1,0) \\ \Longrightarrow \quad & V_{pq}(\alpha,\alpha) + V_{pq}(\beta,\beta) \;\le\; V_{pq}(\alpha,\beta) + V_{pq}(\beta,\alpha) \end{aligned} \tag{2.8}$$

By Theorem 2.5 we have the following consequence.

Corollary 2.6. *The $\alpha\beta$-swap algorithm is applicable for the MAP-MRF energy* (1.1) *if and only if each second-order term $V(\cdot,\cdot)$ satisfies*

$$V(\alpha,\alpha) + V(\beta,\beta) \;\le\; V(\alpha,\beta) + V(\beta,\alpha) \quad \forall\, \alpha,\beta \in \mathcal{L} \tag{2.9}$$

The original paper that introduced $\alpha\beta$-swap defined *semi-metrics* as an intuitive yet sufficient condition for the algorithm to be applicable [24].

Definition 2.7 ([24]). *A second-order term $V(\cdot,\cdot)$ is said to be a semi-metric if it satisfies*

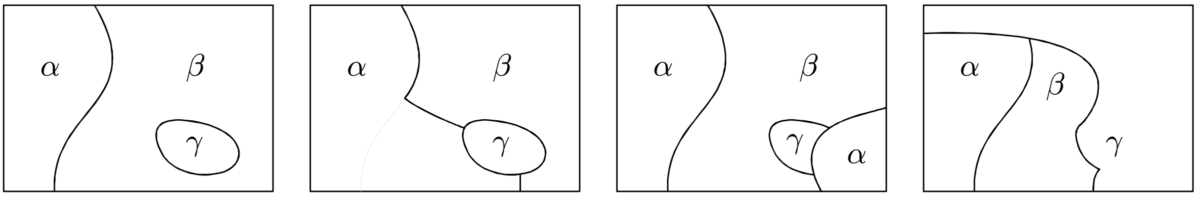
$$\begin{aligned} & V(\alpha,\alpha) = 0 \\ & V(\alpha,\beta) = V(\beta,\alpha) \ge 0 \quad \forall\, \alpha,\beta \in \mathcal{L} \end{aligned}$$

If V is a semi-metric then clearly it satisfies (2.9) and the $\alpha\beta$-swap algorithm is applicable.

2.2.2 α-expansion for metrics

The α-expansion algorithm [24] performs local search using a different class of moves than the $\alpha\beta$-swap algorithm. Given a current labeling $\hat{f}$, an α-expansion move gives each variable the following choice: either keep the current assignment $\hat{f}_p$, or switch to a particular label α. All variables make this choice simultaneously, so there are an exponential number of possible moves with respect to any particular α. Figure 2.3 illustrates some possible expansion moves. The name 'α-expansion' suggests that the label α can grow or 'expand' its territory in the current labeling, but cannot contract.

If we can find the best move efficiently, then we again have a powerful VLSN technique. In fact, it turns out that α-expansion is more effective than $\alpha\beta$-swap both in theory [24] and in



current labeling $\hat{f}$ an α-expansion other α-expansion a γ-expansion

Figure 2.3: Example expansion moves, all with respect to the current 2D labeling $\hat{f}$ is shown at left. An α-expansion move is made from binary choices: α can either 'expand' to pixel p, or leave $\hat{f}_p$ as is.

practice [139]. The α-expansion algorithm is the basis for technical contributions in Chapters 4 and 5 of this dissertation.

We can define the full search neighbourhood of α-expansion as the set of all possible expansion moves with respect to current labeling $\hat{f}$:

$$\mathcal{M}(\hat{f}) = \bigcup_{\alpha \in \mathcal{L}} \mathcal{M}^\alpha(\hat{f}) \quad \text{where} \quad \mathcal{M}^\alpha(\hat{f}) = \{\, f \;:\; f_p \neq \hat{f}_p \;\Rightarrow\; f_p = \alpha \,\}. \tag{2.10}$$

In one sense the move space $\mathcal{M}^\alpha$ seems more restrictive than that of swap moves $\mathcal{M}^{\alpha\beta}$, but in another sense it is more powerful. For an α-expansion move, all variables with label $\hat{f}_p \neq \alpha$ can change their labels, whereas an $\alpha\beta$-swap move can only change the variables with current $\hat{f}_p \in \{\alpha, \beta\}$. Surprisingly, local search with expansion moves will find a labeling $\hat{f}$ within a constant factor from the globally optimal labeling f^* [24]. Section 2.2.3 explains these approximation guarantees, and Chapters 4 and 5 extend the bound to even harder energies. The α-expansion algorithm is generally implemented as shown below.

```
α-EXPANSION(E) — local search using α-expansion moves
1 $\hat{f}$ := arbitrary labeling
2 **repeat**
3     **for each** $\alpha \in \mathcal{L}$
4         $f := \operatorname{argmin}_{f \in \mathcal{M}^\alpha(\hat{f})} E(f)$
5         **if** $E(f) < E(\hat{f})$
6             $\hat{f} := f$
7 **until** converged
8 **return** $\hat{f}$
```

For a particular label $\alpha \in \mathcal{L}$, we need an efficient way to find the α-expansion move $f \in \mathcal{M}^\alpha(\hat{f})$ with minimal $E(f)$ on line **4**. Expansion moves are fundamentally binary so we can encode a move f by a binary vector $\mathbf{x}$ as

$$f_p = \begin{cases} \hat{f}_p & \text{if } x_p = 0 \\ \alpha & \text{otherwise.} \end{cases}$$

We can construct a binary energy E' of the form (2.6) but with data terms D'_p and smooth terms V'_{pq} now defined according to expansion moves:

$$\begin{array}{ll} D'_p(0) := D_p(\hat{f}_p) & V'_{pq}(0,0) := V_{pq}(\hat{f}_p, \hat{f}_q) \\ D'_p(1) := D_p(\alpha) & V'_{pq}(0,1) := V_{pq}(\hat{f}_p, \alpha) \\ & V'_{pq}(1,0) := V_{pq}(\alpha, \hat{f}_q) \\ & V'_{pq}(1,1) := V_{pq}(\alpha, \alpha) \end{array} \tag{2.11}$$

where D_p and V_{pq} are the terms of the original multi-label energy $E(f)$.

We know that minimizing E' is efficient if $E'(\mathbf{x})$ is a submodular function, so finally we must ask *for which multi-label energies does* (2.11) *result in submodular E'?* Using the definition of second-order submodular functions we have

$$\begin{array}{rrcl} & V'_{pq}(0,0) \quad + V'_{pq}(1,1) & \leq & V'_{pq}(0,1) \quad + V'_{pq}(1,0) \\ \Longrightarrow & V_{pq}(\hat{f}_p, \hat{f}_q) + V_{pq}(\alpha, \alpha) & \leq & V_{pq}(\hat{f}_p, \alpha) + V_{pq}(\alpha, \hat{f}_q) \end{array} \tag{2.12}$$

By Theorem 2.5 we have the following consequence.

Corollary 2.8. *The α-expansion algorithm is applicable to the MAP-MRF energy* (1.1) *if and only if each second-order term $V(\cdot,\cdot)$ satisfies*

$$V(\alpha,\alpha) + V(\beta,\gamma) \;\leq\; V(\alpha,\gamma) + V(\beta,\alpha) \quad \forall\, \alpha,\beta,\gamma \in \mathcal{L} \tag{2.13}$$

Again, the original paper that introduced the algorithm defined *metrics* as a simpler yet sufficient condition for α-expansion to be applicable [24].

Definition 2.9 ([24]). *A second-order term $V(\cdot,\cdot)$ is said to be a metric if it is a semi-metric and additionally satisfies*

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

If V is a metric then clearly it satisfies (2.13) and the α-expansion algorithm is applicable.

2.2.3 Approximation bounds of α-expansion

A surprising result from the original α-expansion paper [24] is that local search with expansion moves will terminate at a solution that is guaranteed to have a low energy, and is therefore 'approximately' optimal in a theoretical sense. Good approximation guarantees are highly valued because we know *a priori* that, no matter where we begin our search, we will arrive at a solution that is in some sense reasonable. For NP-hard minimization problems this is the best we can expect in theory and in practice.

Understanding the approximation bound of α-expansion will be helpful for reading Chapters 4 and 5 of this dissertation. Without loss of generality, we assume all second-order terms V_{pq} are the same cost function denoted simply by V. The quality of the approximation guarantee depends on the range of costs in terms V, and so the bound is parameterized accordingly. The following theorem holds for any energy with $D_p(\cdot) \geq 0$ and metric $V_{pq}(\cdot,\cdot) \geq 0$.

Theorem 2.10 ([24])**.** *If f^* is a global minimum of the MAP-MRF energy* (1.1)*, and $\hat{f}$ is a local minimum w.r.t. expansion moves, then*

$$E(\hat{f}) \leq 2cE(f^*) \quad \textit{where} \quad c = \frac{\max_{\alpha \neq \beta \in \mathcal{L}} V(\alpha, \beta)}{\min_{\gamma \neq \zeta \in \mathcal{L}} V(\gamma, \zeta)} \tag{2.14}$$

In other words, α-expansion is a $2c$-approximation algorithm where $c \geq 1$ depends on the ratio of largest to smallest costs in V. Below are some 5-label examples of second-order potentials V, shown in matrix form, that are commonly used in vision.

Potts [119]

0	1	1	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

$c = 1$

linear

0	1	2	3	4
1	0	1	2	3
2	1	0	1	2
3	2	1	0	1
4	3	2	1	0

$c = 4$

truncated linear

0	1	2	2	2
1	0	1	2	2
2	1	0	1	2
2	2	1	0	1
2	2	2	1	0

$c = 2$

Underneath we see the coefficient c corresponding to each case. The simplest potential (Potts) simply penalizes $f_p \neq f_q$ equally, and gives the best approximation bound. When the range of values is large, *e.g.* for "linear" smooth costs, the bound (2.14) gets worse. Chapter 5 introduces a new algorithm that can beat this bound for a wide class of second-order potentials.

Chapter 3

Global Optimization of Multi-Surface Interactions

Many objects contain spatially distinct regions, each with a unique colour/texture model. Mixture models ignore the spatial distribution of colours within an object, and thus cannot distinguish between coherent parts versus randomly distributed colours. We show how to encode geometric interactions between distinct region+boundary models, such as regions being interior/exterior to each other along with preferred distances between their boundaries. This is similar to the work of Li, Wu, Chen & Sonka [102], except in our construction we do not need “domain unwrapping” nor do we have topological limits on shapes.

With a single graph cut, our method extracts only those multi-region objects that satisfy such a combined model. We show applications in medical segmentation and scene layout estimation. This chapter is based directly on my joint publication with Yuri Boykov at the 2009 *International Conference on Computer Vision* (ICCV) [36]. The work has since been used as the basis for state-of-the-art cardiac segmentation tool [148] and extended with new optimization techniques.

3.1 Overview and Related work

State-of-the-art segmentation methods benefit from an appearance model of the object’s interior and its boundary. Such methods include active contours, level sets, graph cuts, and random walker. With binary segmentation, the object’s entire appearance must be incorporated into a single mixed model. Most real-world objects are better described by a combination of regions with distinct appearance models, and attempts to use multi-label segmentation reflect this, *e.g.* [61, 125]. Our new multi-region segmentation framework maintains a separate region+boundary model for each part of an object, and allows these parts to interact spatially.

Figure 3.1 shows the most basic type of object that we can deal with effectively, and suggests the main advantage we have over standard binary or Pott’s-like models.

Our work is a few short steps from a number of existing techniques either from a conceptual or technical point of view. For example, what we call a multi-*region* model is ultimately a multi-label model, though we add simple yet important geometric constraints and then optimize with a single graph cut[1]. To help make our contribution clear, we begin by situating our work relative to other methods.

Pictorial structures We briefly juxtapose our work with the well-known *pictorial structures* [44], not because our work is directly related, but because we address an analogous problem for objects of a completely different sort. Like [44], our models guarantee optimality only under certain conditions. The table below contrasts our works.

	pictorial struct. [44]	**this work**
shape of each part	fixed template	arbitrary region
spatial prior	relative part positions	boundary distances
optimization	dynamic programming	single graph cut
optimum guaranteed	if tree connectivity	if no “frustrated cycles”

Here “arbitrary region” means that each region does not itself have a specific preferred shape. Such part models can be good, or very bad, depending on the application. One can think of this work as introducing basic distance priors between shapes in a globally optimal way, though incorporating shape priors [155] themselves could be powerful.

Multi-label segmentation Our multi-region models are, generally speaking, a type of multi-label model. One superficial distinction is that an n-region model potentially has 2^n corresponding labels. The reason will be apparent from our graph construction, and we discuss a related idea called *log transformation* [121] toward the end of the chapter.

Our first contribution, stated in terms of multi-label models, is to introduce priors on the distance between pairs of discontinuities (or “region boundaries” as we call them). This is achieved by certain long-range interactions between pixels, and stands in contrast to Pott’s or random walk models, applied for example in [125] and [61] respectively.

Second, multi-label models often require approximate methods such as α-expansion [24]. We strive for an intuitive characterization of the conditions under which our models can be optimized by a single graph cut. A fully general characterization of when multi-label global

[1] Our ideas may also apply in other optimization settings, e.g. [5, 118].

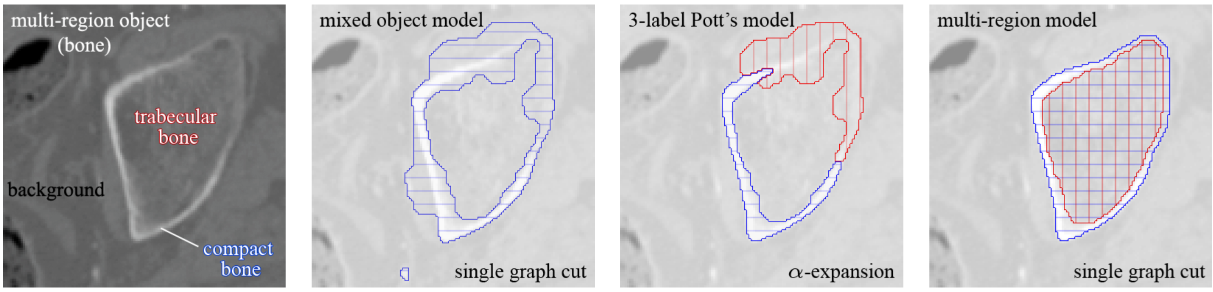


Figure 3.1: Our simplest motivating example. Standard binary [20, 123] and multi-label [24, 125] models fail because object/background colours are hard to separate. In the absence of user localization, above at center is the best result we can expect from such models. Now we can design multi-region models with geometric interactions to segment such objects more robustly in a single graph cut.

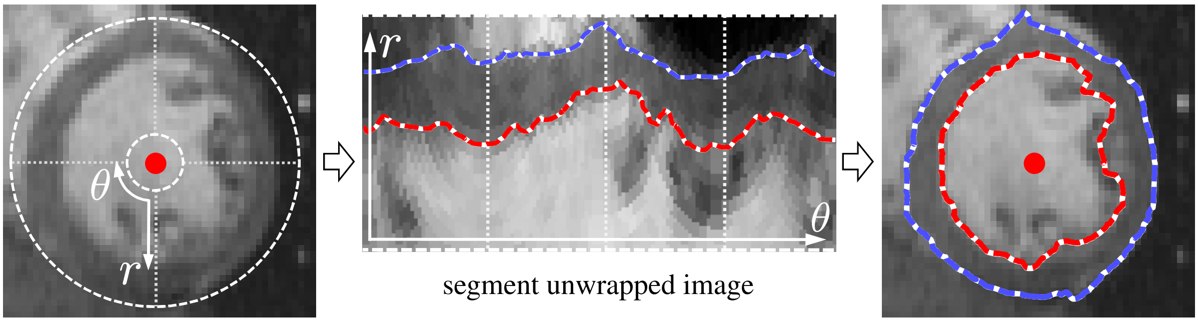


Figure 3.2: To segment an image, Li et al. [102] must work within a band that already follows the object's rough shape by estimating from a center-line/point. They then 'unwrap' the band into polar coordinates because their construction (Figure 3.3) requires it.

optima are guaranteed [127] does not have a meaningful interpretation for specific problems. Elegant interpretations do exist for special cases however, such as Ishikawa's convex characterization [71]. Rather than testing multi-label models against abstract criteria [121, 127], we describe one way to design easy-to-optimize models in an intuitive piecewise manner.

Optimal polar surfaces Li, Wu, Chen & Sonka [102] proposed a multi-surface segmentation technique that inspired our work. The main drawback of their method is that it is hard to use on anything except cylindrical objects; topological changes, bifurcations, or even strong curvature all require careful pre-segmentation. Figure 3.2 shows the underlying problem: their need for a polar representation of the image domain from which they can *unwrap* and optimize only along columns.

They start by assuming that a center-point (center-line) of an object in 2D (3D) is given. After casting outward rays and unwrapping them to obtain a polar representation of the image, they can segment multiple nested surfaces along the resulting columns. They model the segmentation as a *closure set* problem on a special graph, but our Figure 3.3 suggests an equivalent s-t min cut construction for the simplest case. They can encode a minimum and maximum distance constraint between consecutive surfaces. This all assumes that each surface intersects each ray at only one location. Their construction should also allow for soft spring-like forces, although they do not state this.

Our graph construction sidesteps the unwrapping issue entirely. We do not need center-lines, have no topological constraints, and do not suffer from geometric distortion introduced by unwrapping. Briefly, our construction represents a multi-region object by a directed graph comprising an unordered set of layers, with one layer per region. Each layer has one vertex per image pixel[2]. Each layer by itself is just an independent binary graph cut problem familiar in binary segmentation [20]. We introduce inter-layer arcs in the graph that give effects analogous to [102] yet are easier to implement and useful in more general settings.

The chapter is organized as follows. Section 3.2 introduces our multi-region segmentation framework, describing our energy, geometric interaction terms, and our regional terms.

[2] This assumption serves to make our notation more bearable. In general, the layers may represent an image at different resolution, matching the scale at which the corresponding part's features appear in the data.

Section 3.3 demonstrates two applications: medical segmentation and scene layout estimation. Certain combinations of geometric interactions cannot be optimized by graph cuts, and Section 3.4 discusses ways to handle these cases. Section 3.5 concludes and suggests further applications.

3.2 Our Multi-Region Framework

We begin by describing three intuitive geometric interactions in their simplest form:

Containment. Region B must be inside region A, perhaps with repulsion force between boundaries.

Exclusion. Regions A and B cannot overlap at any pixel, perhaps with repulsion force between boundaries.

Attraction. Penalize the area $A - B$, exterior to B, by some cost $\alpha > 0$ per unit area. Thus A will prefer not to grow too far beyond the boundary of B.

As suggested above, we can introduce a distance prior between region boundaries in the form of a hard or soft margin. The prior is enforced in the graph construction by an inter-layer neighbourhood at each pixel p. The local weight and shape for this neighbourhood can vary at each pixel. Figure 3.4 shows how these interactions combine to add discriminative power to object models in segmentation.

3.2.1 Multi-region energy

We define $\mathcal{P}$ to be the set of pixel indices and $\mathcal{L}$ to be the set of region indices. Our binary variables are $x \in \mathbb{B}^{\mathcal{L}\times\mathcal{P}}$ which we index as x_p^i over pixels $p \in \mathcal{P}$ and over regions $i \in \mathcal{L}$. The set $\mathcal{L}$ is *not* ordered. For now we interpret $x_p^i = 1$ to mean that pixel p is interior to region i. The notation x_p denotes a vector of *all* variables that correspond to pixel p, one for each of the

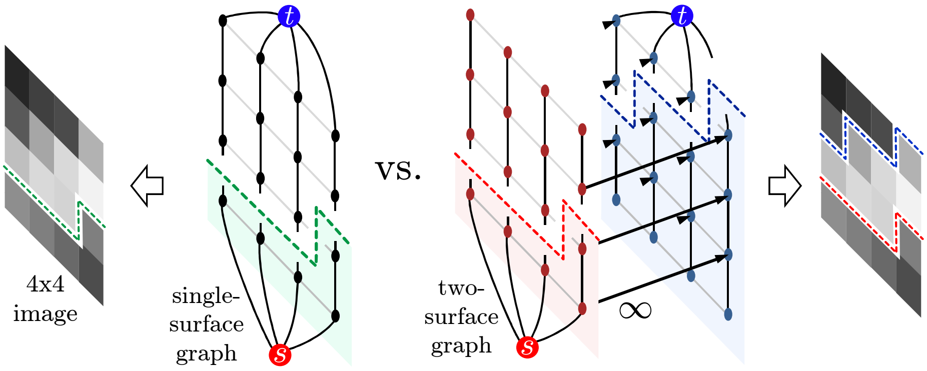


Figure 3.3: LEFT: s-t min cut construction corresponding to [102]; any cut must separate top row from bottom row. RIGHT: Basic idea from [102]. Each column separates top from bottom at two distinct locations, one forced to be strictly above the other.

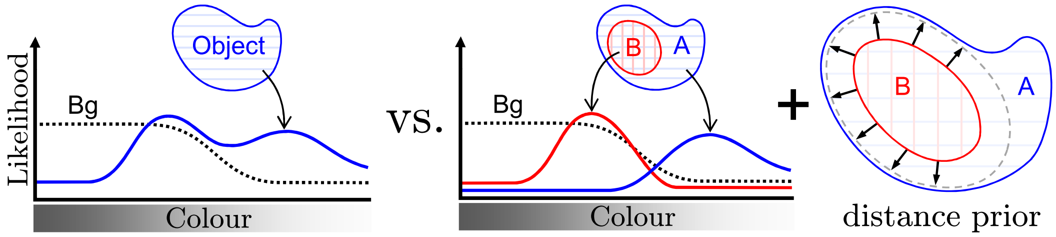


Figure 3.4: LEFT: Mixed colour model corresponding to Figure 3.1. The vertical axis indicates how likely we are to observe a colour for the given class (Bg/Object). RIGHT: Two-region model corresponding to the final result in Figure 3.1. Trabecular bone (B) is forced to be inside a band of compact bone (A) of some estimated thickness.

$|\mathcal{L}|$ regions. If $x_p = \mathbf{0}$ then pixel p is considered "background." The notation x^i refers to all variables of a particular region $i \in \mathcal{L}$.

To express our multi-region energy, we start with two familiar components: data terms and regularization terms. Each pixel p has associated function D_p that defines a cost for every *combination* of regions. Each region i is regularized independently in a standard way by a collection of smoothness terms V^i defined as

$$V^i(x^i) = \sum_{pq \in \mathcal{N}^i} V^i_{pq}(x^i_p, x^i_q) \tag{3.1}$$

where each neighbourhood $\mathcal{N}^i$ typically defines nearest-neighbour grid connectivity.

Ideally each data cost $D_p(x_p)$ could be arbitrary but, because D_p is a function of $|\mathcal{L}|$ binary variables, graph cuts requires that D_p be *submodular* [87]. Ramifications of this are discussed in Section 3.2.3. Each V^i plays the same surface-regularization role as in standard binary segmentation. For the case $|\mathcal{L}|=1$ our D_p and V^i obviously describe a standard binary energy, solvable by graph cut [20].

When $\mathcal{L}$ indexes multiple regions, we can add a new category of energy terms to encode inter-region interactions. Our multi-region energy takes the overall form

$$E(\mathbf{x}) = \sum_{p \in \mathcal{P}} D_p(x_p) \; + \sum_{i \in \mathcal{L}} V^i(x^i) \; + \overbrace{\sum_{\substack{i,j \in \mathcal{L} \\ i \neq j}} W^{ij}(x^i, x^j)}^{\text{interaction terms}}. \tag{3.2}$$

where each W^{ij} encodes all geometric interactions between regions i and j.

To understand how our interaction terms W^{ij} are indexed over both region pairs (i, j) and pixel pairs (p, q), it helps to consider Figure 3.5 along with the definition for one particular pair of regions

$$W^{ij}(x^i, x^j) = \sum_{pq \in \mathcal{N}^{ij}} W^{ij}_{pq}(x^i_p, x^j_q). \tag{3.3}$$

The inter-region neighbourhood $\mathcal{N}^{ij}$ is the set of all pixels pairs (p, q) at which region i is assigned some geometric interaction with region j. We allow $(p, p) \in \mathcal{N}^{ij}$ because they refer to separate variables, unlike in $\mathcal{N}^i$. Note that W^{ii} and V^i would describe the same set of energy terms, but the conceptual distinction is just as important as the distinction between V_{pp} and D_p.

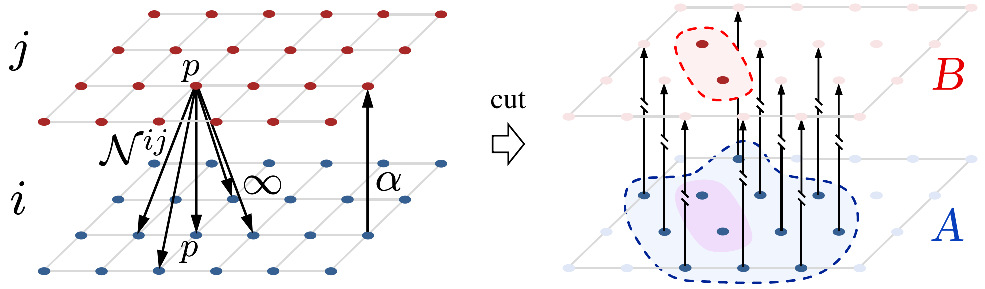


Figure 3.5: LEFT: Graph construction for region layers $i, j \in \mathcal{L}$ showing a subset of inter-region connectivity $\mathcal{N}^{ij}$. The ∞-cost arcs, shown emanating only from x_p^j, enforce a 1-pixel margin between region boundaries. RIGHT: The α-cost arcs attract the outer boundary by penalizing only the area $A - B$.

Section 3.2.2 details the energy terms and corresponding graph construction for our containment, exclusion, and attraction interactions. Section 3.2.3 then discusses limitations of our higher-order data terms.

3.2.2 Geometric interactions

We now describe how our geometric interactions can be implemented with a single graph cut. The basic “i contains j” interaction is simplest, so we start there. All we do is introduce a term $W_{pp}^{ij}(0,1) = \infty$ at every pixel $p \in \mathcal{P}$. Those familiar with graph constructions may prefer to think of it as an ∞-cost arc from vertex x_p^j to x_p^i, thus prohibiting any cut that labels them 1 and 0 respectively. More generally we can add similar terms W_{pq}^{ij} for $p \neq q$. For example, to add a hard uniform margin to our containment constraint, we set $W_{pq}^{ij}(0,1) = \infty$ for all q within some radius of p.

The tables below list energy terms corresponding to our three main interactions.

i **contains** j		
x_p^i	x_q^j	W_{pq}^{ij}
0	0	0
0	1	∞
1	0	0
1	1	0

i **excludes** j		
x_p^i	x_q^j	W_{pq}^{ij}
0	0	0
0	1	0
1	0	0
1	1	∞

i **attracts** j		
x_p^i	x_p^j	W_{pp}^{ij}
0	0	0
0	1	0
1	0	α
1	1	0

$$\alpha > 0 \tag{3.4}$$

Figure 3.5 shows the graph construction corresponding to the containment and attraction interactions. A soft containment cost $W_{pq}^{ij}(0,1) > 0$ for $p \neq q$ creates a spring-like repulsion force between the inner and outer boundaries. Note that our distinction between “containment” and “attraction” is largely artificial since they are the same type of constraint but with opposite orientation.

The exclusion interaction is more difficult because it cannot be optimized by graph cuts until we perform a simple transformation. The reason is because graph cuts can only optimize certain *submodular* functions (see Section 2.1.3 for review). As shown in [87], a second-order energy $E(\mathbf{x})$ over binary $\mathbf{x}$ is submodular if and only if it can be expressed as a sum of pairwise functions

$$E(\mathbf{x}) = \sum_i E_i(x_i) + \sum_{i,j} E_{ij}(x_i, x_j) \tag{3.5}$$

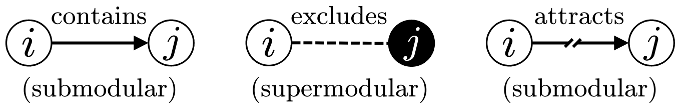
where each second-order term satisfies

$$E_{ij}(0,0) + E_{ij}(1,1) \leq E_{ij}(0,1) + E_{ij}(1,0). \tag{3.6}$$

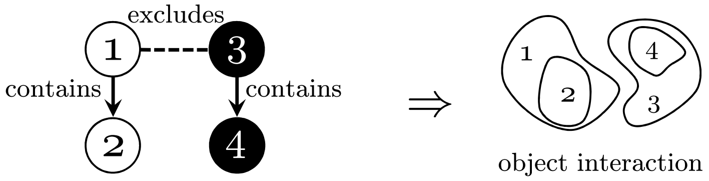
Our containment and attraction interactions are submodular, but for our exclusion terms W^{ij} in (3.4) clearly the reverse inequality holds, so exclusion is *super*modular. Because exclusion is everywhere supermodular, we can flip the meaning of layer j's variables so that $x^j_p = 0$ designates the region's interior. Our exclusion terms $W^{ij}(x^i, \bar{x}^j)$ thus become submodular, so long as we can flip the variables.

The idea of flipping variable meanings among supermodular terms is not a new idea. It lies at the heart of *roof-duality* methods in quadratic pseudo-boolean optimization (QPBO) [18, 85, 124]. These methods are more sophisticated than graph cuts, consuming more time and memory, so we prefer not to rely on them unless necessary (Section 3.4).

Let us now explore the overall geometric interactions permitted by combining the three basic ones in (3.4). To aid the discussion, we introduce graphical depictions of each interaction between two objects i and j.

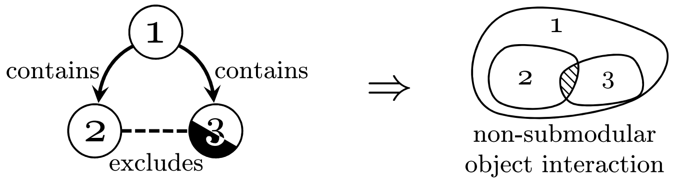


We can allow more sophisticated interactions, such as a hierarchy of nested regions or regions excluded from one another. The example below models two mutually exclusive regions, each with an interior part. A black circle indicates that the region's label is complemented in order for the overall problem to remain submodular.



(3.7)

There are many useful interactions that we cannot model with graph cuts. The example below describes two mutually exclusive regions, both contained within another region.



(3.8)

The above configuration cannot be trivially converted to a submodular energy. It introduces what is called a *frustrated cycle* among the overall pairwise energy terms. A cycle is called frustrated if it contains an odd number of non-submodular terms (see $\mathcal{P}3, \mathcal{P}4$ in [124]). This means that with graph cuts we can only model interactions that are bipartite with respect to exclusion, and submodular interactions cannot be added between layers that use opposite $0/1$ labels. If we step outside these constraints then global optima are no longer guaranteed, but approximations such as QPBO-I [124] or $\alpha\beta$-swap [24] may still be effective. (Section 3.4 explains why α-expansion often cannot be applied.)

3.2.3 Regional data terms

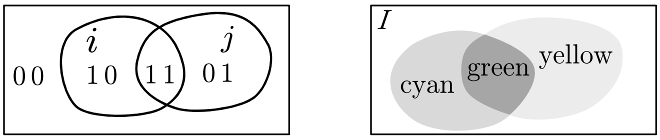
We begin by showing how the likelihoods in Figure 3.4 are used to drive the segmentation in Figure 3.1. We have $\mathcal{L} = \{A, B\}$ so each data term D_p defines up to 4 costs. Given image data I with individual pixel intensities I_p, each function D_p is naturally described by the table below.

x_p^A	x_p^B	D_p
0	0	$-\log \Pr(Bg \mid I_p)$
0	1	K
1	0	$-\log \Pr(A \mid I_p)$
1	1	$-\log \Pr(B \mid I_p)$

(3.9)

The unspecified cost K brings us to an important point. The cost K is not driven by the image data itself, because the "A contains B" object model prohibits this configuration. For this particular model, each $D_p(x_p)$ is added alongside pairwise term W_{pp}^{AB} having cost $W_{pp}^{AB}(0,1) = \infty$. The three likelihoods (3.9) can therefore be arbitrary for this object model, without concern for K or for submodularity. Submodularity of our overall energy (3.2) thus depends on a *combination* of data terms and interaction terms.

Suppose however that there were no geometric constraints between two layers i and j. The data terms $D_p(x_p)$ must then be submodular (or supermodular if the label for j is flipped). To understand what this means intuitively, consider two regions i and j that represent subtractive colours.



(3.10)

Here, submodularity requires that each data term satisfy

$$D_p(0,0) + D_p(1,1) \leq D_p(0,1) + D_p(1,0). \tag{3.11}$$

One symmetric way to satisfy (3.11) is to say, for example, that D_p for a cyan pixel I_p does not simply encourage region i, but also *discourages* region j by an equal amount.

For models with strong geometric interactions, such as containment and exclusion, these constraints on D_p are usually satisfied for reasons suggested by (3.9).

Higher-order data terms A data term D_p may model three or more regions with dependent data costs, but graph cuts can only encode pairwise energy terms directly. Any function of three or more variables can be transformed into a combination of pairwise and unary terms in polynomial time [18]. Transforming a submodular 3^{rd}-order term preserves submodularity among the resulting pairwise terms [87]. For a 4^{th}-order term or higher there are submodularity-preserving transformations only for certain cases [49, 156]. To solve the resulting pairwise problem with a single graph cut, one must truncate the non-submodular data terms to approximate the desired energy. (None of our medical examples needed truncation.) An alternative is to use QPBO [18] and its extensions [124] directly on the non-submodular energy.

3.3 Applications

We choose two problems that we hope demonstrate the diverse applications of our framework. Section 3.3.1 shows how our multi-region energy (3.2) helps to model many objects in medical image segmentation. Section 3.3.2 proposes a novel way to regularize basic scene layout estimation using Hoiem-style[3] data terms [66].

3.3.1 Medical segmentation

Medical image segmentation is a domain full of multi-part objects that are hard to detect with rigid part-models such as [44]. This is why so many state-of-the-art algorithms [5, 20, 61, 102] rely on region+boundary models over arbitrary shapes using mainly length/area priors. Of these techniques, only the recent work of Li, Wu, Chen & Sonka [102] attempts to globally optimize priors on the distance between multiple surfaces. As shown in Figures 3.2 and 3.3, they rely on accurate center-line estimation (a difficult problem in itself) and cannot handle complex topologies.

Figures 3.1, 3.6, 3.7 and 3.8 show experimental results of our multi-region framework using class-specific models (bone, knee, heart, kidney). The heart result was computed using QPBO-I, and the rest were computed in a single graph cut. Our early experiments are all 2D but they extend to N-D in a straight-forward manner. Using the Boykov-Kolmogorov max-flow algorithm [22] our running times are longer than binary graph cut in roughly linear proportion to the number of vertices and arcs added to the graph.

3.3.2 Scene layout estimation

Given a photograph of a scene, we wish to break the image into rough geometric labels “bottom” (B), “top” (T), “left wall” (L), ”right wall” (R) and “front-facing” (F). This application is described by Hoiem et al. [66], and we actually use data terms based on their local geometric class estimators. See Figure 3.9 for an example result. Instead of using α-expansion to find a local minimum of a Pott’s energy, we design a set of interactions between class regions that can be optimized by a single graph cut.

In our setup, we let regions $\mathcal{L} = \{B, L, T, R\}$ and treat F as background. Ideally we want every pixel $p \in \mathcal{P}$ to be assigned a unique region, but adding this constraint introduces frus-

[3] We thank Derek Hoiem so, *so* much for making code [66] available.

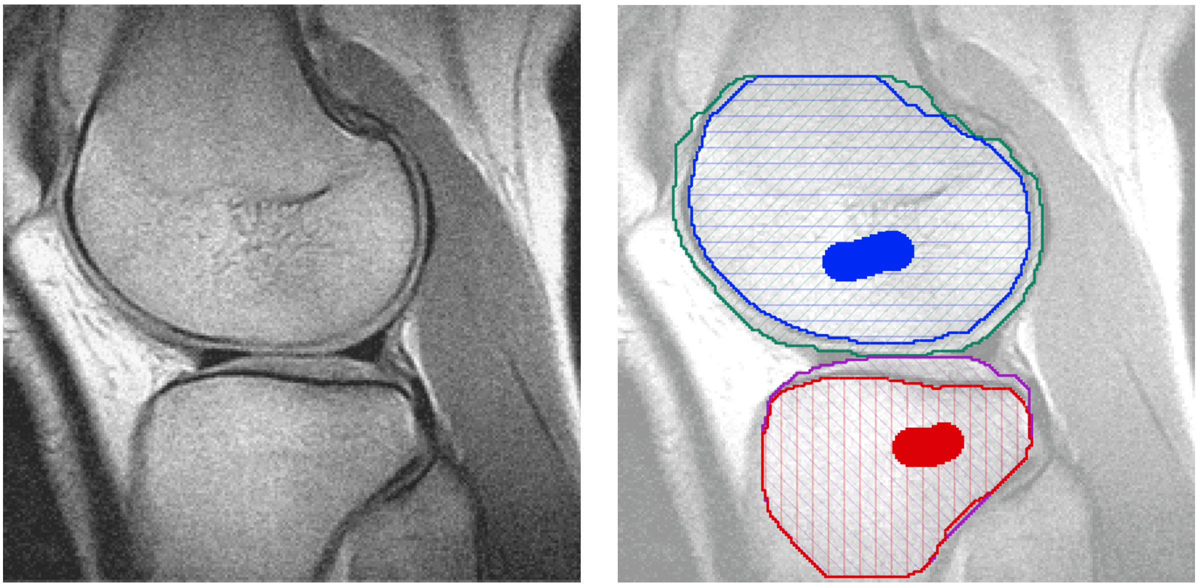


Figure 3.6: User-driven segmentation of knee joint, measuring thickness of cartilage. Above uses the 4-part submodular interaction portrayed in (3.7), and was computed by a single graph cut. Given the some pixels manually classified by the user (dark regions), bone and cartilage are segmented automatically using a combination of image gradients and anisotropic distance prior (margin) between surfaces. A two-part model, using these same user input for either tibia or femur, gives poor results.

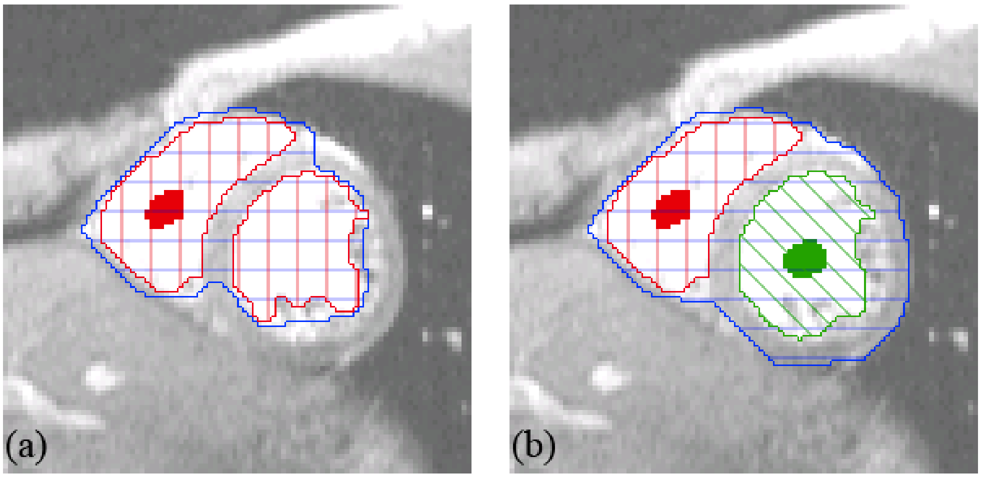


Figure 3.7: User-driven heart segmentation using the non-submodular interaction portrayed in (3.8), solved by QPBO-I. The user first marks a part of the right ventricle (a), but the sampled colour model is attracted to both ventricles. The user then marks the left ventricle as a separate region (b). The outer wall is segmented automatically by compromising between image gradients and distance prior (margin). This model cannot be handled by Li *et al.* [102], and was the starting point for the cardiac segmentation model of Ulén *et al.* [148].

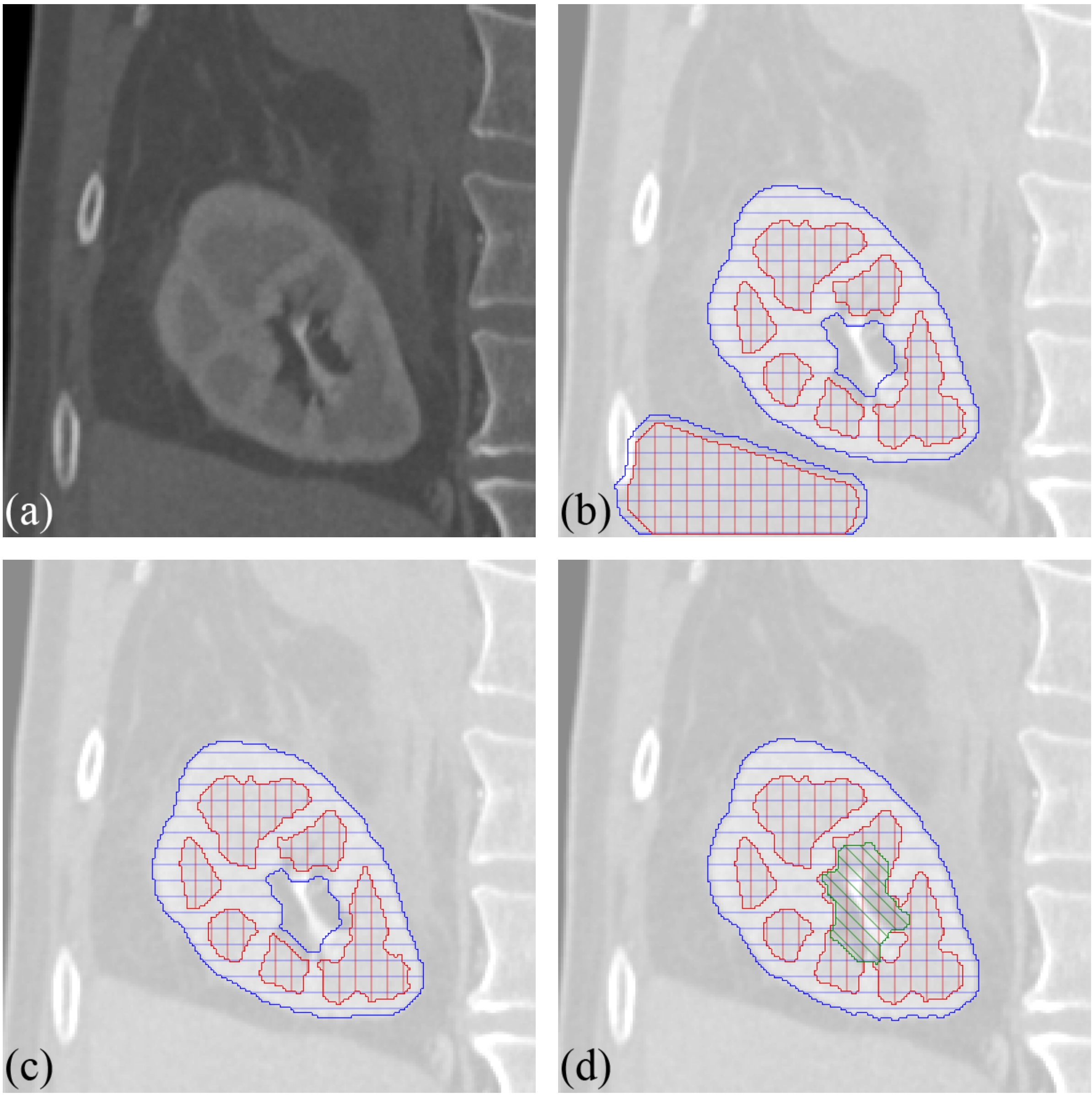


Figure 3.8: Kidney segmentation (a) is very difficult to automate due to low contrast and complex topology. Binary graph cuts simply cannot get reasonable results without heavy user interaction, and even multi-label methods need some form of localization [125, 61]. In (b–c) we model the kidney as medulla surrounded by a slightly brighter cortex of minimum thickness. On this challenging example our method is very sensitive to colour/geometric parameters, *e.g.* (b), but has discriminative power to extract only the correct object (c) without any localization. We also show an alternate 3-region object model (d) that eliminates the unwanted margin between medulla and collection cavity (dark/bright interior). This kind of topology would be impossible to segment using Li *et al.* [102] due to the unwrapping requirement.

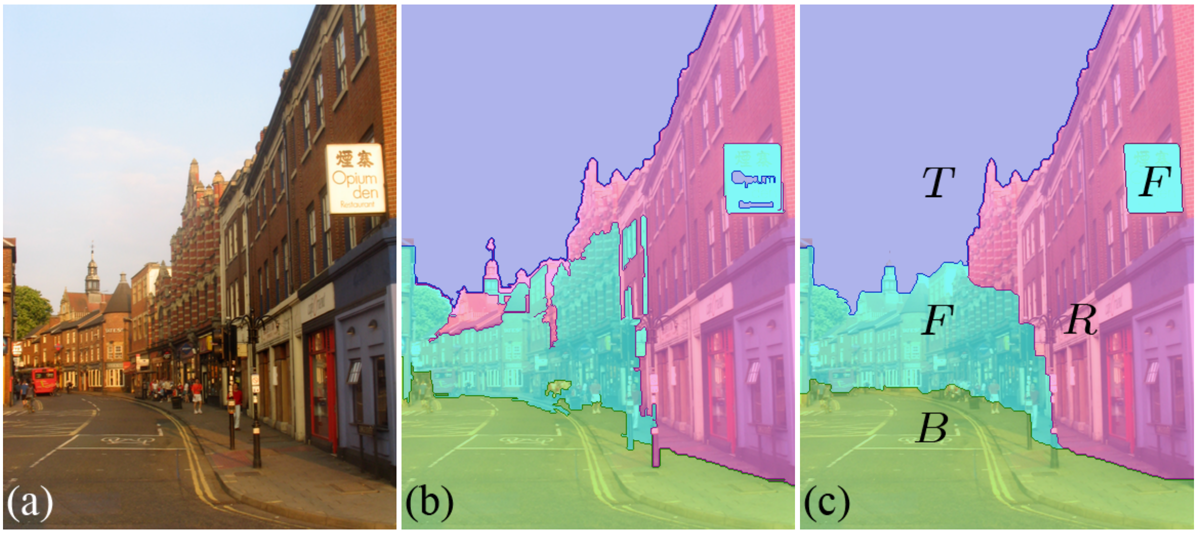
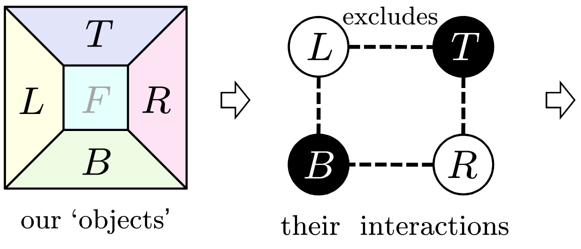


Figure 3.9: Scene layout estimation. Given a scene (a) we first generate data terms from local surface class confidences given by Hoiem et al. [66]. The maximum likelihood solution is shown in (b). With a single graph cut, our multi-region framework regularizes noise/gaps in the data (c) while keeping most important geometric classes (B, L, T, R, F) mutually exclusive throughout the image.

trated cycles (Section 3.2.2). We propose the subset of interactions and data terms portrayed in Figure 3.10.

To encourage the "box" layout seen in Figure 3.10 we borrow an idea from [105] and bias region B against cutting underneath itself using length terms V^B, and likewise for orientations L, T, R. Unlike [105] we do this with a soft penalty so that strong local data terms can override the prior, such as the front-facing sign in Figure 3.9c.

We still have two unwanted configurations BT and LR that have no corresponding likelihood. To discourage these labels we want to maximize corresponding D_p, but higher-order



$BLTR$	$D_p(\cdot)$
1 0 1 0	$D_p(F)$
0 0 1 0	$D_p(B)$
1 1 1 0	$D_p(L)$
1 0 0 0	$D_p(T)$
1 0 1 1	$D_p(R)$
0 0 0 0	$D_p(BT)$
1 1 1 1	$D_p(LR)$

Figure 3.10: Our scene object interactions and corresponding higher-order data terms. Two unwanted labels are due to limitations imposed by frustrated cycles (Section 3.2.2). The configurations not listed have cost $\geq \infty$ due to the four exclusion constraints above.

submodularity requires

$$\begin{aligned} D_p(BT) &\leq D_p(B) + D_p(T) - D_p(F), \text{ and} \\ D_p(LR) &\leq D_p(L) + D_p(R) - D_p(F). \end{aligned} \tag{3.12}$$

We truncate these terms to retain submodularity, potentially allowing either B to overlap T, or L to overlap R. Experimental results are shown in Figure 3.11.

Note that even if we did prohibit labels BT and LR, we would not be minimizing a Pott's energy. Instead, the equivalent multi-label formulation has labels $\mathcal{L} = \{l_\emptyset, l_1, \ldots, l_n\}$ where we designate $l_\emptyset$ the *null* label, corresponding to region F in our scene layout formulation. In this type of multi-label model, all pixel label pairs $f_p, f_q \neq l_\emptyset$ have

$$V_{pq}(f_p, f_q) = V_{pq}(f_p, l_\emptyset) + V_{pq}(l_\emptyset, f_q). \tag{3.13}$$

Because this model always penalizes (l_i, l_j) transitions more than $(l_i, l_\emptyset)$ transitions, oversmoothing creates gaps between l_i and l_j in regions with weak data, unlike [66, 105].

3.4 Discussion

Given one of our multi-region models, one could apply $\alpha\beta$-swap to the corresponding multi-label energy. Unfortunately this provides no optimality guarantees, and Figure 3.12 suggests how our distance priors create local minima for $\alpha\beta$-swap. Often the α-expansion algorithm cannot even be applied because the equivalent multi-label energy is not a *metric* [24] and would create non-submodular terms at the expansion step. Specifically, let $V_{pq}(f_p, f_q)$ denote the pairwise cost corresponding to Figure 3.12. The costs here do not satisfy the triangle inequality because

$$V_{pq}(B, \emptyset) \nleq V_{pq}(B, A) + V_{pq}(A, \emptyset). \tag{3.14}$$

The Pott's-like model suggested by (3.13) *is* a metric, however, and can be optimized effectively with α-expansion. On the few scene layout examples we tried, α-expansion either found or came close to the global optimum.

Multi-label constructions Recall that our set of regions $\mathcal{L}$ is not ordered in any way. We are thus *not* building a 'layer cake' construction typical of discrete and continuous total-variation methods in multi-label optimization [71, 117, 118]. A special case of our multi-region energy (3.2) does coincide with a particular Ishikawa construction [71]. To construct a total-variation ($V_{pq} \propto |f_p - f_q|$) Ishikawa graph for n labels, order $n-1$ regions as $\mathcal{L} = \{1, \ldots, n-1\}$ and introduce hard "i contains $i+1$" constraints between subsequent layers.

Also recall that an n-region model represents up to 2^n corresponding labels, which is the ultimate objective of the *log transformation* [121]. They start with an energy over discrete variables $x_i \in \{1 \ldots m\}$ and try to represent each x_i using as close to $\log_2 m$ binary variables as possible. Their approach is much more general because they start from a multi-label energy and test it against a criterion for transformation to submodular binary encoding. The criterion itself is clear but it is not always obvious how to satisfy it when designing an energy for a particular application. In contrast, we *start* with binary variables and build up our multi-region

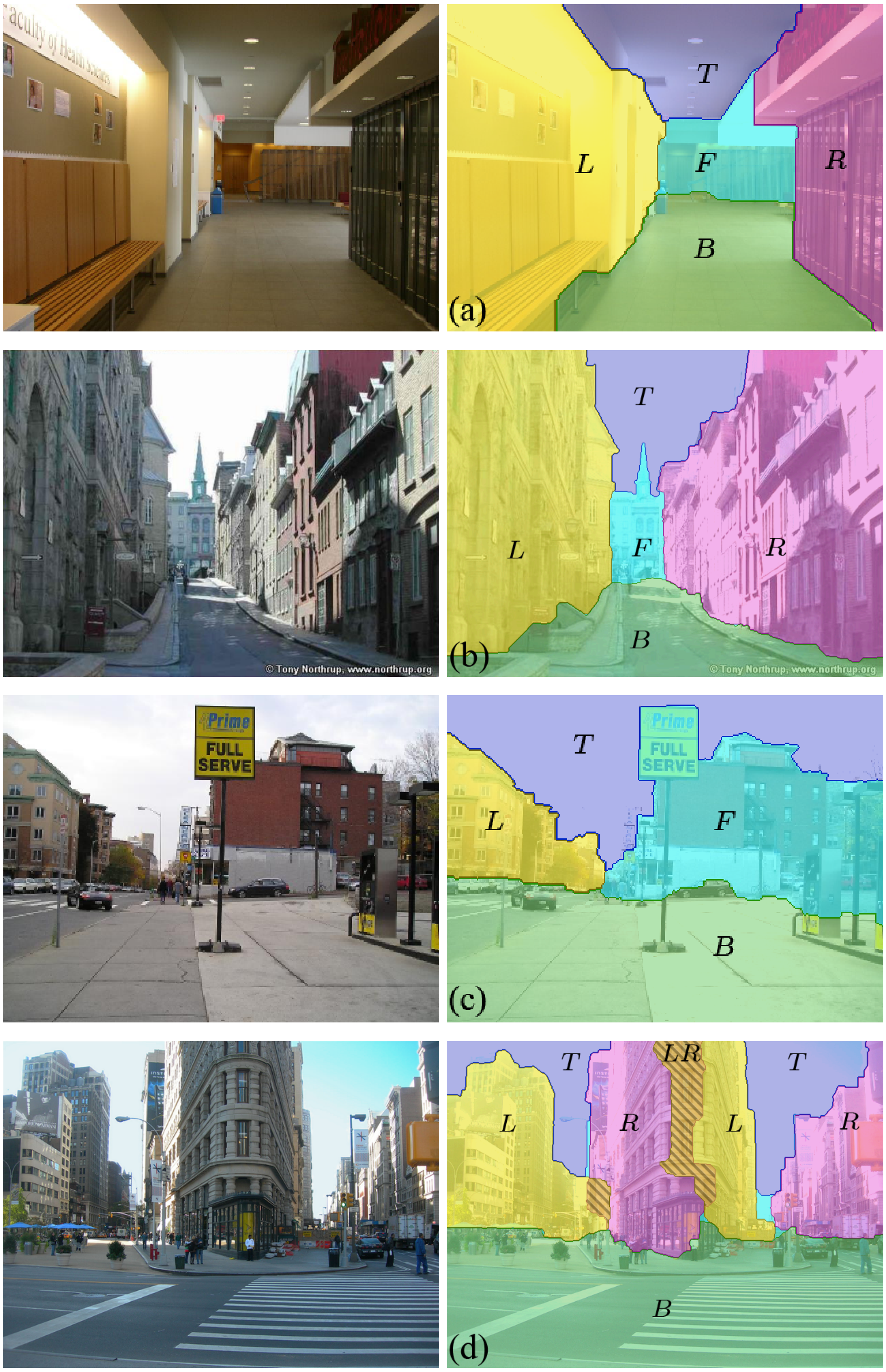
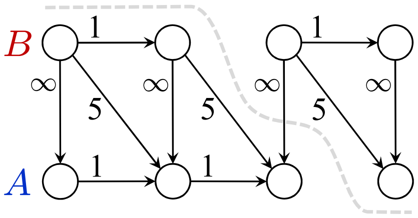

T
L
F
R
B
(a)
T
L
F
R
B
(b)
© Tony Northrup, www.northrup.org
© Tony Northrup, www.northrup.org
Prime
FULL
SERVE
T
L
F
B
(c)
T
LR
T
L
R
L
R
B
(d)

Figure 3.11: Scene layout results using our proposed interactions in Section 3.3.2, showing estimates for indoor (a) and outdoor (b–d) scenes. Smoothness parameters were tuned for each image. Diagonal shading on the Flatiron image (d) indicates that scene classes L and R overlap. This may happen when certain data terms conflict because our graph cut construction cannot simultaneously prohibit all classes from overlapping. Section 3.4 discusses ways to resolve this. See Figure 3.9b for an example of the data terms that drive this segmentation.



	costs $D_p(\cdot)$			
B	0	0	0	∞
A	4	0	0	∞
$\emptyset$	3	0	0	0

labeling $\hat{f}$				$E(\hat{f})$
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	3
A	A	A	$\emptyset$	4
B	B	$\emptyset$	$\emptyset$	6

Figure 3.12: Example of how our interaction terms cause $\alpha\beta$-swap to get stuck in local minima. The graph and D_p encode a 4-pixel segmentation with "A contains B" constraint. Diagonal arcs encourage a 1-pixel margin between boundaries of B and A. Our s-t min cut construction finds global optimum $f^* = (B, B, A, \emptyset)$ with $E(f^*) = 0$, but the corresponding 3-label energy is hard for $\alpha\beta$-swap to optimize. The labeling $\hat{f} = (\emptyset, \emptyset, \emptyset, \emptyset)$ is already a local minimum regardless of which labels are swapped (right).

models from intuitive pairwise interactions. We show that there are applications where such models are useful, without the need for an explicit transformation from multi-label.

Constructions along 'rays' On page 25 we described a related construction by Li, Wu, Chen & Sonka [102] that optimizes along columns sampled from the image domain. Notice that because their columns are known *a priori* they can encode both a min and max distance prior, whereas our framework assumes rays are not known and can only encode a min distance[4]. Thus there is an advantage to their method when a good pre-segmentation is available.

On the subject of paper [102], we mention two connections between their work and existing works in vision. First, it is standard to convert their closure set problem into an equivalent s-t min cut, and we note that the corresponding min cut graph in their single-surface case happens to be a particular Ishikawa construction [71]. Their innovation can be thought of as building parallel Ishikawa constructions that influence one another. Second, there is a binary segmentation paper [152] that takes similar advantage of rays embedded in the image domain. Rather than unwrap the image domain and introduce geometric distortion of length/area, Veksler discretizes the rays and embeds them directly in the neighbourhood of a grid graph. One could implement multi-surface priors like Li et al. by extending Veksler's grid framework instead.

QPBO and approximations There are many multi-region models that are useful yet contain frustrated cycles. Even the simple 3-region interaction portrayed in (3.8) and the scene layout application are two examples where the ideal set of interactions cannot be optimized with a single graph cut. We can still formulate the (potentially NP-hard) energy and apply global methods like QPBO-P [18] or a reasonably fast approximation like QPBO-I [124]. QPBO-I can give good results on examples like Figures 3.7 and 3.11d, in only 1–5 subsequent 'improve' attempts. Figure 3.13 shows how QPBO-I succeeded in resolving the violated constraints from on the 'flatiron' image.

[4] In our framework, it is actually possible to create a spring-like attraction force between boundaries of i and j via opposing "i attracts j" and "j attracts i" interactions of large radius. However, the strength of this attraction is unfortunately coupled with surface regularization strength, leading to unwanted oversmoothing for most applications.

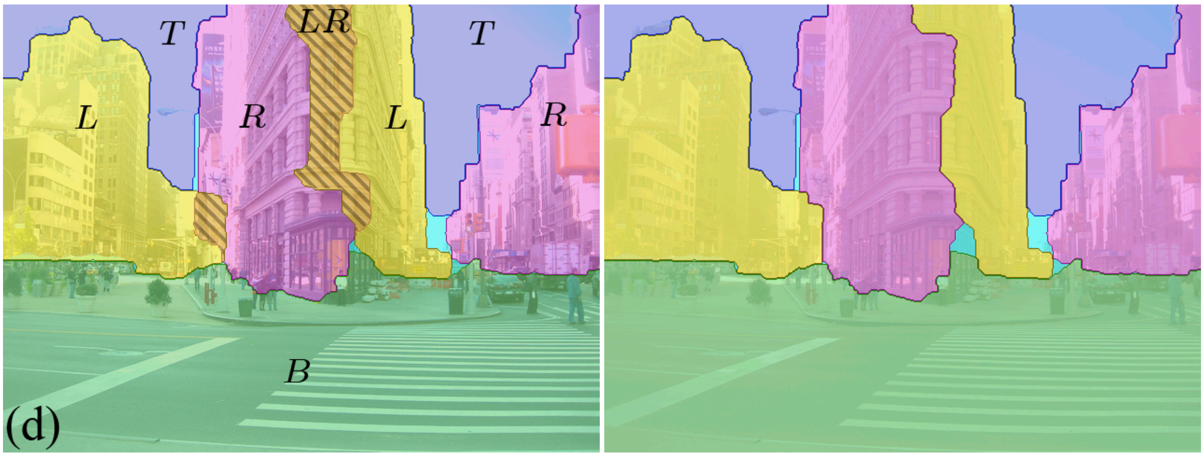


Figure 3.13: LEFT: Failure case from Figure 3.11d. RIGHT: Solution by QPBO-I where unwanted class labels BT, LR are prohibited.

Local search with multi-label moves Given a model that contains frustrated cycles among region layers, it may also be possible to design move-making algorithms that operate on subsets of regions, rather than relying on QPBO. This is in the spirit of "range-moves" [93, 151] where at each iteration we choose a large subset of interactions that can be trivially converted to submodular. Care must be taken to ensure that the energy of the labeling never increases, but large moves can be developed in this way. For example, we have verified that we can implement the vertical/horizontal moves in [105] using a simple "T excludes B" construction with special D_p and V^i based on the current labeling.

More generally we can minimize a multi-label energy $E(f)$ by defining a move space for the local search algorithm in Section 2.2. The main novelty over range-moves [151] will be (1) the geometric interpretation of long-range interactions and (2) our attempt to flip the meaning of binary variables to eliminate non-submodular interactions, resulting in a larger move space. Let us define the region subsets upon which submodular moves can be made:

$$\begin{aligned}\mathcal{S} = \{\, L \subseteq \mathcal{L} \;:\; & L \text{ is free of frustrated cycles and no} \\ & \text{superset of } L \text{ is free of frustrated cycles} \,\}.\end{aligned}$$

The set of all possible moves with respect to current labeling $\hat{f}$ can be defined as

$$\mathcal{M}(\hat{f}) = \bigcup_{L\in\mathcal{S}} \mathcal{M}^L(\hat{f}) \quad \text{where} \quad \mathcal{M}^L(\hat{f}) = \{\, f \;:\; f_p \neq \hat{f}_p \;\Rightarrow\; f_p, \hat{f}_p \in L \,\}. \tag{3.15}$$

Each set of regions $L \in \mathcal{S}$ has no frustrated cycles among its interactions. This means we can always find a binary 'flipping' such that there is a one-to-one correspondence between elements of $L = \{\ell_1, \ldots, \ell_k\}$ and feasible binary assignments $\mathbf{x}_p = (x_p^1, \ldots, x_p^k)$ where $k = |L|$. The flipping allows interaction terms W_{pq}^{ij} are submodular, and so we can compute an optimal move on subset L in a single graph cut using the constructions found throughout this chapter.

3.5 Conclusions and Future Work

With our multi-region framework, not only can more difficult objects now be segmented, but designing tractable models is also quite easy. The main ideas were to keep a separate appearance model for each spatially distinct region, and to allow geometric priors between region boundaries. Along the way, we discussed many parallels between the works of Li *et al.* [102], Ishikawa [71] and Veksler [152], and we hope these comments were helpful. Our experiments suggest that more robust medical segmentation tools could be designed around these ideas.

There are many other applications that can potential be revisited with these ideas in mind. Particularly promising are a more sophisticated concept of shape priors [50, 155] and topological constraints [153], but also ratio minimization [84], EM-style algorithms like GrabCuts [123], and combining pictorial structures with segmentation [44, 81]. Complex objects can be modeled by a hierarchy of nested regions that interact, with each region potentially driven by different data.

Finally, we note that there has been much past success in transferring ideas from discrete optimization into continuous settings, *e.g.* [118, 112]. We hypothesize that some of the ideas discussed in this chapter may also apply in continuous settings.

Chapter 4

Energies with Label Costs

In a labeling problem we are given a set of observations $\mathcal{P}$ (pixels, features, data points) and a finite set of labels $\mathcal{L}$ (categories, geometric models, disparities). The goal is to assign each observation $p \in \mathcal{P}$ a label $f_p \in \mathcal{L}$ such that the joint labeling f minimizes some objective function $E(f)$.

In computer vision and machine learning the available data is usually ambiguous, unreliable, or simply insufficient as to make direct inference possible. To explain such data we must incorporate biases and assumptions into our models, ideally a form of high-level reasoning or, more commonly, low-level *regularization* such as those discussed in Chapter 1. However, even low-level regularizers (biases) often make the corresponding inference problem NP-hard. Our work is about how to effectively optimize energies with two such regularizers: a preference for fewer unique labels in the solution (*label costs*), and a preference for spatial smoothness (*smooth costs*). Figures 4.1, 4.2, and 4.3 suggest how these criteria cooperate to give clean results.

The work in this chapter was initially published in the 2010 *Conference on Computer Vision and Pattern Recognition* (CVPR) [38] and subsequently expanded in the *International Journal of Computer Vision* (IJCV) [39].

4.1 Some Useful Regularizers

Regularization combining smoothness and label costs has a long history in vision going back to well known papers by Leclerc [98], Zhu & Yuille [166], and many others. Until recently, however, label costs could not be optimized by the powerful combinatorial algorithms popular in computer vision. The main contributions of our work (originally reported in [38]) are as follows. We are first to describe a general label cost functional ($\star$) that depends on a specific subset of used labels, rather than on a number of labels. Moreover, we propose several combinatorial optimization algorithms with guaranteed optimality bounds for minimizing energies combining data costs, smooth costs, and label costs.

Label costs Start by considering a basic (unregularized) energy $E(f) = \sum_p D_p(f_p)$, where optimal f_p can be determined trivially by minimizing over independent 'data costs'. Suppose, however, that we wish to explain the observations using as few unique labels as necessary. We

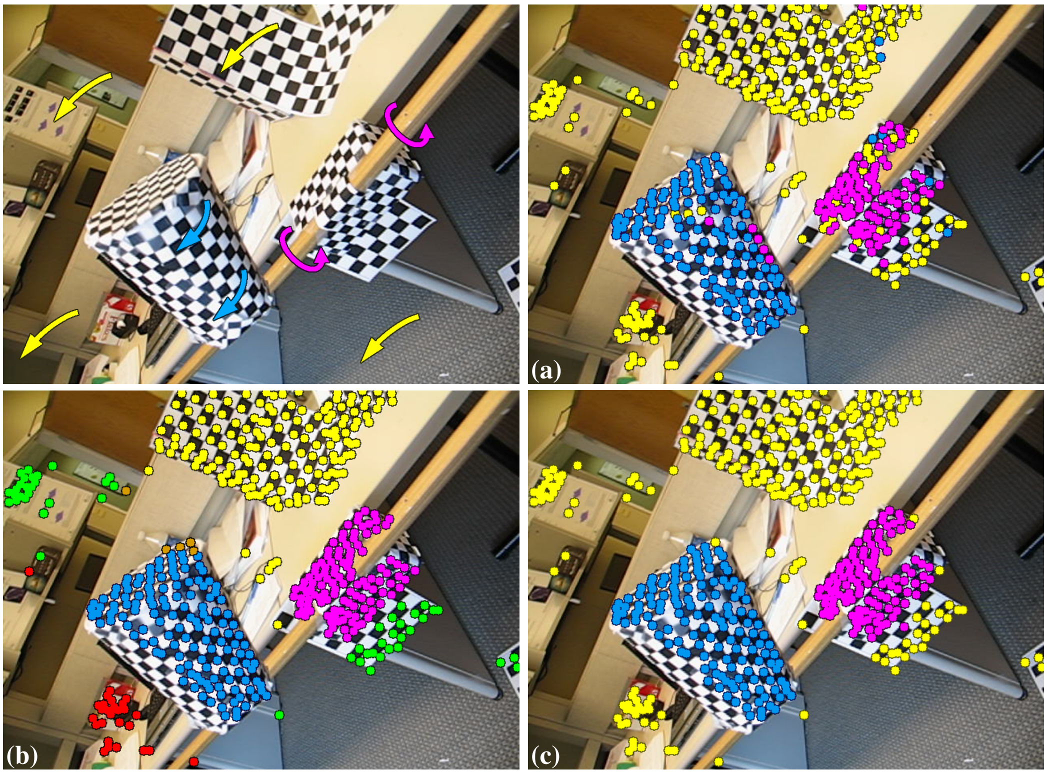


Figure 4.1: Motion segmentation on the 1RT2RCR sequence [145]. Energy (4.1) finds 3 dominant motions (a) but labels many points incorrectly. Energy (4.2) gives coherent segmentations (b) but finds redundant motions. Our energy combines the best of both (c).

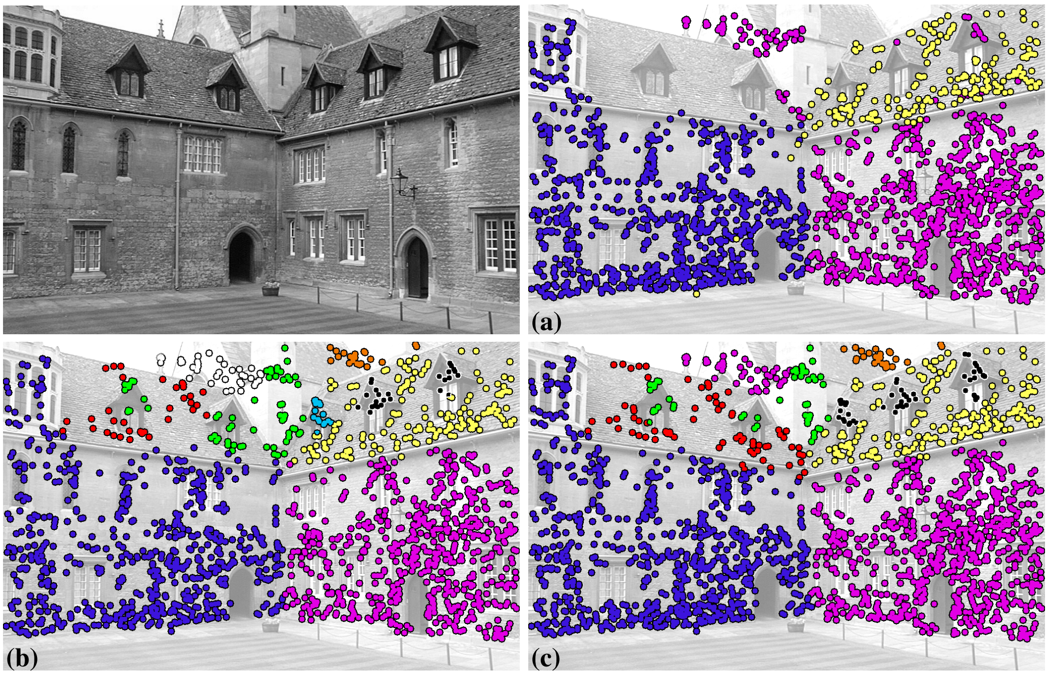


Figure 4.2: Planar homography detection on VGG (Oxford) Merton College 1 image (right view). Energy (4.1) finds reasonable parameters for only the strongest 3 models shown in (a), and still assigns a few incorrect labels. Energy (4.2) finds reasonable clusters (b) but fits 9 models, some of which are redundant (nearly co-planar). Our energy ($\star$) finds both good parameters and labels (c) for 7 models.

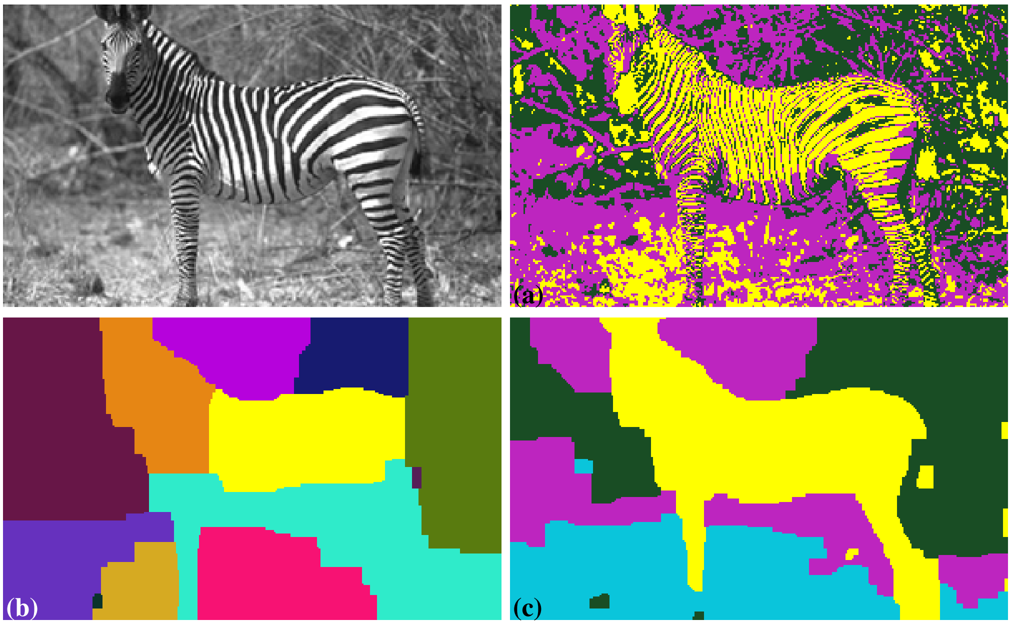


Figure 4.3: Unsupervised segmentation using histogram models. Energy (4.1) clusters in colour space, so segments (a) are incoherent. Energy (4.2) clusters over pixels and must either over-segment or over-smooth (b), just as in [164]. Our energy ($\star$) balances these criteria (c) and corresponds to Zhu & Yuille [166] for segmentation.

can introduce *label costs* into $E(f)$ to penalize each unique label that appears in f:

$$E(f) = \sum_{p \in \mathcal{P}} D_p(f_p) + \sum_{l \in \mathcal{L}} H(l) \cdot \delta_l(f) \tag{4.1}$$

where $H(l)$ is the non-negative label cost of label l, and $\delta_l(\cdot)$ is the corresponding indicator function

$$\delta_l(f) \overset{\text{def}}{=} \begin{cases} 1 & \exists p : f_p = l \\ 0 & \text{otherwise.} \end{cases}$$

Again, energy (4.1) balances the individual preferences of variables (the data costs) against the global preference to have rely on fewer unique labels (the label costs). It turns out that this formulation is equivalent to the well-studied *uncapacitated facility location* (UFL) problem, which we review in Section 4.3.5. For example, in computer vision, Li [101] recently posed multi-body motion estimation in terms of UFL. For multi-model fitting, each label corresponds to a candidate model and label costs penalize overly-complex models, preferring to explain the data with fewer, cheaper labels (see Figure 4.1a).

Smooth costs Spatial smoothness is a standard regularizer in computer vision. The idea here is that groups of observations are often known *a priori* to be positively correlated, and should thus be encouraged to have similar labels. Neighbouring image pixels are a classic example of this. Such pairwise priors can be expressed by the energy

$$E(f) = \sum_{p \in \mathcal{P}} D_p(f_p) \; + \sum_{pq \in \mathcal{N}} V_{pq}(f_p, f_q) \tag{4.2}$$

where each V_{pq} penalizes $f_p \neq f_q$ in some manner. If each V_{pq} defines a metric, then minimizing (4.2) is known as the *metric labeling* problem [24, 78] and can be optimized effectively with the α-expansion algorithm.

This regularizer prefers spatially coherent segmentations, but has no incentive to combine non-adjacent segments and thus a tendency to suggest redundant labels in multi-model fitting (see Figure 4.1b). Still, spatial smoothness priors are important for a wide array of vision applications.

Our combined energy We propose a discrete energy that essentially combines the UFL and metric labeling problems.

$$E(f) = \overbrace{\sum_{p\in\mathcal{P}} D_p(f_p)}^{\text{data cost}} + \overbrace{\sum_{pq\in\mathcal{N}} V_{pq}(f_p, f_q)}^{\text{smooth cost}} + \overbrace{\sum_{L\subseteq\mathcal{L}} H(L)\cdot\delta_L(f)}^{\text{label cost}} \qquad (\star)$$

where the indicator function $\delta_L(\cdot)$ is now defined on label subset L as

$$\delta_L(f) \overset{\text{def}}{=} \begin{cases} 1 & \exists p : f_p \in L \\ 0 & \text{otherwise.} \end{cases}$$

Our energy actually generalizes label costs to label *subset* costs $H(L)$, where we use notation for per-label costs $H(l)$ to mean $H(\{l\})$ throughout. Energy ($\star$) balances two demonstrably important regularizers, as suggested by Figure 4.1c. Figures 4.2 and 4.3 show other vision applications where our combined label cost energy makes sense.

4.2 Related work

A number of recent publications have relied on label costs in some form. For example, in [38] we proposed our subset costs in ($\star$) as a form of *co-occurrence cost* in object recognition. This application was thoroughly and independently developed by Ladický *et al.* [94], also within an α-expansion framework but with a heuristic extension; see Section 4.8 for discussion. Others have independently proposed label cost energies for specific applications. For example, we learned from personal correspondence that John Winn developed an extension of α-expansion to *instance cost* potentials in 2004 that only appeared as part of a supervised part-based object recognition framework [67], though his approach to deriving an algorithm is quite different from ours[1]. Special case energy (4.1) corresponds to objective functions studied in vision by Torr [144] and in a number of independent later works for specific applications [101, 97, 9]. Our combined energy ($\star$) has recently been extended to convex continuous *total variation* (TV) formulations [163].

Label costs can be viewed as a special case of other global interactions recently studied in vision, for example by Werner [159] and Woodford *et al.* [160]. Werner proposed a cutting plane algorithm to make certain high-order potentials tractable in an LP relaxation framework. The algorithm is very slow but much more general, and he demonstrates global *class*

[1] In [67] the algorithm is briefly described on page 6 and mixes binary and multi-label variables in a way such that we are unsure of the exact method of implementation/proof, but the goal is clearly analogous to a special case of our extended α-expansion for energy ($\star$).

size constraints for enforcing simple marginal statistics in image segmentation. Our potential $H(l) \cdot \delta_l(f)$ corresponds to a soft constraint that the number of variables taking label l be zero; this cost is concave w.r.t. the number of variables taking l. Woodford *et al.* optimize energies involving marginal statistics and they call these *Marginal Probability Fields* (MPFs). They focus on a number of hard cases with convex costs and propose specialized (but slow) algorithms based on *dual decomposition*.

This chapter studies label costs from a general perspective, including discussion of multiple algorithms, optimality bounds, extensions, and fast special cases. Our work on these algorithms was inspired by an array of generic model-fitting applications in vision that benefit from label costs: geometric model fitting [144], rigid motion estimation [101, 145], MDL-based segmentation [166], finite mixture models [15]. This chapter presents a number of synthetic and real examples illustrating generic applications for the label costs and evaluating the proposed optimization techniques.

Chapter outline Section 4.3 presents our extension to α-expansion and corresponding optimality bounds. We also analyze fast UFL heuristics for a special case of ($\star$) without smooth costs. Section 4.4 describes a multi-model fitting algorithm based on our energy, and Section 4.5 discusses connections to standard *expectation maximization* (EM) and K-means algorithms. Section 4.6 details our experiments illustrating generic applications in vision. Section 4.7 empirically compares a number of alternative combinatorial optimization algorithms applicable to label cost energies. Besides the extended version of α-expansion designed specifically for energy ($\star$), we tested a number of alternative methods based on standard α-expansion [24] for (4.2) with additional heuristics addressing the label costs term. Section 4.8 discusses applications of high-order label costs, more related works, and possible extensions.

4.3 Fast Algorithms to Minimize Label Costs

Our main technical contribution is to extend the well-known α-expansion algorithm [24] to incorporate label costs at each expansion (Section 4.3.1) and prove new optimality guarantees (Section 4.3.3). Section 4.3.5 reviews known results for the 'easy' case (4.1) with only data and per-label costs.

4.3.1 α-expansion with label costs

Minimizing the multi-label energy ($\star$) is NP-hard in general for $|\mathcal{L}| \geq 3$. The α-expansion algorithm [24] maintains a current labeling f' and iteratively 'moves' to a better one until no improvements can be made. At each iteration, some label $\alpha \in \mathcal{L}$ is chosen and variables f_p are simultaneously given a *binary* choice to either stay as $f_p = f'_p$ or switch to $f_p = \alpha$. This key step (line **4** below) is called *expansion* because label α is given a chance to grow arbitrarily. If each V_{pq} is a metric [24], the best possible expansion move can be computed efficiently by a single graph cut. We now describe the binary expansion step in sufficient detail for understanding label costs. For a more complete description of the α-expansion algorithm, see Section 2.2.

Let labeling $f = (f_1, \ldots, f_n)$ and let f^α denote a feasible α-expansion w.r.t. current labeling f'. Using subset of variables $\mathcal{P}_\alpha = \{p \in \mathcal{P} : f'_p \neq \alpha\}$, the possible labelings f^α can be expressed one-to-one with binary indicator variables $\mathbf{x} = (x_p)_{\mathcal{P}_\alpha}$ using relation

$$\begin{array}{lll} x_p = 0 \iff f^\alpha_p = f'_p & & \\ x_p = 1 \iff f^\alpha_p = \alpha & & \forall p \in \mathcal{P}_\alpha. \end{array} \tag{4.3}$$

Let $E^\alpha(\mathbf{x})$ denote the binary energy that encodes all possible α-expansion moves (4.3) relative to f'. The α-expansion algorithm computes an optimum $\mathbf{x}^*$ for E^α and thereby f^α, by solving a single s-t min cut problem.

As an illustrative example, suppose multi-label energy $E(f)$ is such that the optimal expansion with respect to labeling f' is f^α:

$$\begin{array}{rcl} f' = \begin{array}{|c|c|c|c|c|c|}\hline \beta & \alpha & \gamma & \gamma & \beta & \beta \\ \hline\end{array} & \rightarrow & \begin{array}{|c|c|c|c|c|c|}\hline \alpha & \alpha & \alpha & \gamma & \beta & \beta \\ \hline\end{array} = f^\alpha \\ & & \begin{array}{|c|c|c|c|c|c|}\hline 1 & \underline{1} & 1 & 0 & 0 & 0 \\ \hline\end{array} = \mathbf{x}^* \end{array} \tag{4.4}$$

where $\underline{1}$ means x_2 is fixed to 1. Here only f_1 and f_3 changed to label α while the rest preferred to keep their labels. The α-expansion algorithm iterates the above binary step until finally $\mathbf{x}^*_p = 0$ wherever $f'_p \neq \alpha$ for all possible $\alpha \in \mathcal{L}$.

Encoding label costs The energy in example (4.4) was such that f_5 and f_6 preferred to stay as label β rather than switch to α. Suppose we introduce a cost $H(\beta) > 0$ that is added to $E(f)$ if and only if there exists some $f_p = \beta$. The binary energy for an expansion move must encode a potential reward of $H(\beta)$ for replacing all $f'_p = \beta$ with label α. If $H(\beta)$ is large enough, the optimal expansion move for our small example would affect f_5 and f_6:

$$\begin{array}{rcl} f' = \underset{1 \qquad\quad\; 5 \;\; 6}{\begin{array}{|c|c|c|c|c|c|}\hline \beta & \alpha & \gamma & \gamma & \beta & \beta \\ \hline\end{array}} & \rightarrow & \begin{array}{|c|c|c|c|c|c|}\hline \alpha & \alpha & \alpha & \gamma & \alpha & \alpha \\ \hline\end{array} = f^\alpha \\ & & \begin{array}{|c|c|c|c|c|c|}\hline 1 & \underline{1} & 1 & 0 & 1 & 1 \\ \hline\end{array} = \mathbf{x}^* \end{array} \tag{4.5}$$

Our main algorithmic contribution is a way to encode such label costs into the expansion step and thereby encourage solutions that use fewer labels.

Energy $E^\alpha(\mathbf{x})$, when expressed as a multilinear polynomial, is a sum of linear and quadratic terms over $\mathbf{x}$. For the specific example (4.5), we can encode cost $H(\beta)$ in E^α by simply adding $H(\beta) - H(\beta)x_1x_5x_6$ to the binary energy. Because this specific term is cubic and $H(\beta) \geq 0$, it can be optimized by a single graph cut using the construction in [87].

To encode general label costs for arbitrary $L \subseteq \mathcal{L}$ and f', we must optimize the modified expansion energy

$$E^\alpha_h(\mathbf{x}) = E^\alpha(\mathbf{x}) + \sum_{\substack{L \subseteq \mathcal{L} \\ L \cap \mathcal{L}' \neq \emptyset}} \Big(H(L) - H(L) \prod_{p \in \mathcal{P}_L} x_p \Big) + C^\alpha(\mathbf{x}) \tag{4.6}$$

where set $\mathcal{L}'$ contains the unique labels in the current labeling f', and set $\mathcal{P}_L = \{\, p : f'_p \in L\}$. Term C^α simply accounts for the case when $\alpha \notin \mathcal{L}'$ and so the introduction of label α must itself be penalized; this term is discussed shortly, but for now we focus on the 'reward' terms in (4.6).

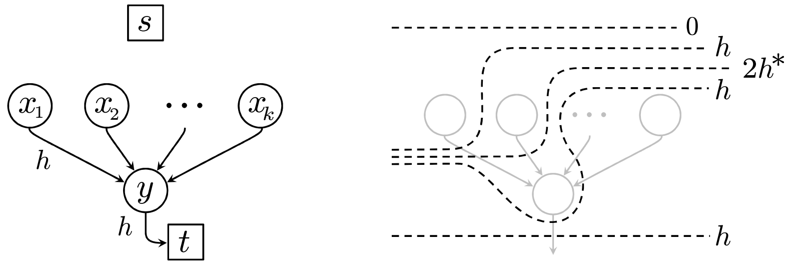


Figure 4.4: LEFT: Graph construction that encodes $h - hx_1x_2 \cdots x_k$ when we define $x_p = 1 \Leftrightarrow p \in T$ where T is the sink side of the cut. RIGHT: In a minimal s-t cut, the subgraph contributes cost either 0 (all $x_p = 1$) or h (otherwise). A cost greater than h (*e.g.*$*$) cannot be minimal because setting $y = 0$ cuts only one arc.

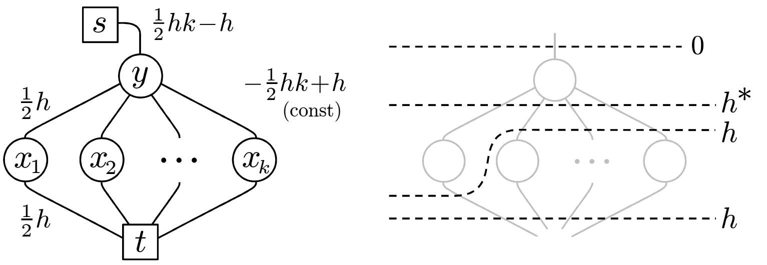


Figure 4.5: The alternate *undirected* graph construction corresponding to Figure 4.4 may be easier to understand. The weights are found by reparameterizing (4.8) such that $\bar{x}y$ and $x\bar{y}$ terms receive identical coefficients. Cut $*$ is not minimal with respect to auxiliary variable y.

Each product term in (4.6) adds a higher-order clique $\mathcal{P}_L$ beyond the standard α-expansion energy $E^\alpha(\mathbf{x})$. Freedman and Drineas [49] generalized the graph construction of [87] to handle terms $c\prod_p x_p$ of arbitrary degree when $c \le 0$. This means we can transform each product seen in (4.6) into a sum of quadratic and linear terms that graph cuts can still optimize globally. The transformation for a particular label subset $L \subseteq \mathcal{L}$ with $|\mathcal{P}_L| \ge 3$ is

$$-H(L)\prod_{p\in\mathcal{P}_L} x_p = H(L)\cdot \min_{y_L\in\{0,1\}} \Big[(|\mathcal{P}_L|-1)y_L - \sum_{p\in\mathcal{P}_L} x_p y_L\Big] \tag{4.7}$$

where y_L is an auxiliary variable that must be optimized alongside $\mathbf{x}$ whenever $H(L) > 0$. Since each $x_p y_L$ term has non-positive coefficient, the corresponding binary energy is submodular [18] and can be minimized by a single graph cut (Section 2.1).

To encode the potential (4.7) into an s-t min-cut graph construction, we reparameterize the right-hand side such that each quadratic monomial has exactly one complemented variable (*e.g.* $x\bar{y}$) and non-negative coefficient (arc weight). The particular reparameterization we use is

$$-H(L) + H(L)\bar{y}_L + \sum_{p\in\mathcal{P}_L} H(L)\cdot \bar{x}_p y_L \tag{4.8}$$

where $\bar{x} = 1 - x$. Figure 4.4 shows the subgraph corresponding to (4.8) after cancelling the constant $-H(L)$ using (4.7).

Subgraphs of this type have been used in vision before, most notably the P^n Potts potentials of Kohli *et al.* [79]. Our indicator potentials $\delta_L(\cdot)$ are different in that, at the binary step (4.6), each clique $\mathcal{P}_L$ is determined *dynamically* from the current labeling f' and is not expressed as such in the original energy ($\star$). A P^n Potts potential can be represented by a combination of label subset costs but not the other way around. The idea is to apply 'regional' subset costs derived from the coefficients of the P^n Potts potential. Section 4.8 describes this transformation in detail.

A final detail for α-expansion is the case when label α was not present in the current labeling f'. The corrective term C^α in (4.6) incorporates the label costs for α itself:

$$C^\alpha(\mathbf{x}) = \sum_{\substack{L \subseteq \mathcal{L} \setminus \mathcal{L}' \\ \alpha \in L}} \Big(H(L) - H(L) \prod_{p \in \mathcal{P}} \bar{x}_p \Big). \tag{4.9}$$

The binary energy should only pay the label costs where $\alpha \in L$ if label α was introduced during the expansion move. If we find that $\mathbf{x}^* = 0$ then we know that (a) label α did not appear in f' and (b) it was also not worth introducing α in the expansion move, so $f^\alpha = f'$. The term (4.9) can be encoded by a subgraph analogous to Figure 4.4, but the following is more efficient: first compute optimal $\mathbf{x}^*$ for (4.6) without considering C^α, then explicitly add it to $E^\alpha_h(\mathbf{x}^*)$ if $\mathbf{x}^* \neq \mathbf{0}$, and reject the expansion if the energy would increase.

4.3.2 $\alpha\beta$-swap with label costs

For the $\alpha\beta$-swap algorithm we denote the binary energy for any particular $\alpha\beta$-swap by $E^{\alpha\beta}(\mathbf{x})$. The terms in this energy are slightly different from E^α used in α-expansion, but we can also extend $\alpha\beta$-swap to incorporate label costs by adding similar high-order terms to $E^{\alpha\beta}$. For a complete description of the $\alpha\beta$-swap algorithm, see Section 2.2.

Let f' denote the current labeling and let $f^{\alpha\beta}$ denote a feasible $\alpha\beta$-swap move with respect to f'. The set of pixels that can swap their labels is $\mathcal{P}_{\alpha\beta} = \{p \in \mathcal{P} : f'_p \in \{\alpha, \beta\}\}$. The possible swap moves $f^{\alpha\beta}$ correspond one-to-one with binary indicator variables $\mathbf{x} = (x_p)_{p \in \mathcal{P}_{\alpha\beta}}$ as

$$\begin{array}{lcll} x_p = 0 & \Longleftrightarrow & f^{\alpha\beta}_p = \alpha & \\ x_p = 1 & \Longleftrightarrow & f^{\alpha\beta}_p = \beta & \quad \forall p \in \mathcal{P}_{\alpha\beta}. \end{array}$$

Let $\mathcal{L}'_{\alpha\beta} = \mathcal{L}' \setminus \{\alpha, \beta\}$, which is the set of labels used in f' by pixels *not* involved in the swap. Then the binary expansion energy *with label costs* can be written as

$$E^{\alpha\beta}_h(\mathbf{x}) = E^{\alpha\beta}(\mathbf{x}) + \sum_{\substack{L \subseteq \mathcal{L} \setminus \mathcal{L}'_{\alpha\beta} \\ \alpha \in L, \beta \notin L}} \Big(H(L) - H(L) \prod_{p \in \mathcal{P}_{\alpha\beta}} x_p \Big) + \sum_{\substack{L \subseteq \mathcal{L} \setminus \mathcal{L}'_{\alpha\beta} \\ \alpha \notin L, \beta \in L}} \Big(H(L) - H(L) \prod_{p \in \mathcal{P}_{\alpha\beta}} \bar{x}_p \Big) \tag{4.10}$$

However, label costs can be trivially incorporated into $\alpha\beta$-swap by a test-and-reject approach similar to the way $C^\alpha(\mathbf{x})$ was handled for α-expansion. First attempt a standard swap move by minimizing $E^{\alpha\beta}(\mathbf{x})$ to compute $\mathbf{x}^*$. Then compare $E^{\alpha\beta}_h(\mathbf{x}^*)$ to the energy values $E^{\alpha\beta}_h(\mathbf{0})$ and $E^{\alpha\beta}_h(\mathbf{1})$, *i.e.* when the swap is all-α or all-β respectively. Finally, apply the move with minimum energy.

4.3.3 Approximation guarantees of α-expansion

In what follows we assume that energy ($\star$) is configured[2] so that $D_p \geq 0$, V_{pq} is a metric [24], and thus $E(f) \geq 0$.

Theorem 4.1. *If f^* is a global minimum of energy ($\star$) and $\hat{f}$ is a local minimum w.r.t. α-expansion then*

$$E(\hat{f}) \;\leq\; (2c + c')E(f^*) \;+ \sum_{L \subset \mathcal{L}} H(L) \tag{4.11}$$

where $c = \max_{pq \in \mathcal{N}} \left(\frac{\max_{\alpha \neq \beta \in \mathcal{L}} V_{pq}(\alpha,\beta)}{\min_{\gamma \neq \zeta \in \mathcal{L}} V_{pq}(\gamma,\zeta)} \right)$, $c' = \max_{\substack{L \subset \mathcal{L} \\ H(L) > 0}} |L| - 1.$

Proof of Theorem 4.1. The proof idea follows Theorem 6.1 of [24]. Let us fix some $\alpha \in \mathcal{L}$ and define

$$\mathcal{P}_\alpha \overset{\text{def}}{=} \left\{ p \in \mathcal{P} : f_p^* = \alpha \right\}. \tag{4.12}$$

We can produce a labeling f^α within one α-expansion move from $\hat{f}$ as follows:

$$f_p^\alpha = \begin{cases} \alpha & \text{if } p \in \mathcal{P}_\alpha \\ \hat{f}_p & \text{otherwise.} \end{cases} \tag{4.13}$$

Since $\hat{f}$ is a local optimum w.r.t. expansion moves we have

$$E(\hat{f}) \leq E(f^\alpha). \tag{4.14}$$

Let $E(\cdot)|_\mathcal{S}$ denote a restriction of the summands of energy ($\star$) to only the following terms:

$$E(f)|_\mathcal{S} = \sum_{p \in \mathcal{S}} D_p(f_p) + \sum_{pq \in \mathcal{S}} V_{pq}(f_p, f_q).$$

We separate the unary and pairwise terms of $E(f)$ via interior, exterior, and boundary sets with respect to pixels $\mathcal{P}_\alpha$:

$$\begin{array}{lllll} \mathcal{I}_\alpha = & \mathcal{P}_\alpha & \cup & \{ pq \in \mathcal{N} : p, q \in \mathcal{P}_\alpha \} \\ \mathcal{O}_\alpha = & \mathcal{P} \setminus \mathcal{P}_\alpha & \cup & \{ pq \in \mathcal{N} : p, q \notin \mathcal{P}_\alpha \} \\ \mathcal{B}_\alpha = & & & \{ pq \in \mathcal{N} : p \in \mathcal{P}_\alpha, q \notin \mathcal{P}_\alpha \}. \end{array}$$

The following facts now hold:

$$E(f^\alpha)|_{\mathcal{I}_\alpha} \;=\; E(f^*)|_{\mathcal{I}_\alpha} \tag{4.15}$$

$$E(f^\alpha)|_{\mathcal{O}_\alpha} \;=\; E(\,\hat{f}\,)|_{\mathcal{O}_\alpha} \tag{4.16}$$

$$E(f^\alpha)|_{\mathcal{B}_\alpha} \leq cE(f^*)|_{\mathcal{B}_\alpha}. \tag{4.17}$$

Inequality (4.17) follows from the fact that $V(f_p^\alpha, f_q^\alpha) \leq cV(f_p^*, f_q^*)$ for any $pq \in \mathcal{B}_\alpha$.

[2] Adding an arbitrary constant to $D_p(\cdot)$ or $V_{pq}(\cdot,\cdot)$ does not affect the optimal labeling, so finite costs can always be made non-negative.

Let E_H denote the label cost terms of energy E. Using (4.15), (4.16) and (4.17) we can rewrite (4.14) as

$$\begin{aligned} & E(\hat{f}\,)|_{\mathcal{I}_\alpha} + \ E(\hat{f}\,)|_{\mathcal{B}_\alpha} + E_H(\hat{f}\,) && (4.18)\\ \le\ & E(f^\alpha)|_{\mathcal{I}_\alpha} + \ E(f^\alpha)|_{\mathcal{B}_\alpha} + E_H(f^\alpha) && (4.19)\\ \le\ & E(f^*)|_{\mathcal{I}_\alpha} + cE(f^*)|_{\mathcal{B}_\alpha} + E_H(f^\alpha) && (4.20)\end{aligned}$$

Depending on $\hat{f}$ we can bound $E_H(f^\alpha)$ by

$$E_H(f^\alpha) \ \le\ E_H(\hat{f}) \ + \sum_{\substack{L\subseteq\mathcal{L}\setminus\hat{\mathcal{L}}\\ \alpha\in L}} H(L) \tag{4.21}$$

where set $\hat{\mathcal{L}}$ contains only the unique labels in $\hat{f}$.

To bound the total energy we sum expressions (4.18) and (4.20) over all labels $\alpha \in \mathcal{L}^*$ to arrive at the following:

$$\sum_{\alpha\in\mathcal{L}^*}\Big(E(\hat{f}\,)|_{\mathcal{I}_\alpha} + E(\hat{f}\,)|_{\mathcal{B}_\alpha}\Big) \le \sum_{\alpha\in\mathcal{L}^*}\Big(E(f^*)|_{\mathcal{I}_\alpha} + cE(f^*)|_{\mathcal{B}_\alpha}\Big) + \sum_{L\subseteq\mathcal{L}\setminus\hat{\mathcal{L}}} H(L)\cdot|L\cap\mathcal{L}^*|. \tag{4.22}$$

Observe that, for every $pq \in \mathcal{B} = \bigcup_{\alpha\in\mathcal{L}}\mathcal{B}_\alpha$, the term $V_{pq}(\hat{f}_p, \hat{f}_q)$ appears twice on the left side of (4.22), once for $\alpha = f_p^*$ and once for $\alpha = f_q^*$. Similarly every $V(f_p^*, f_q^*)$ appears $2c$ times on the right side of (4.22). Therefore equation (4.22) can be rewritten as

$$\begin{aligned} E(\hat{f}) \ \le\ & E(f^*) + (2c-1)E_V(f^*) - E(\hat{f})|_{\mathcal{B}} && (4.23)\\ & + E_H(\hat{f}) - E_H(f^*) + \sum_{L\subseteq\mathcal{L}\setminus\hat{\mathcal{L}}} H(L)\cdot|L\cap\mathcal{L}^*|. \end{aligned}$$

We have now derived an *a posteriori* bound (4.23) with respect to any particular $\hat{f}$ and f^*. Observe that the second line of (4.23) involving label costs is equal to

$$\sum_{\substack{L\subseteq\mathcal{L}\setminus\mathcal{L}^*\\ L\cap\hat{\mathcal{L}}\neq\emptyset}} H(L) \ + \sum_{\substack{L\subseteq\mathcal{L}\setminus\hat{\mathcal{L}}\\ L\cap\mathcal{L}^*\neq\emptyset}} H(L)\cdot(|L\cap\mathcal{L}^*| - 1)\,. \tag{4.24}$$

The right-hand sum includes label costs that f^* pays but that $\hat{f}$ does not. Expression (4.24) can be bounded by

$$\begin{aligned} &\le \sum_{\substack{L\subseteq\mathcal{L}\setminus\mathcal{L}^*\\ L\cap\hat{\mathcal{L}}\neq\emptyset}} H(L) \ + \ c'\cdot\sum_{\substack{L\subseteq\mathcal{L}\setminus\hat{\mathcal{L}}\\ L\cap\mathcal{L}^*\neq\emptyset}} H(L), \quad \text{where } c' = \max_{\substack{L\subseteq\mathcal{L}\\ H(L)>0}} |L| - 1 && (4.25)\\ &\le \sum_{L\subseteq\mathcal{L}\setminus\mathcal{L}^*} H(L) \ + \ c'E_H(f^*) && (4.26)\end{aligned}$$

where c' is understood to be zero if all $H(L) = 0$. Combining (4.23) with (4.26) and using the fact that $c'E_H(f^*) \geq 0$ we can simplify the bound as

$$E(\hat{f}) \leq \; E(f^*) \; + \; (2c-1)E_V(f^*) \; + \; c'E_H(f^*) \; + \sum_{L \subseteq \mathcal{L} \setminus \mathcal{L}^*} H(L) \tag{4.27}$$

$$\leq (2c+c')E(f^*) \; + \; \sum_{L \subset \mathcal{L}} H(L). \tag{4.28}$$

Assuming $D_p \geq 0$ we have *a priori* bound (4.28). ■

These bounds suggest the following properties in practice:

- if label costs are modest we inherit an approximation guarantee comparable to α-expansion,
- if label costs are arbitrarily large the bound is poor, and
- if the optimal solution includes label costs defined over large subsets then the bound worsens.

Poor local minima are caused by the fact that α-expansion allows only one label to expand at a time. Performing expansions in greedy order (rather than arbitrary order) may help empirically, but a hardness result of Feige [43] still applies to our problem (discussed in Section 4.3.5).

For discussion, we note that (4.11) follows from a more general *a posteriori* bound (4.27) that does not assume $D_p \geq 0$. Bound (4.27) holds for all $\hat{f}$ and f^*, so the approximation error is determined by the minimum of the three additive terms above over all global optima f^*. The additive bound (4.27) is informative in a way that the familiar multiplicative bound $E(\hat{f}) \leq 2cE(f^*)$ for α-expansion is not. To see why, consider that the multiplicative bound for α-expansion is only tight when the total data cost $E_D(f^*) = 0$, and does not even hold for $E_D(f^*) < 0$. Yet, biasing the data costs with some $D'_p(\cdot) := D_p(\cdot) + \epsilon_p$ for arbitrary constant ϵ_p affects neither the global optima nor the optimal expansion moves. The α-expansion algorithm is indifferent to ϵ_p, and this property distinguishes it from the *isolation heuristic* algorithm for multi-terminal cuts [33]. The isolation heuristic is applicable to metric labeling when V_{pq} are Potts interactions, also has multiplicative bound of 2, but can compute arbitrarily bad solutions to multi-label problems depending on ϵ_p. The comparative robustness of α-expansion is not reflected in the multiplicative bound.

Worst-case examples The simplified bound (4.11) describes the worst-case performance in special cases, but bound (4.27) is tight more generally. The table below describes a worst-case problem instance with $\mathcal{P} = \{p, q\}$ and $\mathcal{L} = \{\alpha, \beta, \gamma\}$. We also assume a label cost $H(\gamma) \geq 0$ and a Potts potential that penalizes $f_p \neq f_q$ with weight $w > 0$.

$$\begin{array}{r|c|c|} \text{data costs}\searrow & p & q \\ \hline \alpha & 0 & \infty \\ \hline \beta & \infty & 0 \\ \hline \gamma & w & w \\ \hline \end{array} \qquad \begin{array}{c} \text{label cost} \\ \swarrow \\ \boxed{h_\gamma} \end{array} \qquad \begin{array}{l} f^* = (\alpha, \beta) \\ \\ \hat{f} \;\; = (\gamma, \gamma) \end{array} \tag{4.29}$$

This example has global optimum $E(f^*) = w$ and so the local minimum $E(\hat{f}) = 2w + H(\gamma)$ is tight with respect to (4.11). Note that by adding positive ϵ_p to each $D_p(\cdot)$ our additive bound (4.27) remains tight, unlike the multiplicative bound.

More generally we can design bad local minima from the following n-variable problem structure. Let $a, b, h \geq 0$ be constants such that $a = h + w$ where w is still the weight of all Potts potentials. Let $\mathcal{N} = \{\{1,2\},\{3,4\},\ldots\}$ be the neighbour set for Potts potentials. The data costs and label costs in the table below have optimal labeling $f^* = \{1,\ldots,n\}$, yet labeling $\hat{f} = \{n+1, n+1, n+2, n+2, \ldots\}$ is a local minimum w.r.t. expansion moves. (A blank entry signifies $D_p = \infty$)

$$\begin{array}{ccc}
\text{data costs}\searrow & \overbrace{\hspace{8em}}^{n\text{ variables}} & \swarrow\text{label subset costs}
\end{array}$$

$$\begin{array}{r}
n\text{ labels}\left\{\vphantom{\begin{array}{c}0\\0\\0\\0\\0\\0\end{array}}\right. \\
\vphantom{\begin{array}{c}a\\a\\a\end{array}}
\end{array}
\begin{array}{|cccccc|}
\hline
0 & & & & & \\
 & 0 & & & & \\
 & & 0 & & & \\
 & & & 0 & & \\
 & & & & 0 & \\
 & & & & & 0 \\
\hline
a & a & & & & \\
 & & a & a & & \\
 & & & & a & a \\
\hline
\end{array}
\quad
\begin{array}{|c|}
\hline
\\
\\
h \\
\\
\\
\\
\hline
b \\
\hline
b \\
\hline
b \\
\hline
\end{array}
\begin{array}{l}
\left.\vphantom{\begin{array}{c}0\\0\\0\\0\\0\\0\end{array}}\right\} E(f^*) = h + \tfrac{1}{2}nw \\
\left.\vphantom{\begin{array}{c}a\\a\\a\end{array}}\right\} E(\hat{f}) = na + \sum b
\end{array}
\tag{4.30}$$

We verify that $\hat{f}$ is generally tight for bound (4.27) as follows

$$\begin{aligned}
E(\hat{f}) = na + \textstyle\sum b &= nh + nw + \textstyle\sum b \\
&= E(f^*) + \tfrac{1}{2}nw + (n-1)h + \textstyle\sum b \\
&= E(f^*) + E_V(f^*) + c' E_H(f^*) + \textstyle\sum b.
\end{aligned}\tag{4.31}$$

The above is tight for (4.11) when $h = 0$ and nearly tight when $w = 0$ aside for one double-counted label cost h. This example demonstrates how high-order label costs in the optimal labeling can worsen the approximation.

4.3.4 Local label costs

We can generalize the concept of label costs by making them spatially localized. The label cost term in energy ($\star$) could be expressed more generally as

$$\sum_{P \subseteq \mathcal{P}} \sum_{L \subseteq \mathcal{L}} H_P(L) \cdot \delta_L(f_P) \tag{4.32}$$

where $\delta_L(f_P)$ is 1 if there exists $p \in P$ such that $f_p \in L$, and otherwise 0. Our basic energy ($\star$) is a special case that assumes $H_P(L) = 0$ for all non-global cliques $P \subsetneq \mathcal{P}$. Note that the fast test-and-reject approaches mentioned in Sections 4.3.1 and 4.3.2 are no longer feasible for this more general case.

Such potentials amount to *local* label cost terms. Local label costs and subset costs are useful together when labels belong to known categories with specific location priors, such as "pay a fixed penalty if any label from {*sky, cloud, sun*} appears in the bottom of an image." Local label costs are also crucial for the *hierarchical fusion* algorithm introduced in Chapter 5.

4.3.5 Energies with only per-label costs

In the absence of smooth costs and high-order label costs (*i.e.* $V_{pq} = 0$ and $H(L) > 0 \Rightarrow |L| = 1$) our energy reduces to a special case (4.1) known in operations research as the *uncapacitated facility location* (UFL) problem. The UFL problem assigns a facility (a label) to each client (a variable) such that the cost to clients is balanced against the cost of 'opening' facilities to serve them. Letting $\mathcal{L}$ be the set of facilities and $\mathcal{P}$ the set of clients, the standard integer programming formulation for UFL [30] is

$$\min \sum_{p\in\mathcal{P}} \left(\sum_{l\in\mathcal{L}} d_{pl} x_{pl} \right) + \sum_{l\in\mathcal{L}} h_l y_l \tag{4.33}$$

$$\text{s.t.} \sum_{l\in\mathcal{L}} x_{pl} = 1 \qquad \forall p \in \mathcal{P}, \tag{4.34}$$

$$x_{pl} \le y_l \qquad \forall p \in \mathcal{P}, l \in \mathcal{L}, \tag{4.35}$$

$$x_{pl}, y_l \in \{0,1\},$$

where d_pl is the cost of assigning client p to facility l, and $h_l \ge 0$ is the cost of opening facility l. Through equality (4.34), each solution to UFL corresponds to a labeling f via the assignment $x_{pl} = 1 \Leftrightarrow f_p = l$. Since (4.35) implies $y_l = \delta_l(f)$ at any minimum of (4.33), setting $D_p(l) = d_{pl}$ makes UFL equivalent to minimization problem (4.1).

In vision, the UFL problem has recently been applied to motion segmentation by Li [101] and by Lazic *et al.* [97]. Each facility represents a potential rigid motion, and each client is a correspondence that must be assigned to one motion. The goal is then to choose a good subset of motions, much like Figure 4.1a. Li optimizes the integer program corresponding to UFL by *linear programming (LP) relaxation*, then rounds fractional facility variables to $\{0,1\}$ in a straight-forward manner. Because general LP solvers are slow, this approach affords relatively few candidate models in practice. Li implements four application-specific heuristics to aggressively prune out candidate models before building an LP problem instance. Lazic *et al.* optimize the same energy using message-passing algorithms [97, 96]. More recently, Barinova *et al.* [9] used UFL to model a class of object-detection problems and used the same greedy algorithm as our concurrent work [38].

The general[3] UFL problem is NP-hard by simple reduction from SET-COVER. A hardness result for approximating SET-COVER by Feige [43] implies that UFL cannot be approximated better than $(1-\epsilon)\ln|\mathcal{P}|$ for $\epsilon > 0$ in polynomial time unless the complexity class NP is only slightly super-polynomial, *i.e.* $\mathrm{NP} \subseteq \mathrm{DTIME}[n^{O(\log\log n)}]$. This leads to the following observation about our bound for α-expansion in Section 4.3.3.

Observation 4.2. *Feige's hardness result [43] is evidence that no ϵ-approximation algorithm can exist for our general energy ($\star$), even when only per-label costs are used. In this sense, label costs are harder to optimize than are smooth costs.*

Kuehn & Hamburger [91] proposed a natural *greedy* algorithm where facilities are opened one at a time. Cornuejols *et al.* [29] showed that the greedy algorithm provides a constant-factor approximation bound, but only with respect to the gap between best and worst solutions; this bound is not informative when the range of costs involved are prohibitively large.

Hochbaum [64] later proposed a *set-greedy* algorithm that achieves a $\ln|\mathcal{P}|$-approximation regardless of the costs involved, which is optimal in the sense outlined by Feige. Hochbaum also showed that neither greedy nor set-greedy is strictly better than the other, and that the best choice depends on the problem instances at hand. We present the original greedy algorithm rather than the set-greedy algorithm.

We note that Hochbaum's algorithm is straight-forward but of a very different nature than α-expansion. It is interesting to ask whether there exists an algorithm that generalizes the approximation guarantees of both her set-greedy algorithm and of α-expansion for our general energy. We leave this for future work.

Greedy UFL In terms of our multi-label energy (4.1), the greedy UFL algorithm starts from an empty set of labels and greedily introduces one label at a time until no subsequent label would allow the overall cost to decrease. Once a label l is introduced, its cost $H(l)$ is assumed to be paid for regardless of subsequent steps. To express the greedy algorithm we introduce a function of label subsets $Z(S)$ where $S \subseteq \mathcal{L}$. The problem of minimizing $E(f)$ in (4.1) can then be rewritten as

$$\min_f E(f) = \min_{S \subseteq \mathcal{L}} Z(S) \tag{4.36}$$

$$\text{where } \; Z(S) \; = \sum_{p \in \mathcal{P}} \min_{l \in S} D_p(l) \; + \; \sum_{l \in S} H(l) \tag{4.37}$$

and $Z(\{\})$ is defined to be $+\infty$. The overall algorithm is described in pseudo-code below.

```
GREEDYUFL [91, 30]
$S := \{\}$
while exists $l \notin S$ such that $Z(S \cup \{l\}) < Z(S)$
    $j := \arg\min_{l \notin S} Z(S \cup \{l\}) - Z(S)$
    $S := S \cup \{j\}$
```

The greedy algorithm runs in $O(|\mathcal{L}|^2|\mathcal{P}|)$ time for label set $\mathcal{L}$ and variable set $\mathcal{P}$. Our C++ library implements GREEDYUFL and it is 5–20 times faster than α-expansion for energies of the form (4.1) while yielding similar results. Besides this classic heuristic, other greedy moves have been proposed for UFL such as the *greedy-interchange* and *dynamic programming* heuristics (see [29, 30] for a review).

Babayev [7] and Frieze [51] noted in 1974 that the set function $Z(S)$ is supermodular as a minimization problem. We state can state this formally by the following observation

Theorem 4.3 ([7, 51])**.** *The greedy UFL set function $Z(S)$ is supermodular, i.e. it satisfies*

$$Z(S \cup \{j, k\}) - Z(S \cup \{k\}) \geq Z(S \cup \{j\}) - Z(S). \tag{4.38}$$

Proof. Simply expand the left and right sides of equation (4.38) based on the definition of

[3] *Metric*-UFL is a special case that can be approximated to within a constant factor [134]. In our work we assume arbitrary costs $D_p(\cdot)$. Unfortunately, some papers refer to metric-UFL simply as UFL.

$Z(S)$ and we have

$$\begin{aligned} Z(S\cup\{j\}) - Z(S) &= \sum_{p\in\mathcal{P}} \min\{0, D_p(j) - \min_{l\in S} D_p(l)\} \;+\; H(j) && (4.39)\\ &\le \sum_{p\in\mathcal{P}} \min\{0, D_p(j) - \min_{l\in S\cup\{k\}} D_p(l)\} \;+\; H(j) && (4.40)\\ &= Z(S\cup\{j,k\}) - Z(S\cup\{k\}) && (4.41)\end{aligned}$$

Inequality (4.40) holds because clearly $\min_{l\in S} D_p(l) \ge \min_{l\in S\cup\{k\}} D_p(l)$. ■

The greedy bound for UFL by Cornuejols *et al.* [29] then follows from a general bound on minimizing supermodular functions by Nemhauser *et al.* [111]. Note that introducing a new label $j \notin S$ in step **3** does not consider the potential reward for eliminating labels from S once j is made available. This is in contrast to a j-expansion move with label costs, where introducing j may be beneficial because existing labels could be eliminated despite $Z(S\cup\{j\})$ not reflecting this in the classical algorithm. The 2-variable problem instance below illustrates this difference for some constants $a > 1,\ b > 0$. GREEDYUFL finds an arbitrarily poor energy of $(2 + a)b$ whereas α-expansion with label costs finds an energy of $3b$ regardless of initial labeling.

data costs ↘ ↙ label costs

$$\begin{array}{c|c|c|} & p & q \\ \hline \alpha & 0 & \infty \\ \hline \beta & \infty & 0 \\ \hline \gamma & 2b & b \\ \hline \end{array} \quad \begin{array}{|c|} \hline b \\ \hline 2b \\ \hline ab \\ \hline \end{array} \qquad \begin{array}{rl} \text{greedy} & \hat{f} = (\alpha,\gamma) \\ \alpha\text{-expansion} & \hat{f} = (\alpha,\beta) \end{array} \tag{4.42}$$

Our subset costs $H(L)$ suggest a generalization of the classic UFL problem to add *facility subset costs*. Each subset cost represents a *shared* setup cost for opening particular set of facilities, after which the individual facilities can be opened with their own costs $H(l)$ for $l \in L$. We call the new problem *UFL with subset costs* (UFL-S) and, in the language of energy minimization, we can express this as minimizing the energy

$$E(f) = \sum_{p\in\mathcal{P}} D_p(f_p) \;+ \sum_{L\subseteq\mathcal{L}} H(L)\cdot\delta_L(f) \tag{4.43}$$

The greedy algorithm can be adapted to this generalized UFL problem by redefining $Z(S)$ in the obvious way

$$Z(S) \;= \sum_{p\in\mathcal{P}} \min_{l\in S} D_p(l) \;+\; \sum_{\substack{L\subseteq\mathcal{L}\\ L\cap S\neq\emptyset}} H(L). \tag{4.44}$$

However, the following theorem shows that the new $Z(S)$ corresponding to (4.37) is no longer supermodular and so the approximation results of Cornuejols *et al.* no longer apply.

Theorem 4.4. *The greedy UFL set function $Z(S)$ for facility subset costs is neither supermodular nor submodular.*

Proof. Similar to Observation 4.3 we now have

$$Z(S \cup \{j\}) - Z(S) \;=\; \sum_{p \in \mathcal{P}} \min\{0, D_p(j) - \min_{l \in S} D_p(l)\} \;+ \sum_{\substack{L \subseteq \mathcal{L} \setminus S \\ j \in L}} H(L) \tag{4.45}$$

$$Z(S \cup \{j,k\}) - Z(S \cup \{k\}) \;=\; \sum_{p \in \mathcal{P}} \min\{0, D_p(j) - \min_{l \in S \cup \{k\}} D_p(l)\} \;+ \sum_{\substack{L \subseteq \mathcal{L} \setminus (S \cup \{k\}) \\ j \in L}} H(L) \tag{4.46}$$

Since the set $\{L \mid \{j\} \subseteq L \subseteq \mathcal{L} \setminus (S \cup \{k\})\}$ in (4.45) is a subset of $\{L \mid \{j\} \subseteq L \subseteq \mathcal{L} \setminus S\}$ in (4.46), the sum of label costs $H(L)$ in (4.45) can exceed the sum of label costs included in (4.46) and so we cannot say that $Z(S \cup \{j,k\}) - Z(S \cup \{k\}) \geq Z(S \cup \{j\}) - Z(S)$ in general. It follows that the greedy set function $Z(S)$ is neither submodular nor supermodular for the generalized UFL-S problem. ■

Finally, the greedy algorithm may be enhanced by applying the *tabu search* meta-heuristic to the UFL problem [136]. Empirical results in [136] show that tabu search finds global optima for many examples in the UFL literature at a reasonable increase in running time.

4.4 Working With a Continuum of Labels

Our experimental Section 4.6 focuses on *multi-model fitting* problems, which are the most natural applications of energy ($\star$). The goal is to estimate parameters for an unknown number of models supported by noisy data with outliers. As was first argued in [69], energies like ($\star$) are powerful criteria for multi-model fitting in general. However, there is a technical hurdle with using combinatorial algorithms for model fitting. In such applications each label represents a specific model, including its parameter values, and the set of all labels $\mathcal{L}$ is a continuum. In line fitting, for example, $\mathcal{L} = \mathbb{R}^2$. Practically speaking, however, the combinatorial algorithms from Section 4.3 require a *finite* set $\mathcal{L}$ of labels (models). Below we review a technique to effectively explore the continuum of model parameters by working with a finite subset of models at any given iteration t.

PEARL algorithm [69]

```
1 propose initial models $\mathcal{L}_0$ (e.g. via random samples from data)
2 run $\alpha$-expansion to compute optimal labeling $f$ w.r.t. $\mathcal{L}_t$
3 re-learn model parameters to get $\mathcal{L}_{t+1}$;  $t{:=}t{+}1$;  goto 2
```

PEARL was the first to use regularization energies and EM-style iterative optimization for geometric multi-model fitting. Other geometric model fitting works have used separate elements such as RANSAC-style random sampling [144, 101] or EM-style iteration [14], but none have combined them in a single optimization framework. The experiments in [69] show that their energy-based formulation beats many state-of-the-art algorithms in this area. In other settings (segmentation, stereo) these elements have been combined in various application-specific ways [166, 14, 123, 164].

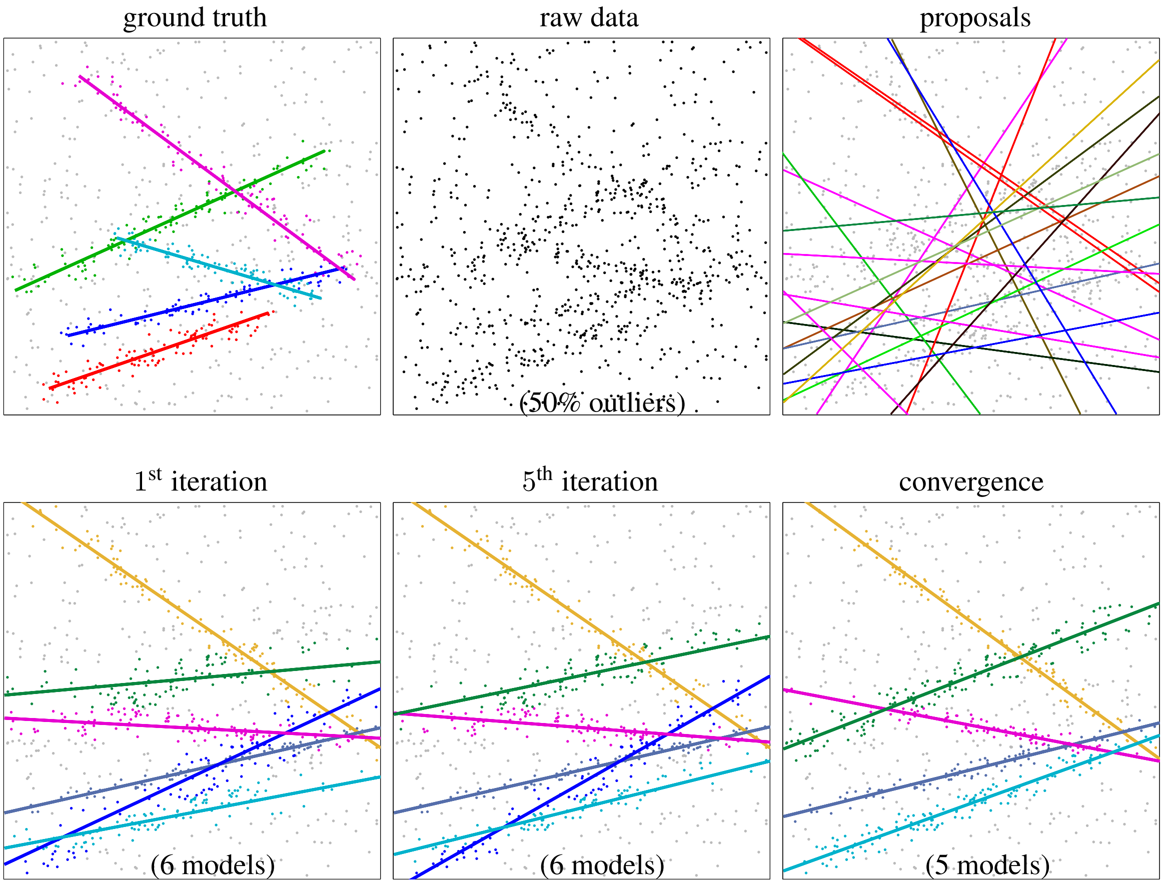


Figure 4.6: Re-estimation helps to align models over time. Above shows 900 raw data points with 50% generated from 5 line intervals. Random sampling proposes a list of candidate lines (we show 20 out of 100). The 1st segmentation and re-estimation corresponds to Li [101], but only the yellow line and gray line were correctly aligned. The decreasing energies in Figure 4.7 correspond to better alignments like the subsequent iterations above. If a model loses enough inliers during this process, it is dropped due to label cost (dark blue line).

Our work contributes better algorithms for the expansion step of PEARL (step **2**), proposes a more general form of label costs in energy ($\star$), describes fast methods for the special case without the spatial smoothness term, and discusses a broader class of multi-model fitting problems in vision.

Review of PEARL for ($\star$) For simplicity, we will discuss PEARL in the context of geometric model fitting, as in [69]. Figure 4.6 illustrates the algorithm's progression. Step **1** of PEARL is to propose an initial set of models $\mathcal{L}_0$. Each proposal can be generated by randomly sampling the smallest subset of data points needed to define a geometric model, exactly as in RANSAC [46]. A larger set of proposals $\mathcal{L}_0$ is more likely to contain models that approximate the true ones. Of course, $\mathcal{L}_0$ will contain many incorrect models as well, but optimizing energy ($\star$) over $\mathcal{L}_0$ (step **2**) will automatically select a small subset of labels from among the best models in $\mathcal{L}_0$, see iteration 1 in Figure 4.6. In this example we used only the label cost regularizer in ($\star$) ignoring the spatial smoothness term, and data fidelity $D_p(l)$ represented an orthogonal distance from point p to line l, see Section 4.6.1. We also fit one additional outlier model ϕ with $D_p(\phi) = \mathrm{const}$.

The initial set of selected models can be further improved as follows. From here on, we represent model assignments by two sets of variables: segmentation variables $\{f_p\}$ that for each data point p specifies the index of a model from the finite set $\mathcal{L}_0$, and parameter variables $\{\theta_l\}$ that specify model parameters currently associated with each model index. Then, energy ($\star$) is equivalent to

$$E(f;\theta) \;= \sum_{p\in\mathcal{P}} D_p(f_p,\theta_{f_p}) \;+ \sum_{pq\in\mathcal{N}} V_{pq}(f_p,f_q,\theta_{f_p},\theta_{f_q}) \;+ \sum_{L\subseteq\mathcal{L}} H(L,\theta_L)\cdot\delta_L(f). \qquad (\star')$$

For simplicity, assume that the smoothness terms in ($\star'$) are Potts interaction potentials [24] and the third term represents simple per-label costs as in (4.1). Then, specific model parameters θ_l assigned to a cluster of points $\mathcal{P}_l = \{p | f_p = l\}$ only affect the first term in ($\star'$), which is a sum of unary potentials. In most cases, it is easy to compute a parameter value $\hat{\theta}_l$ that locally or even globally minimizes $\sum_{p\in\mathcal{P}_l} D_p(l,\theta_l)$. The re-estimated parameters $\{\hat{\theta}_l\}$ correspond to an improved set of labels $\mathcal{L}_1$ that reduces energy ($\star'$) for fixed segmentation f (step **3**).

Now one can re-compute segmentation f by applying the algorithms in Section 4.3 to energy ($\star$) over a new set of labels $\mathcal{L}_1$ (step **2** again). PEARL's re-segmentation and re-estimation steps **2-3** reduce the energy. Iterating these steps generates a sequence of re-estimated models $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2, ...$ converging to a better local minima of energy ($\star$). In our experiments, convergence is typically achieved in 5–20 iterations. In most cases, iterating improves the solution significantly beyond the initial result, see Figure 4.6.

Figure 4.7 shows effectiveness of re-estimation. Starting with only 250 samples (blue plot), re-estimation converges to better solutions than those computed from 1400 samples without re-estimation (a first thick dot on the violet plot). For this example, the algorithm needs at least 250 random samples to be stable, but more than 700 samples is redundant. Figure 4.8 shows an analogous plot for color-model fitting in unsupervised image segmentation, see Section 4.6.2. Recall that Li [101] does not re-estimate beyond the first iteration. His solutions correspond to thick dots at the begging of each plot in Figure 4.7. This approach would heavily rely on

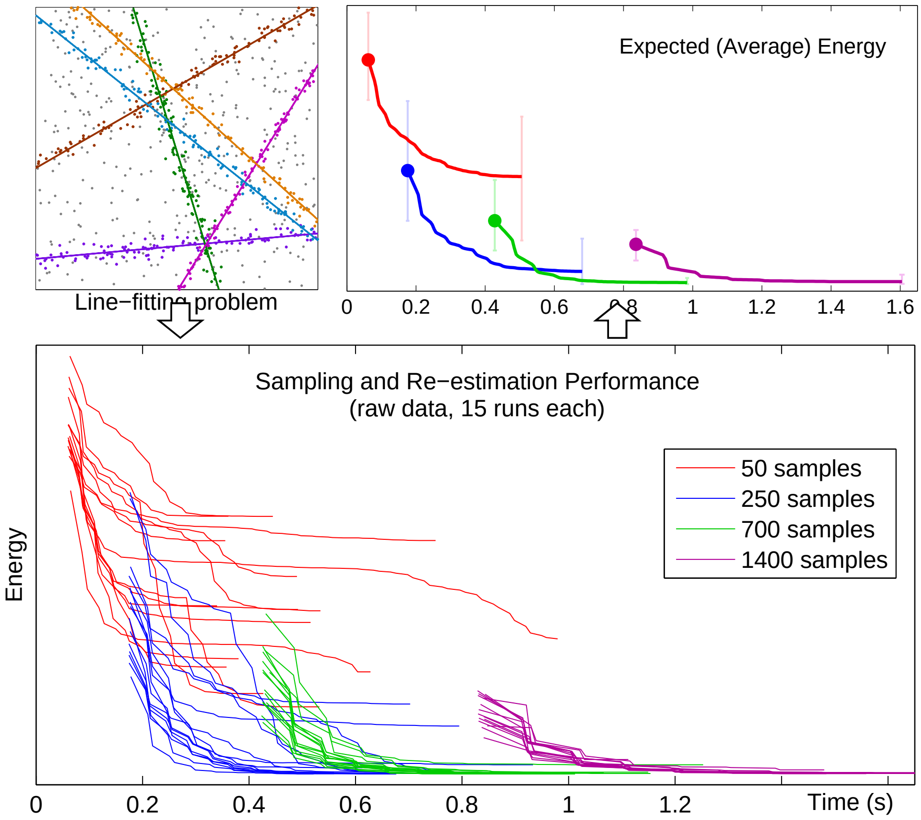


Figure 4.7: Energy ($\star'$) over time for a line-fitting example (1000 points, 40% outliers, 6 ground truth models). Only label cost regularization was used. Re-estimation reduces energy faster and from fewer samples. The first point (•) in each series is taken after exactly one segmentation/re-estimation, and thus suggests the speed of Li [101] using a fast greedy algorithm instead of LP relaxation.

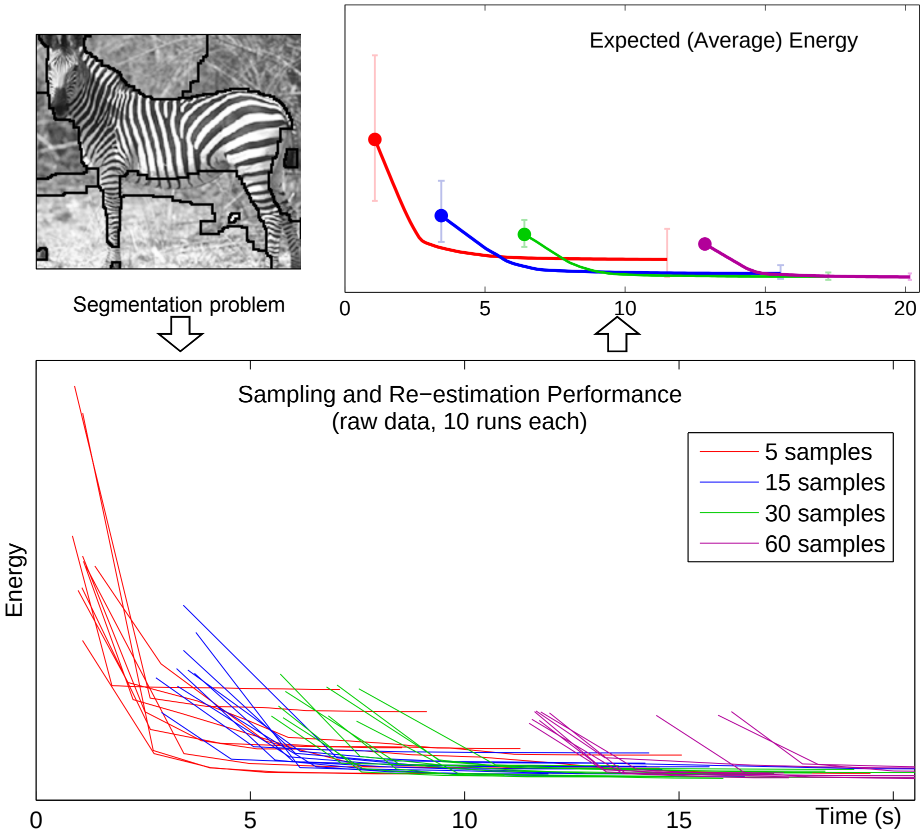


Figure 4.8: Energy ($\star'$) over time for image segmentation (222 × 183 pixels). Smooth cost and label cost were regularized together. The models are 256-dimensional greylevel histograms. See Section 4.6.2 for experimental details.

brute-force random sampling to find solutions of the same quality that we can find with only 250 samples.

Proposal heuristics Re-estimation is a natural way to propose better models from existing ones because it applies to *any* family of models for which a maximum-likelihood estimator can be found. For example, the results in Figures 4.13 and 4.14 were both computed with re-estimation alone.

Re-estimation is by no means the only way to propose new models. Another general heuristic is to fit a new model to the inliers of *two* existing models, and then add this new model to the candidate list; this 'merge' heuristic [147] gives energy $(\star')$ an opportunity to jump out of local minima when computing optimal f. The algorithm in [69] finds lower energy solutions when new 'merge' proposals are added (compare α-SM and α-BM curves in our Section 4.7).

The most effective proposal techniques actually tend to be class-specific and make use of the current solution. A simple example for line fitting is to compute a 'merge' proposal only for pairs of lines that are nearly collinear. Li [101] uses a number of "guided sampling" heuristics specific to motion estimation, but they are only used for the *initial* proposals. In general, proposal heuristics can make our algorithms in Section 4.3 more robust but this is not the point of our work, so all our results use basic re-estimation only.

4.5 Relationship to EM and *K*-means

The main goal of this section is to relate our model fitting algorithm to the standard *expectation maximization* (EM) and *K-means* algorithms. Our discussion will focus on *Gaussian mixture models* (GMM), but we will also consider a geometric example of fitting multiple lines to noisy data points with outliers. To keep things simple for GMM, we use only data terms and label cost terms, even though our full energy $(\star')$ was designed to handle smoothness priors as well.

A number of interesting observations about our model fitting approach can be made:

- K-means minimizes a special case of our energy $(\star')$,
- like K-means, we make *hard assignments* of models to data points (unlike EM), and
- our energy automatically removes unnecessary models from the initial set of proposals (unlike K-means).

Sections 4.5.1–4.5.3 elaborate on these points. Sections 4.5.4 and 4.5.5 show experimental results to help understand the relationship to EM and K-means. Note that our experiments are meant to be illustrative. In particular, we do not suggest that we have a state-of-the-art algorithm for GMM.

The main practical conclusion of this section is that **hard assignment works at least as well as soft assignment when models have (nearly) non-overlapping spatial support**. We claim that many multi-model fitting applications in computer vision satisfy this property, see Figures 4.1, 4.2, and 4.3. Note that in contrast to K-means or EM algorithm our method can also use spatial smoothness prior that is often needed in vision. In this section, however, we focus on a special case of ($\star'$) ignoring the smoothness term mainly to discuss the relationships with the classical multi-model fitting methods.

4.5.1 Standard approaches to finite mixtures

Let some finite set of observed points $X = \{x_p \,|\, p \in \mathcal{P}\}$ be a mixture of independent samples taken from different probability distributions. These distributions are described by probability density functions $\Pr(x \,|\, \theta_l)$ with distinct parameters from a set $\theta = \{\theta_l \,|\, l \in \mathcal{L}\}$, where $\mathcal{L}$ is a finite set of distribution indices (labels). A set of hidden (unobserved) variables $f = \{f_p \in \mathcal{L} \mid p \in \mathcal{P}\}$ represent indices of specific distributions that generated each data point. The probability of sampling from each distribution is defined by a set of mixing parameters $\omega = \{\omega_l \mid l \in \mathcal{L}\}$ such that

$$\Pr(f_p = l) \;:=\; \omega_l, \qquad \sum_{l\in\mathcal{L}} \omega_l = 1, \qquad \omega_l \geq 0.$$

It can be shown that data points in X sampled in this manner correspond to the standard *mixture model* density [15]

$$\Pr(x \,|\, \theta, \omega) = \sum_{l\in\mathcal{L}} \omega_l \cdot \Pr(x \,|\, \theta_l).$$

The problem of estimating a mixture model is to estimate parameters θ and mixing coefficients ω. We will mainly focus on estimating GMM, *i.e.* mixtures of normal distributions $\Pr(x \mid \theta_l) = \mathcal{N}(x \mid \mu_l, \Sigma_l)$ where model parameters $\theta_l = \{\mu_l, \Sigma_l\}$ are the mean and covariance matrix.

Objective functions for EM The classical EM algorithm [15, 40] finds maximum likelihood (ML) estimators for GMM. The ML objective is to find parameters θ and weights ω that maximize the likelihood function

$$\Pr(X \,|\, \theta, \omega) = \prod_{p\in\mathcal{P}} \left(\sum_{l\in\mathcal{L}} \omega_l \cdot \Pr(x_p \,|\, \theta_l) \right). \tag{4.47}$$

As an internal algorithmic step, EM also computes *responsibilities* $\Pr(f_p = l \mid x_p, \theta, \omega)$ to estimate which mixture components could have generated each data point.

The EM algorithm can be generalized [15] to compute *maximum a posteriori* (MAP) estimates of θ and ω maximizing the posterior $\Pr(\theta, \omega \mid X) \propto \Pr(X \mid \theta, \omega) \Pr(\theta) \Pr(\omega)$. For example, a common MAP objective is

$$\Pr(\theta, \omega \,|\, X) \propto \prod_{p\in\mathcal{P}} \left(\sum_{l\in\mathcal{L}} \omega_l \cdot \Pr(x_p \,|\, \theta_l) \right) \cdot \prod_{l\in\mathcal{L}} \omega_l^{\alpha-1} \tag{4.48}$$

which combines the ML objective (4.47) with a uniform prior on θ and Dirichlet prior on weights ω

$$\Pr(\omega) = \mathrm{Dir}(\omega \,|\, \alpha) \propto \prod_{l\in\mathcal{L}} \omega_l^{\alpha-1}, \qquad \alpha > 0. \tag{4.49}$$

The Dirichlet prior is a uniform distribution for $\alpha = 1$ but for $\alpha < 1$ it prefers to estimate ω such that most ω_l are close to zero. A smaller choice of α creates a stronger sparsity effect on ω, and so α is called a *sparsity parameter*. In theory, this prior should encourage mixture models where most components are close to zero. According to [45] and in our own experience (see Figure 4.12), negative values of α are often necessary in practice to effectively remove redundant models. However, the Dirichlet prior is not a proper (integrable) distribution for $\alpha \leq 0$.

Objective functions for K-means Standard K-means can also be seen as an ML approach to estimating mixture models. The elliptical[4] K-means algorithm [137] maximizes the following likelihood on the same probability space

$$\Pr(X \,|\, f, \theta) = \prod_{p \in \mathcal{P}} \Pr\left(x_p \,|\, \theta_{f_p}\right). \tag{4.50}$$

In contrast to EM, this approach directly computes labeling $f = \{f_p \mid p \in \mathcal{P}\}$ rather than responsibilities, while mixing coefficients ω_l are implicitly estimated as percentages of points with $f_p = l$. It is often said that K-means performs *hard assignment* of models to data points, whereas EM performs *soft assignment* leaving room for uncertainty in the labeling f.

It is possible to derive a version of K-means that explicitly estimates mixing weights ω. Assuming that f_p are independent, one gets the following prior on the labeling

$$\Pr(f \,|\, \omega) = \prod_{p \in \mathcal{P}} \Pr(f_p \,|\, \omega) = \prod_{p \in \mathcal{P}} \omega_{f_p}. \tag{4.51}$$

Combining this prior with likelihood (4.50) and assuming non-informative (uniform) priors for ω and θ, Bayes rule then gives posterior distribution

$$\Pr(f, \theta, \omega \,|\, X) \propto \prod_{p \in \mathcal{P}} \omega_{f_p} \cdot \Pr(x_p \,|\, \theta_{f_p}). \tag{4.52}$$

Values of f, θ, ω maximizing this distribution are MAP estimates of these parameters. Like the standard K-means algorithm, one can maximize (4.52) by iterating two steps: first optimize over f for fixed θ, ω and then (independently) optimize over ω and θ for fixed f. We refer to this algorithm as *weighted (elliptical) K-means*.

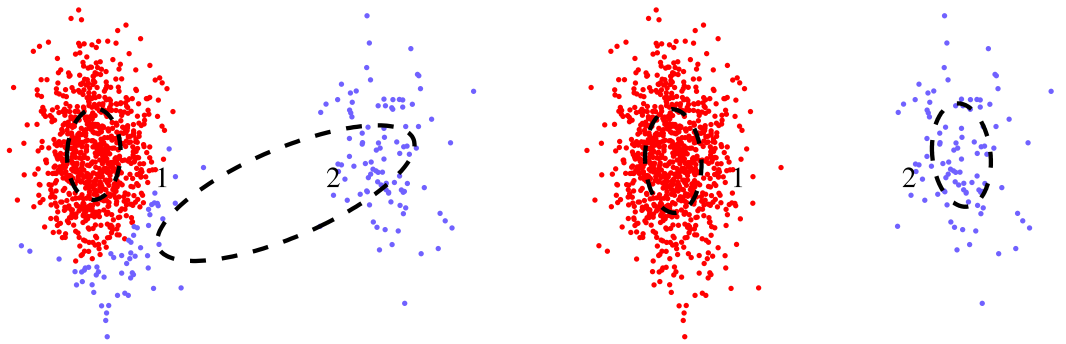
Discussion of priors Instead of a uniform prior on ω used in (4.52) one can add any informative prior for mixture weights. For example, the Dirichlet prior (4.49) gives posterior

$$\Pr(f, \theta, \omega \,|\, X) \propto \prod_{p \in \mathcal{P}} \omega_{f_p} \cdot \Pr(x_p \,|\, \theta_{f_p}) \cdot \prod_{l \in \mathcal{L}} \omega_l^{\alpha - 1}. \tag{4.53}$$

For $\alpha < 1$ this posterior encourages sparsity of weights ω. Objectives (4.50) and (4.52) can be derived from (4.53) for other values of α. Setting $\alpha = 1$ gives the uniform prior on ω and (4.53) reduces to the weighted K-means posterior (4.52). Setting α very large ($\alpha \to \infty$) encourages equal weights $\omega_l = \frac{1}{K}$ and so (4.53) reduces to the standard K-means likelihood (4.50). Figure 4.9 shows how this difference can affect solutions. Standard K-means' bias to equal-size components is another way to understand its sensitivity to the choice of K.

However, our label costs are more closely related to a sparsity prior known as the *spike-and-slab* distribution [109]. See [39] for a detailed discussion of this relationship.

[4] The *elliptical* version of K-means explicitly estimates a covariance matrix Σ so that each set of parameters is $\theta_l = \{\mu_l, \Sigma_l\}$.



elliptical K-means; see (4.50), (4.54) weighted elliptical K-means; see (4.52), (4.55)

Figure 4.9: Mixture of two Gaussians where most data points were generated from the first component ($\omega_1 > \omega_2$). Standard K-means prefers equal cluster sizes because it assumes $\omega_1 = \omega_2$, whereas weighted K-means has no such bias.

4.5.2 Using label costs for finite mixtures

The standard K-means directly minimizes the negative-log of the likelihood function (4.50), giving energy

$$E(f;\theta) = -\sum_{p\in\mathcal{P}} \log \Pr(x_p \,|\, \theta_{f_p}). \tag{4.54}$$

Similarly, the weighted K-means algorithm minimizes the negative-log of the posterior distribution (4.52)

$$E(f;\theta,\omega) = -\sum_{p\in\mathcal{P}} \log\big(\omega_{f_p}\cdot \Pr(x_p \,|\, \theta_{f_p})\big)\,. \tag{4.55}$$

Both of these K-means energies are expressible as data terms D_p in our energy $(\star')$.

Note that posterior energy (4.55) is derived from the i.i.d. assumption (4.51) on assignment variables f_p. This assumption holds when the sampling process does not have any coherence or constraints (*e.g.* occlusions). In some examples, however, variables f_p may be dependent. For example, pairwise interactions could be easily incorporated into a prior for f yielding a posterior energy with the first and second terms in $(\star')$. Such a prior may be also useful for its regularization effect. In the context of GMM estimation, however, it makes more sense to regularize using some sparsity prior such as *step-and-slab* [109]. It can be shown that the global minima of the energy

$$E(f;\theta,\omega) = -\sum_{p\in\mathcal{P}} \log\big(\omega_{f_p}\cdot \Pr(x_p \,|\, \theta_{f_p})\big) \;+\; \sum_{l\in\mathcal{L}} H(l)\cdot\delta_l(f) \tag{4.56}$$

correspond to MAP solutions under the step-and-slap sparsity prior; see [39] for details.

Note that the K-means algorithm for (4.54) is very sensitive to initialization even if the right number of models K is given, see Figure 4.11. If the number of given initial models K is too large, the algorithm will over-fit these K models to data, see Figure 4.10e. The extra label cost term in energy (4.56) removes many problems associated with fixed K. We initialize our method with a relatively large number of randomly sampled models and minimization of (4.56) leads to a solution with a small number of good models, see Figure 4.6.

4.5.3 Label costs as information criterion

Regularizers are useful energy terms because they can help to avoid over-fitting. In statistical model selection, various *information criteria* have been proposed to fulfil a similar role. Information criteria penalize overly-complex models, preferring to explain the data with fewer, simpler models (Occam's razor [107]).

For example, consider the well-known *Akaike information criterion* (AIC) [3]:

$$\min_{\Theta} \ -2\log\Pr(X\,|\,\Theta) \ + \ 2|\Theta| \tag{4.57}$$

where Θ is a model, $\Pr(X \mid \Theta)$ is a likelihood function and $|\Theta|$ is the number of parameters in Θ that can vary. This criterion was also discussed by Torr [144] and Li [101] in the context of motion estimation.

Another well-known example is the *Bayesian information criterion* (BIC) [26, 107]:

$$\min_{\Theta} \ -2\log\Pr(X\,|\,\Theta) \ + \ |\Theta|{\cdot}\log|\mathcal{P}| \tag{4.58}$$

where $|\mathcal{P}|$ is the number of observations. The BIC suggests that label costs should be scaled in logarithmic proportion to the number of data points or, in practice, to the estimated number of observations per model. In contrast, AIC over-fits as we add more observations from the true models. See [26] for an intuitive discussion and derivation of BIC in general, particularly Sections 6.3–6.4, and see Torr's work [144] for insights specific to vision.

4.5.4 Experimental results for GMM estimation

Figure 4.10 juxtaposes representative GMM estimation results by basic EM (4.47), EM with Dirichlet prior (4.48), elliptical K-means (4.54,4.55), and our approach to label cost energy (4.56). For simplicity, Figure 4.10 represents EM's "soft assignment" at each point p using only one color corresponding to the model with the highest *responsibility*. The results for K-means and energy (4.56) show colors corresponding to their "hard assignments".

Implementation of (weighted) elliptical K-means maximizing (4.54,4.55) is fairly straightforward. Since (4.48) automatically controls sparsity of the solution, we can initialize this version of EM with a large number of randomly sampled models. As discussed in [45], this makes EM robust to initialization and helps to avoid local minima.

Energy (4.56) represents a combination of the first and the third terms in ($\star'$). To minimize (4.56) we iterate PEARL (Section 4.4) in combination with the greedy optimization method (Section 4.3.5) for each expansion step. Similarly to [45] and to our EM approach for (4.48), optimization of (4.56) via PEARL avoids local minima when initialized with a large set of randomly sampled models.

The second column in Figure 4.10 shows the results typical for both standard (4.54) and weighted K-means (4.55). The two methods worked similarly on all tests in Figure 4.10 because all models there have approximately the same number of inliers. Such examples can not reveal the bias of standard K-means to equalizing mixing weights (see Figure 4.9).

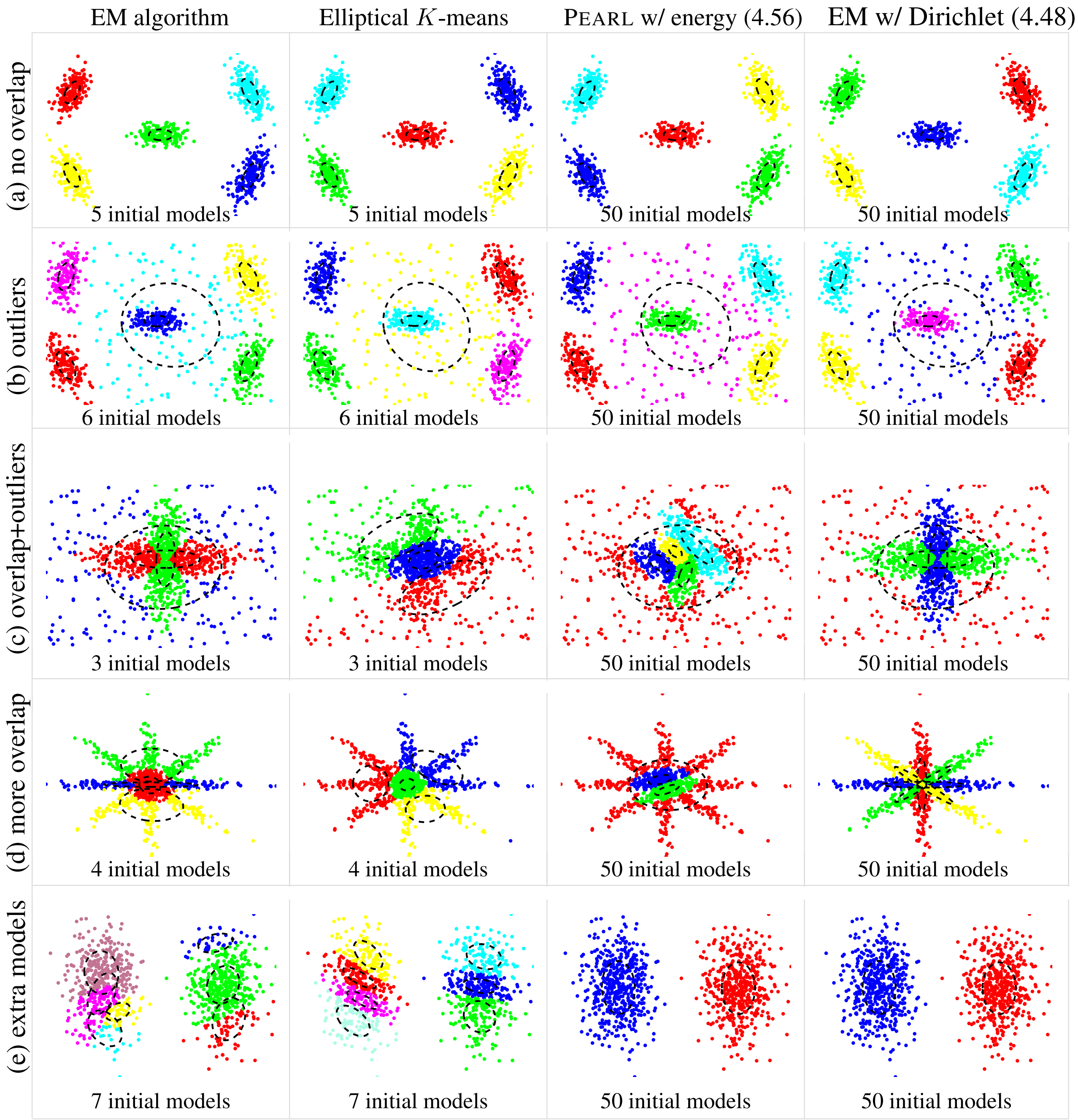


Figure 4.10: Each row shows how GMM algorithms behave on a particular example. This table is for illustrative purposes, and is *not* meant to be a state-of-the-art comparison. (a) If models do not overlap then all algorithms work. (b) Most algorithms can handle uniform outliers by fitting an extra model. (c) EM finds overlapping models thanks to soft assignment; hard assignment has bias towards isolated models. (d) Basic EM (4.47) may easily get stuck in local minima with only a little more ambiguity in the data. But, EM with sparsity prior (4.48) can avoid such minima by choosing solution from a large set of model samples. Bad solution by PEARL in this case of heavy spatial overlap between the models is due to "hard assignments". (e) Basic EM and K-means usually fail when given too many initial models, whereas PEARL with label cost energy (4.56) and EM with Dirichlet-based posterior (4.48) keep the minimum number of models explaining the data. See Section 4.5.4 for discussion.

One important conclusion from Figure 4.10 is that energy (4.56) works well on all examples (a,b,e) where the models do not have significant spatial overlap. This case is very common in computer vision problems where models occlude each other rather than intersect.

If K-means and basic EM (4.47) were initialized with a correct number of models, they also worked very well for spatially non-overlapping models (a,b), however, EM was more sensitive to outliers in (b). If basic EM and K-means are initialized with a wrong number of models (e) they overfit these models to data, while Dirichlet-based posterior (4.48) and label cost energy (4.56) keep the minimal number of necessary models.

In general, EM handled intersecting models in (c) better than K-means and our method with (4.56). Arguably, soft assignments of models to data points help EM to deal with such overlapping models. More severe cases of model mixing in (d) were problematic for basic EM with a fixed number of models (4.47) due to local minima. However, EM for Dirichlet-based posterior (4.48) could avoid such local minima by selecting good models from a large initial sample.

In general, our approach with (4.56) and EM with (4.48) benefit from larger number of initial proposals which increases the chances that correct models are found. The 2 right columns in Figure 4.10 show the minimum number of initial randomly sampled models (proposals) that these algorithms needed to robustly generate good results.

4.5.5 Experimental results for geometric model fitting

Figures 4.11 and 4.12 show representative multi-line fitting results by basic EM (4.47), EM with Dirichlet prior (4.48), elliptical K-means (4.54,4.55), and our approach to label cost energy (4.56). As before, we represent EM's "soft assignment" at each point using only the color of the model with the highest *responsibility*. The results for K-means and energy (4.56) show colors of their "hard assignments".

The data set for experiments in Figures 4.11-4.12 consists of 300 inliers for 5 lines and 180 outliers. Each line model $\theta = \{a, b, c, \sigma\}$ includes noise variance σ. Log-likelihood $D_p(\,l\,) = -\log Pr(x_p|\theta_l)$ for a given data point x_p and line θ_l assumes Gaussian orthogonal error and is given in (4.59). We also fit one uniform outlier model ϕ with likelihood $Pr(x_p|\phi) = \text{const} > 0$ where const was manually tuned. Some additional general details about the experimental setup for line fitting can be found in Section 4.6.1. Optimization of functionals (4.47), (4.48), (4.54), (4.55), and (4.56) via EM, K-means, and PEARL is implemented as in the previous section.

Figure 4.11 demonstrates that the standard K-means for (4.54), (4.55), and basic EM algorithm for (4.47) are very sensitive to local optima. Figure 4.12a shows that such local minima are avoided by optimization algorithms that select a few good lines from a large pool of initial models using sparsity control: label costs in (4.56) or Dirichlet prior in (4.48). The number of models generated by (4.56) and (4.48) is controlled by parameters h and α, see Figures 4.12b and 4.12c.

Our main conclusion from Section 4.5 is that "hard assignments" have no particular disadvantages in cases where spatial overlap between the observed models constitutes only a small portion of their support. In image analysis problems (*e.g.* Figures 4.1, 4.2, and 4.3) models

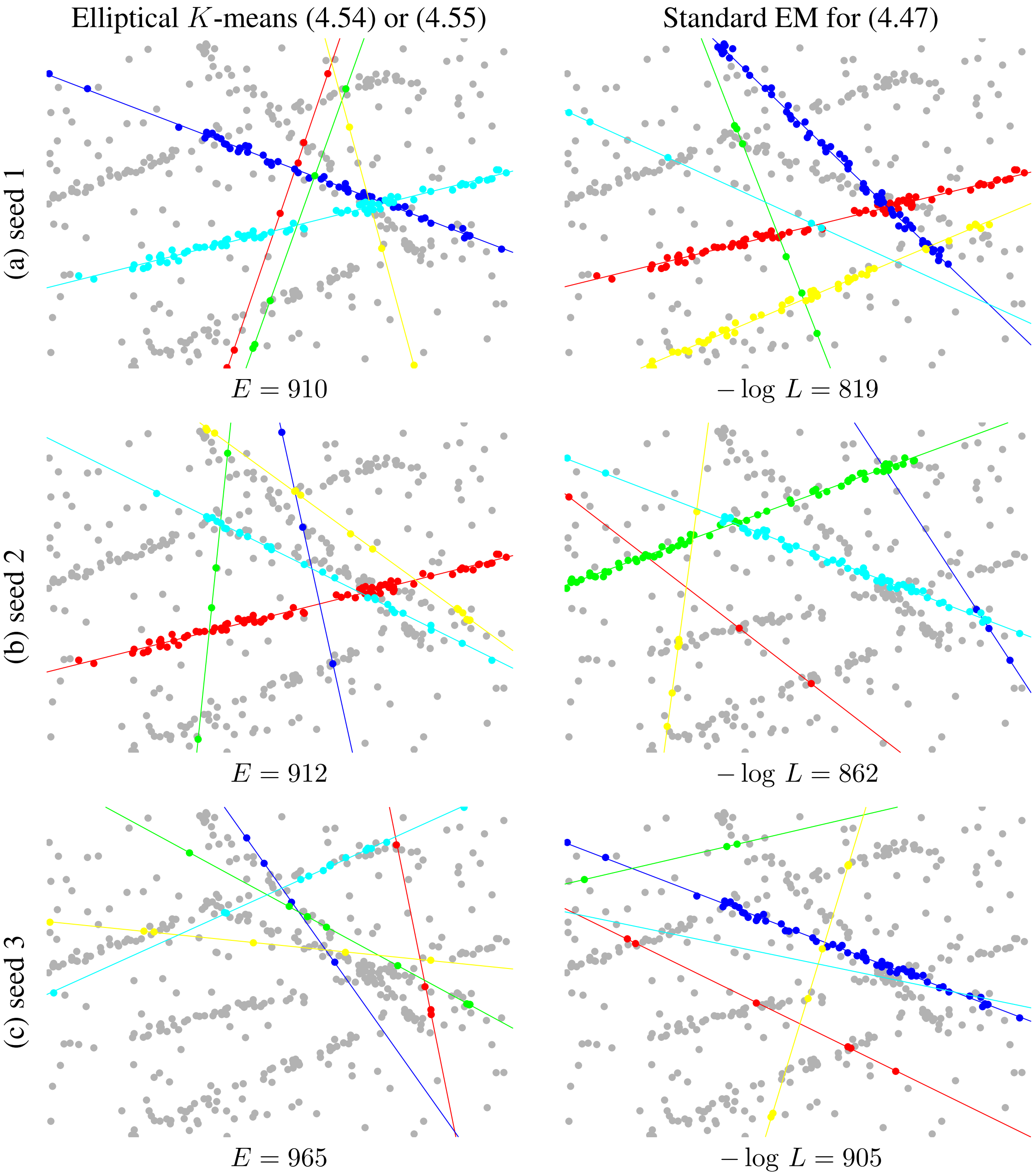


Figure 4.11: Standard K-means and EM with a fixed number of models get stuck in local minima. The data points include (in total) 300 inliers for 5 lines and 180 outliers. Here we assumed that the correct number of models is known and estimated $K = 5$ lines and one outlier model. Solutions in (a)-(c) correspond to different initializations with 5 randomly sampled lines. The ground truth configuration has energy $E = 797$ in (4.55) and log-likelihood $-\log L = 721$ in (4.47).

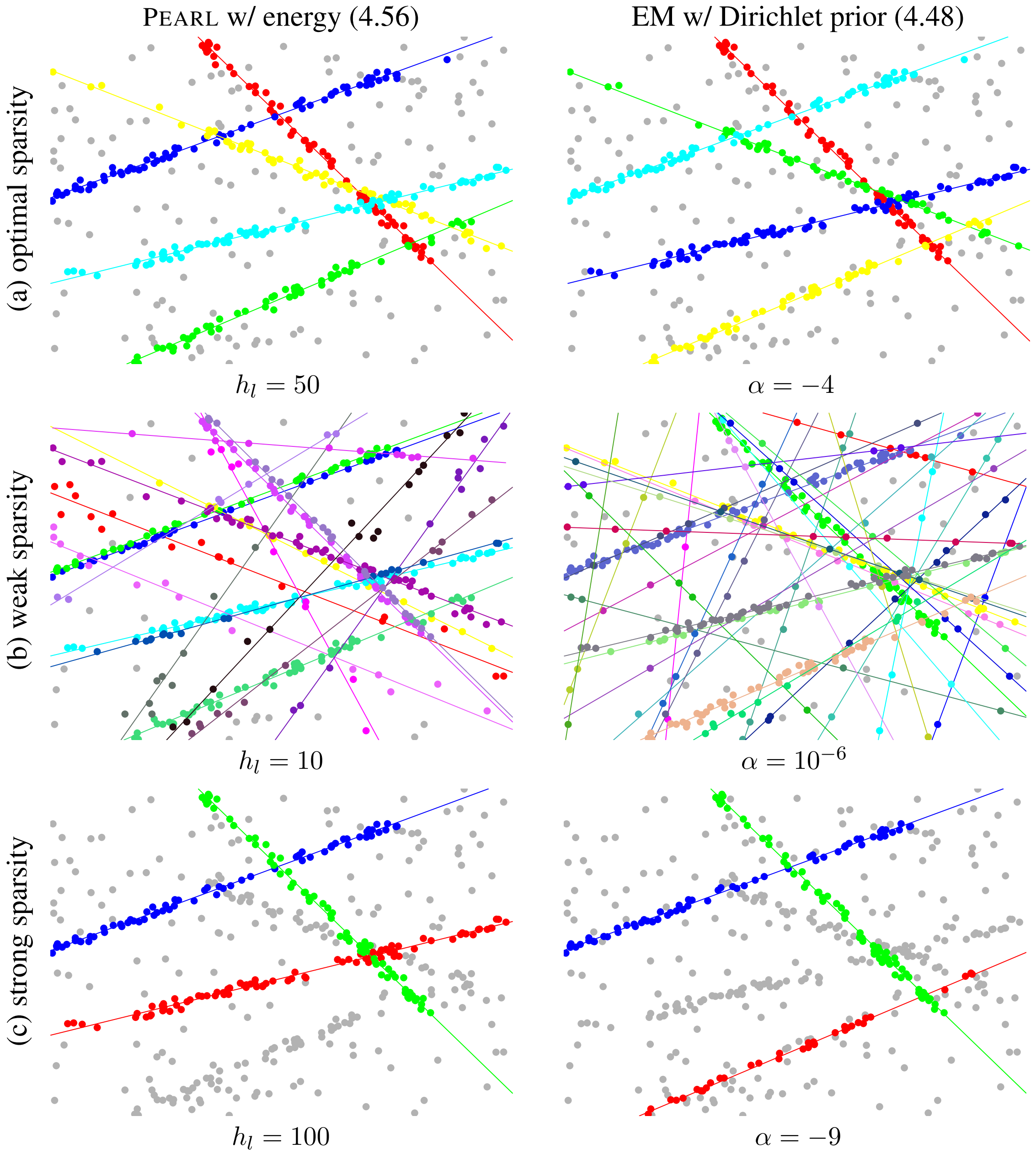


Figure 4.12: Label costs in (4.56) or sparsity prior in (4.48) significantly improve the results on the data from Figure 4.11. Now a small number of models near ground truth (a) can be automatically computed from a large pool of random initial models, as in Figure 4.6. In contrast to Figure 4.11, the results are stable for different initializations as long as the set of initial randomly sampled lines is large enough (*e.g.* 500 lines). Parameters h and α control sparsity of the results (a-c).

often correspond to separate objects with distinct spatial support. Objects normally "occlude" each other rather than "intersect". Thus, "hard assignments" should be appropriate for many multi-model fitting problems in computer vision. In contrast to standard "soft assignment" methods like EM, besides sparsity prior (label costs) our general approach to model fitting can also integrate a spatial smoothness prior - the second term in ($\star'$) that was ignored in this section. Figures 4.1, 4.2, and 4.3 show that this combination of regularizers is useful in vision.

4.6 Applications and Experimental Setup

The experimental setup is essentially the same for each application: generate proposals via random sampling, compute initial data costs D_p, and run the iterative algorithm from Section 4.4. The only changing components are the application-specific D_p and regularization settings. Section 4.6.1 outlines the setup for basic geometric models: lines, circles, homographies, motion. Section 4.6.2 describes the unsupervised image segmentation setup.

4.6.1 Geometric multi-model fitting

Here each label $l \in \mathcal{L}$ represents an instance from a specific class of geometric model (lines, homographies), and each $D_p(l)$ is computed by some class-specific measure of geometric error. The strength of per-label costs and smooth costs were tuned for each application.

Outliers All our experiments handle outliers in a standard way: we introduce a special outlier label ϕ with $H(\phi) = 0$ and $D_p(\phi) = \text{const} > 0$ manually tuned. This corresponds to a uniform distribution of outliers over the domain.

Simple synthetic examples (lines, circles, etc.)

Throughout this chapter we used many illustrative examples of multi-line fitting. Below we detail the corresponding set-up and discuss some additional synthetic tests with simple geometric models. Our energy ($\star'$) was motivated by applications in vision that involve images (Sections 4.6.1–4.6.2; also compression, see [39]), but synthetic examples with simple models help to understand our energy, our algorithm, and their relation to standard methods.

Line fitting Data points are sampled i.i.d. from a ground truth set of line segments (*e.g.* Figure 4.6), under reasonably similar noise; outliers are sampled uniformly. Since the data is i.i.d. we set $V_{pq} = 0$ in ($\star'$) and use the greedy algorithm from Section 4.3.5. We also use fixed per-label costs as in (4.56). Keeping per-label costs independent of θ simplifies the re-estimation of θ itself.

Figure 4.6 is a typical example of our line-fitting experiments with outliers. In 2D each line model l has parameters $\theta_l = \{a, b, c, \sigma\}$ where $ax + by + c = 0$ defines the line and σ^2 is the variance of data; here a, b, c have been scaled such that $a^2 + b^2 = 1$. Each proposal line is generated by selecting two random points from $\mathcal{P}$, fitting a, b, c accordingly, and selecting a random initial σ based on a prior. The data cost for a 2D point $x_p = (x_p^{\mathrm{x}}, x_p^{\mathrm{y}})$ is computed w.r.t.

orthogonal distance

$$D_p(\,l\,) \;=\; -\log\Big(\tfrac{1}{\sqrt{2\pi}\sigma}\exp\Big(-\tfrac{(ax^{\mathrm{x}}_p+bx^{\mathrm{y}}_p+c)^2}{2\sigma^2}\Big)\Big)\,. \tag{4.59}$$

Besides the greedy algorithm for ($\star'$) without smoothness, we also tested α-expansion for high-order label cost potentials (Section 4.3.1). Not surprisingly, the greedy algorithm was by far the best algorithm when smooth costs are not involved. Greedy gives similar energies to α-expansion but is 5–20 times faster.

Figure 4.7 shows the trend in running time as the number of random initial proposals is increased. For 1000 data points and 700 samples, convergence took .7–1.2 seconds with 50% of execution time going towards computing data costs (4.59) and performing re-estimation.

Note that (4.59) does not correspond to a well-defined probability density function. The density for unbounded lines cannot be normalized, so lines do not spread their density over a coherent span. Still, in line-fitting it is common to fit full lines to data that was actually generated from line *intervals*, *e.g.* [69, 168]. The advantage of full lines is that they are a lower-dimensional family of models, but when lines are fit to data generated from intervals this is a model mis-specification, causing discrepancy between the energy being optimized versus the optimal solution from a generative viewpoint. Surprisingly, [69] showed that there are examples where introducing spatial coherence ($V_{pq} > 0$) for i.i.d. line interval data can actually improve the results significantly. We hypothesize that, in this case, spatial coherence can be trained discriminatively to counter the discrepancy caused by fitting unbounded lines to line interval data.

Line interval fitting Figure 4.13 shows three interval-fitting results, all on the same data. Each solution was computed from a different (random) set of 1500 initial proposals. Line intervals require many more proposals than for lines because intervals are higher-dimensional models. Each result in Figure 4.13 took 2–4 seconds to converge, with 90% of the execution time going towards computing data costs and performing re-estimation (in MATLAB).

We model an interval from point a to point b as an infinite mixture of isotropic Gaussians $\mathcal{N}(\mu, \sigma^2)$ for each μ interpolating a and b. The probability of a data point appearing at position x is thus

$$\Pr(x \,|\, a, b, \sigma^2) = \int_0^1 \mathcal{N}\big(x \,|\, (1-t)a + tb, \sigma^2\big)\,\mathrm{d}t. \tag{4.60}$$

In two dimensions, the above integral evaluates to

$$\begin{aligned} &\tfrac{1}{4\pi\sigma^2\|a-b\|}\cdot\exp\Big(-\Big(\tfrac{x^{\mathrm{x}}(b^{\mathrm{y}}-a^{\mathrm{x}})-x^{\mathrm{y}}(b^{\mathrm{x}}-a^{\mathrm{y}})+a^{\mathrm{y}}b^{\mathrm{x}}-a^{\mathrm{x}}b^{\mathrm{y}}}{\sqrt{2}\sigma\|a-b\|}\Big)^2\Big) \\ &\quad\cdot\Big(\mathrm{erf}\Big(\tfrac{(x-b)\cdot(a-b)}{\sqrt{2}\sigma\|a-b\|}\Big)-\mathrm{erf}\Big(\tfrac{(x-a)\cdot(a-b)}{\sqrt{2}\sigma\|a-b\|}\Big)\Big) \end{aligned} \tag{4.61}$$

where $x = (x^{\mathrm{x}}, x^{\mathrm{y}})$ is and $\mathrm{erf}(\cdot)$ is the *error function*.

Given a set $X_l = \{x_p\colon\, f_p = l\}$ of inliers for label l, we find maximum-likelihood estimators $\theta_l = \{a, b, \sigma\}$ by numerically minimizing the negative-log likelihood

$$E(X_l; a, b, \sigma) = -\sum_p \log\Pr(x_p \,|\, a, b, \sigma^2). \tag{4.62}$$

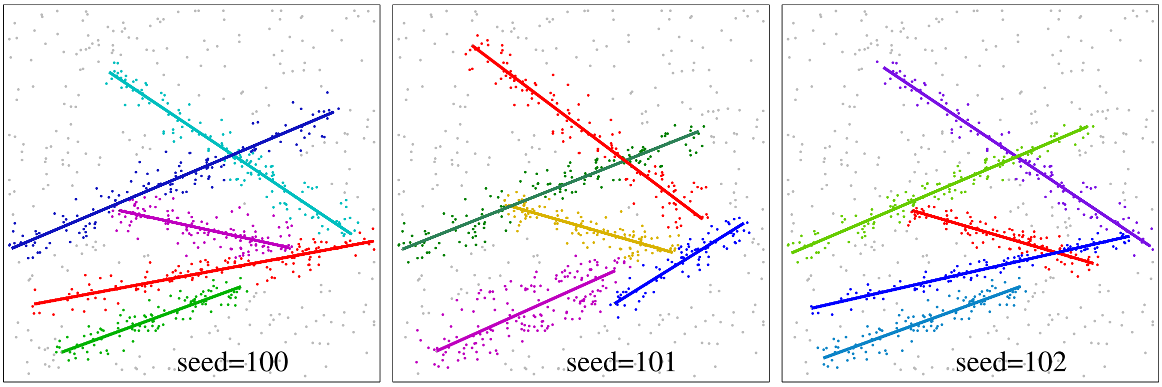


Figure 4.13: We can also fit line *intervals* to the raw data in Figure 4.6. The three results above were each computed from a different set $\mathcal{L}$ of random initial proposals. See Section 4.6.1 for details.

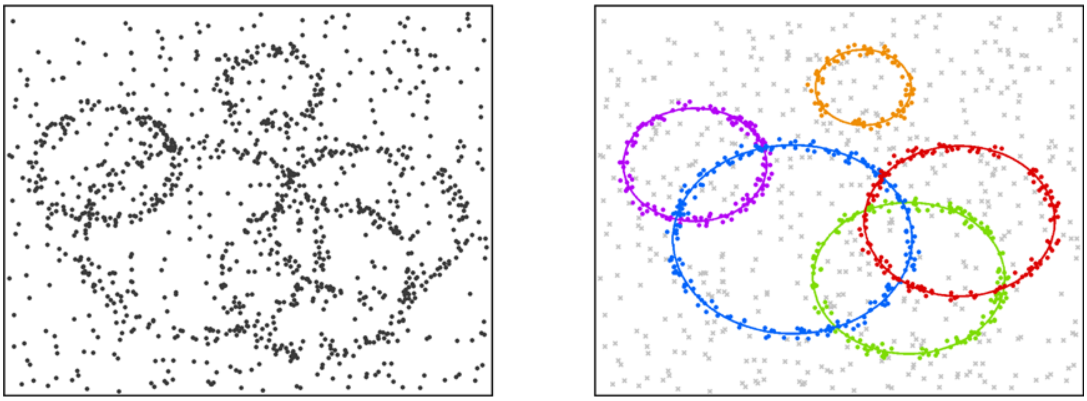


Figure 4.14: For multi-model fitting, each label can represent a specific model from any family (Gaussians, lines, circles...). Above shows circle-fitting by minimizing geometric error of points.

Circle fitting Figure 4.14 shows a typical circle-fitting result. Our circle parameters are center-point a, radius r, and variance σ^2. We model a circle itself as an infinite mixture of isotropic Gaussians along the circumference. Proposals are generated by randomly sampling three points, fitting a circle, and selecting random σ based on some prior. We find ML estimators numerically, much like for line intervals.

Homography estimation

Energy ($\star'$) can be used to automatically detect multiple homographies in uncalibrated wide-base stereo image pairs. Our setup follows [69], so we give only a brief outline.

The input comprises two (static) images related by a fundamental matrix. We first detect SIFT features [106] and do exhaustive matching as a preprocessing step; these matches are our observations. The models being estimated are homographies, and each proposal is generated by sampling four potential feature matches. Data costs measure the symmetric transfer error (STE) [63] of a match w.r.t. each candidate homography. Our set of neighbors $pq \in \mathcal{N}$ is determined by a Delaunay triangulation of feature positions in the first image. Re-estimation is done by minimizing the STE of the current inliers via Levenberg-Marquardt [63]. Figures 4.2c and 4.16 show representative results.

Rigid motion estimation

The general setup follows [69, 101] and is essentially the same as for homography estimation, except now each model is a fundamental matrix $F = [K'\,t]_\times K'RK^{-1}$ corresponding to a rigid body motion (R, t) and intrinsic parameters K [63].

Again, SIFT matches work as data points. Initial proposals are generated by randomly sampling eight matching pairs. Fundamental matrices [63] are computed by minimizing the non-linear SSD error using Levenberg-Marquardt. Data costs measure the squared Sampson's distance [63] of a match with respect to each candidate fundamental matrix. Figures 4.1(c) and 4.18 show representative results.

4.6.2 Image segmentation

Our goal is to automatically partition an image into some small number of regular segments with consistent appearance. In contrast to *superpixels*, our segments can be of any size and need not be contiguous. We propose to label the image using the following form of energy ($\star'$)

$$E(f,M) \;=\; \sum_{l\in\mathcal{L}} \underbrace{\sum_{p:f_p=l} -\log P(I_p \mid M_l)}_{\text{segment appearance}} \;+\; \underbrace{\lambda \sum_{pq\in\mathcal{N}} [f_p \neq f_q]}_{\text{segments' boundaries}} \;+\; \underbrace{\sum_{l\in\mathcal{L}} H(l)\!\cdot\!\delta_l(f)}_{\text{segments' labels}} \tag{4.63}$$

where parameter M_l describes probability distribution associated with label l. For example, if values I_p are image intensities/colors[5] then vector M_l could represent an intensity histogram or parameters of some family of distributions.

[5] In general, I_p could represent any feature at pixel p, *e.g.* texture.

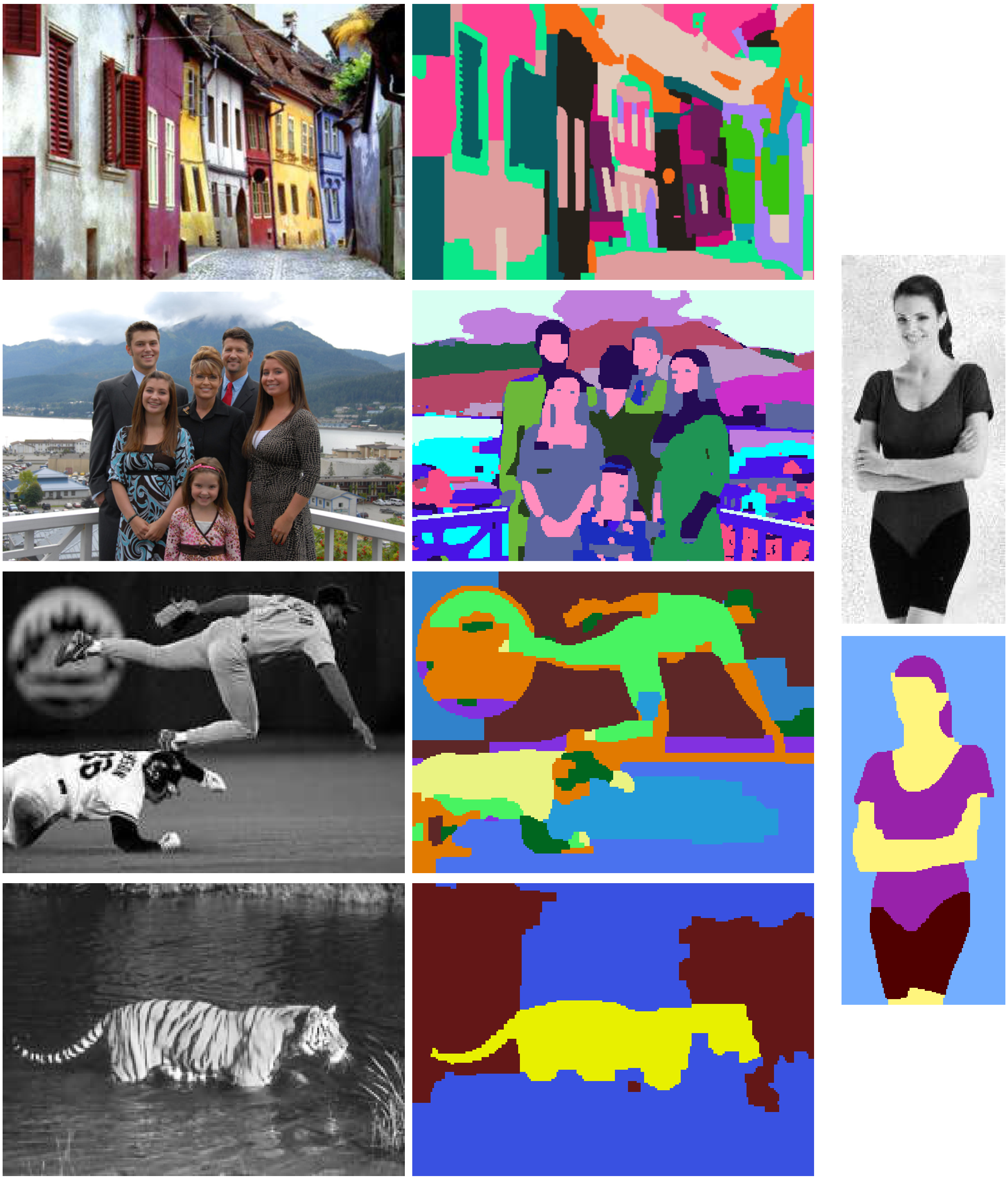


Figure 4.15: Unsupervised segmentation by clustering simultaneously over pixels and color space using Gaussian Mixtures (color images) and non-parametric histograms (gray-scale images). Notice we find coarser clustering on baseball than Zabih & Kolmogorov [164] without over-smoothing. For segmentation, our energy is closer to Zhu & Yuille [166] but our algorithm is more powerful than region-competition.

In what sense does segmentation energy (4.63) correspond to the goals proclaimed at the beginning of the previous paragraph? The third term sums penalties h_l for each label (model M_l) used in the image. This directly encourages a small number of segments. The second term is a standard expression for regularity of segment boundaries.

Information theory helps to show how the first term in (4.63) yields segments with *consistent appearance*. Indeed, following Kraft-McMillan theorem [107], any probability distribution $P(I \mid M)$ corresponds to some coding scheme for storing image intensities. Moreover, $-\log P(I_p \mid M)$ is the number of bits required to represent any given intensity I_p using coding scheme $P(I \mid M)$. Therefore,

$$\sum_{p \in S} -\log P(I_p \mid M)$$

is the number of bits required to describe the appearance of any segment $S \subset \mathcal{P}$ using coding scheme M. When optimizing over distribution M, the expression above yields the shortest possible description of segment S, that is

$$|S| \cdot H(I \mid S) \;\; = \;\; \inf_M \sum_{p \in S} -\log P(I_p \mid M)$$

where $H(I|S)$ is the entropy of intensities in segment S. Thus, optimization over all distribution models M makes the first term of energy (4.63) equal

$$\sum_{l \in \mathcal{L}} |S_l| \cdot H(I \mid S_l)$$

where $S_l = \{p : f_p = l\}$ is a segment with label l. This quantity can be further optimized over segmentation (labeling) f. It achieves its minimum for any segmentation with constant intensity segments where $H(I|S) = 0$. Such segments can be connected or disconnected. The size of the segments is also irrelevant. For example, single pixel segments are optimal for the quantity above. Alternatively, segments could be connected components of the same intensity pixels. More generally, low values of the quantity above correspond to segments with low variability of intensity, that is, segments with consistent or homogeneous appearance.

In our segmentation experiments based on energy (4.63) the appearance models M_l are 256-dimensional histograms for greyscale images, and Gaussian mixtures in RGB space for color images. Initial proposals for models M_l were generated by sampling small patches of the image, just like in [166, 164]. Similarly to [166, 164] we iterated segmentation and model re-estimation steps to optimize our energy over f and M. We did not use segmentation-specific heuristics such as merging or splitting the histograms. Figure 4.8 shows running-time performance of our *coordinate descent* approach using α-expansions to optimize (4.63) over f, as in Section 4.3.

Our results in Figures 4.3 and 4.15 show how energy (4.63) balances regularity and homogeneity of segments. It is particularly instructional to compare image segmentation results in Figure 4.3(b)-(c). The result in (b) uses only spatial regularization as in energy (4.2), see [164]. This approach over-smoothes the segments even when the weight of the regularization term is too small to merge all "zebra" parts. The label costs term in (4.63) allows to obtain "zebra" (c) without over-smoothing. In this case we do not depend on the spatial regularization to merge all "zebra" parts and smoothing weight λ can be significantly reduced.

The label costs term in (4.63) could be used to obtain segments with certain preferred appearance by assigning penalties h_l depending on M_l. Also note that a general version of our label costs term in ($\star$) uses subsets of labels. This allows interesting new ideas for segmentation, as recently demonstrated in [94] in the context of object recognition.

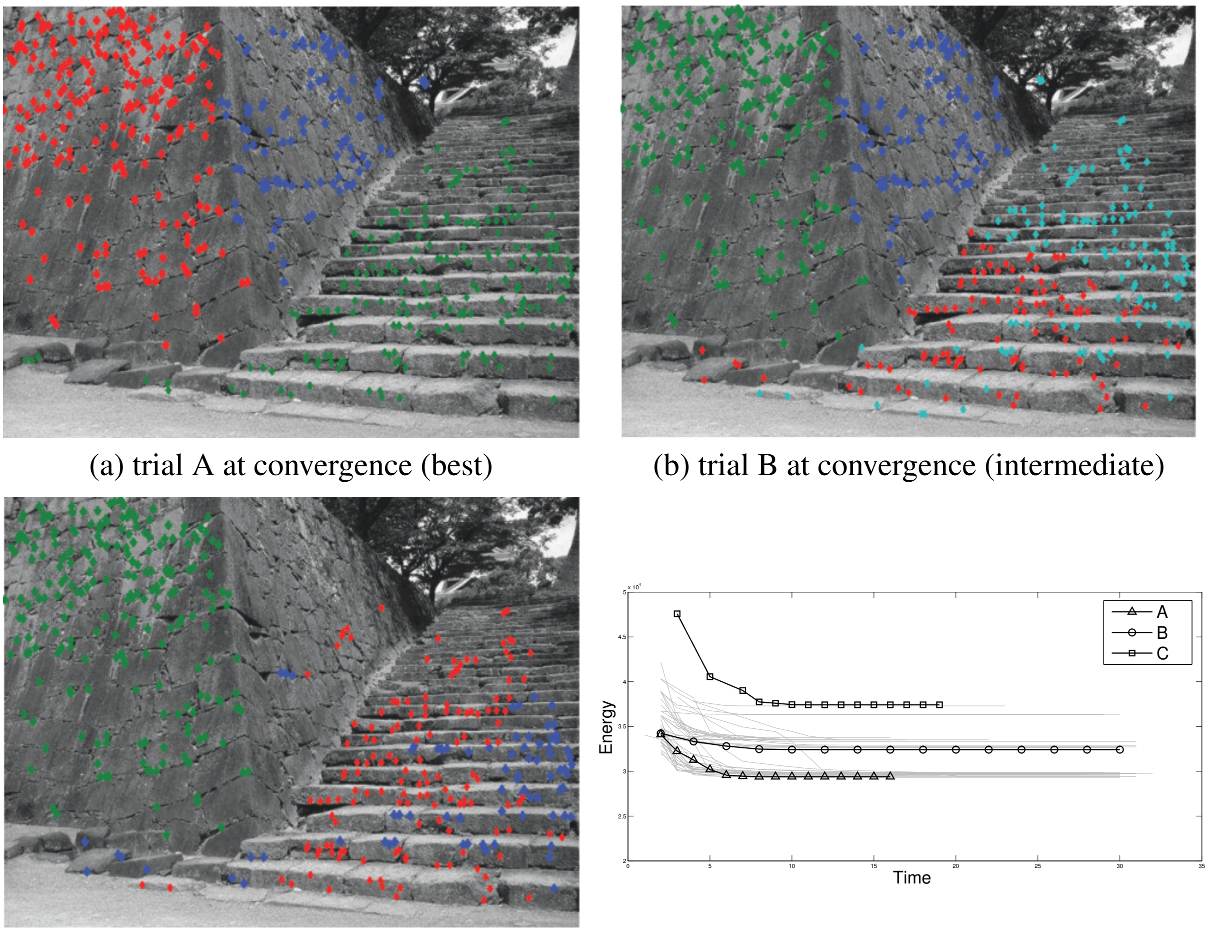
It should be emphasized that we are not first to suggest energies with label costs for segmentation. A large amount of related work on image segmentation is based on *minimum description length* (MDL) principle [107] which provides information theoretic foundation for regularization energies like (4.63). The MDL principle was first proposed for unsupervised segmentation by Leclerc [98]. As further detailed in [39], the specific technical realization of the MDL principle in [98] is distinct from ours. Leclerc derives energies somewhat different from (4.63) and optimizes them using continuation technique similar to *graduated nonconvexity* [16]. Further more, to simplify optimization [98] makes approximations, *e.g.* (2), that effectively ignore the label costs term.

Zhu & Yuille [166] used a continuous image segmentation energy inspired by MDL ideas of Leclerc. Specific formulation in [166] is much closer to ours and their functional is a continuous analogue of (4.63). They developed a *region competition* algorithm based on local contour evolution and explicit merging of adjacent regions to address the label cost term. A subsequent algorithm by Brox & Weickert [25] uses level sets to recursively partition the domain until it no longer pays to add regions (labels). Ben-Ayed & Mitiche [6] use multi-level sets to optimize an MDL-like region merging prior. Our work is first to demonstrate applications of powerful α-expansion approach to MDL-based image segmentation using energy (4.63).

4.7 Empirical Performance of Algorithms

This section presents an empirical comparison for several algorithmic variants to minimizing energy ($\star'$) where both smooth costs and label costs are present. In particular, we compare algorithms from Section 4.3 and several algorithms originally designed for spatial regularization functional (4.2) which can be applied to ($\star'$) using some *merging heuristics* as in [69]. Our goal is to compare running times and energy values obtained on real examples in the context of geometric model fitting described in Section 4.6.1.

Figure 4.16 illustrates our first homography fitting example (see Section 4.6.1). The curves in (d) show how the energy ($\star'$) decreases in 50 different tests running PEARL with the extended α-expansion algorithm from Section 4.3. Each test depends on some initial set of randomly sampled models. The algorithm can converge to different solutions illustrated in Figure 4.16a,b,c. Better results as in (a) correspond to solutions with lower energy values, and worse results as in (c) correspond to poor energy values. The black curve in Figure 4.17 is the average of 50 curves in Figure 4.16d. This section uses such average curves to compare different combinatorial algorithms for minimizing label cost energies. In addition to homography fitting results in Figure 4.17, we also use two rigid motion estimation examples (see Section 4.6.1) to compare similarly obtained average performance curves in Figure 4.18.



(a) trial A at convergence (best)

(b) trial B at convergence (intermediate)

(c) trial C at convergence (worst)

(d) energy plots for 50 different trials

Figure 4.16: Homography fitting example ("Stairs"). Different runs of the algorithm (PEARL with α++) in (d) converge to solutions with different energy values depending on a specific initial collection of randomly sampled models. As shown in (a-c), lower energy solutions correspond to better practical results.

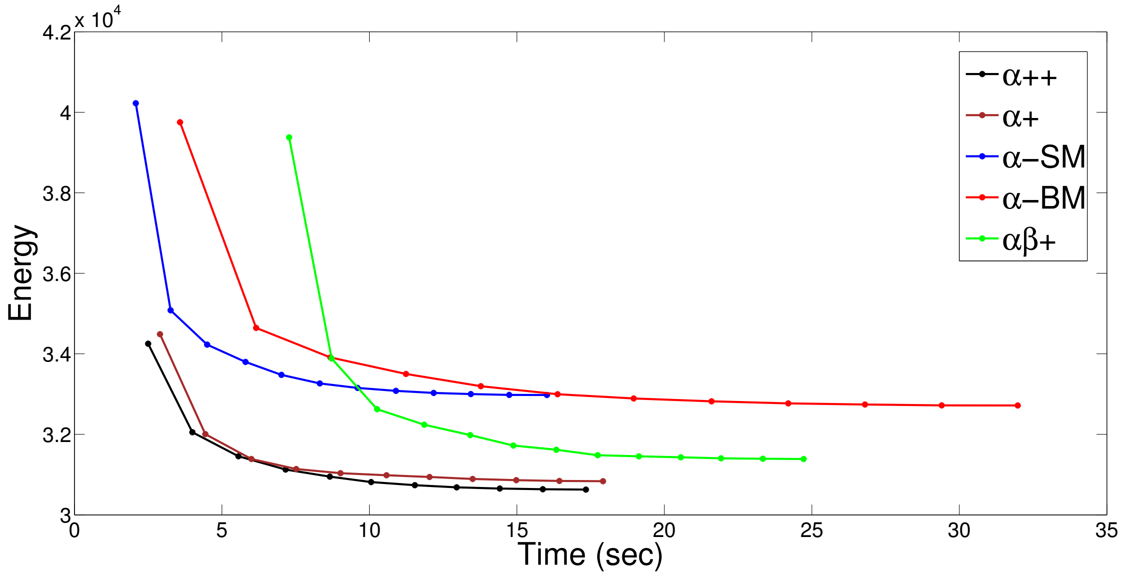


Figure 4.17: Homography fitting example ("Stairs") for different algorithms minimizing energy ($\star$): α++ and α+ are two versions of extended α-expansion from Section 4.3; $\alpha\beta$+ is a straightforward modification of the standard $\alpha\beta$-swap [24]; α-SM and α-BM are standard α-expansions with different merging heuristics [69]. The plots show values of energy ($\star'$) obtained after each iteration of segmentation and re-estimation, see Section 4.4. As in PEARL [69], the labels are initialized by randomly sampling 1000 models. Each plot above is obtained by averaging energy curves for 50 different initializations as in Figure 4.16d.

Now we briefly review combinatorial algorithms compared in this section. In contrast to other tested methods, the extended α-expansion algorithm from Section 4.3 directly addresses label costs in ($\star$) without any extra heuristics. We test two versions of the algorithm: α+ (basic) consistently iterates expansion steps over all labels, and α++ (adaptive) removes labels corresponding to empty expansions until the "last" iteration validating local minima with respect to all labels. Both versions have the same optimality guarantees (see Section 4.3). Our empirical results in Figures 4.17 and 4.18 suggest that α+ and α++ find solutions with comparable energy values. The adaptive method α++ converges faster.

Other tested methods are based on standard algorithms for energy (4.2) adapted to label cost in ($\star$) using some heuristics. For example, [69] uses basic α-expansion [24] for the first two terms in ($\star$) and adds a separate merging step to account for the label costs. Each merging step tries to replace some pair of labels A and B in the current solution with one label C. Two segments $\mathcal{A} = \{p : f_p = A\}$ and $\mathcal{B} = \{p : f_p = B\}$ are merged if and only if assigning some label C to combined segment $\mathcal{A} \cup \mathcal{B}$ lowers overall energy ($\star$). Note that merging decreases the second and the third terms in ($\star$) but it can increase the first (data) term. Iterating standard α-expansions with merging steps is guaranteed to decrease energy ($\star$) after each iteration. Note that separate merging steps for minimizing MDL-based functionals like ($\star$) were also used in [98, 166] in the context of *continuation* methods and *variational* approaches.

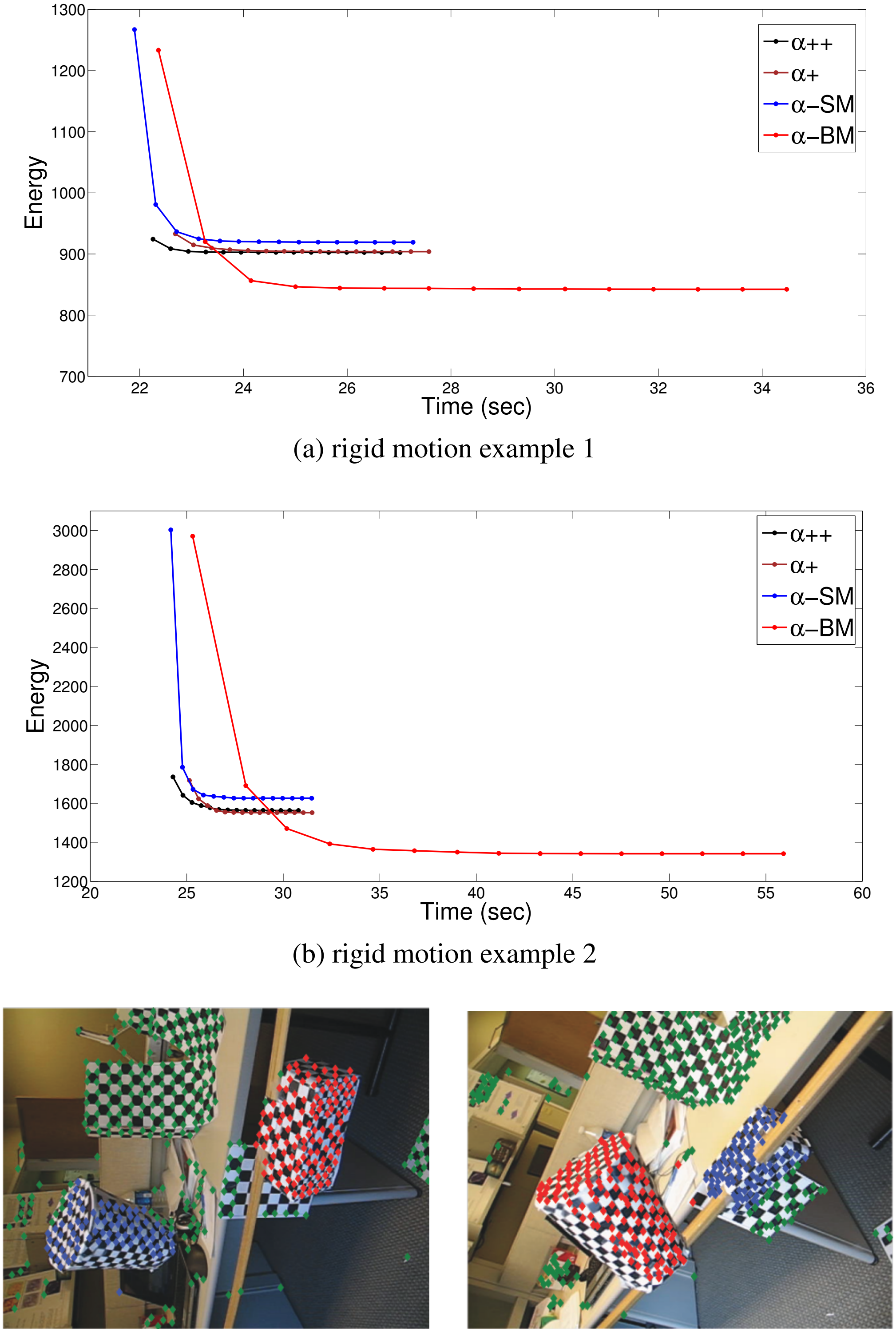
We tested two merging heuristics [69]: α-SM (simple merge) tries to merge two segments using $C = A$ or $C = B$, and α-BM (best merge) tries the optimal label C for two current segments $\mathcal{A} = \{p : f_p = A\}$ and $\mathcal{B} = \{p : f_p = B\}$

$$C^* = \arg\min_C \sum_{p \in \mathcal{A} \cup \mathcal{B}} D_p(C).$$

Due to extra optimization procedure α-BM is slower than α-SM but it generates lower energy values, see Figures 4.17 and 4.18.

We also note that the standard $\alpha\beta$-swap algorithm [24] was originally designed for smoothness energy ($\star$) but can be easily extended to label cost energy ($\star$). At each step the swap algorithm works with two fixed labels A and B and a region $\mathcal{A} \cup \mathcal{B}$. Only two trivial outcomes of a swap move change the label costs: when all nodes in $\mathcal{A} \cup \mathcal{B}$ are assigned either label A or B. The standard swap method does not account for the label cost term in ($\star$). Yet, it is easy to compare the outcome of an optimal $\alpha\beta$-swap move with two trivial solutions and choose one with the lowest value of energy ($\star$). We use symbol $\alpha\beta$+ to refer to this algorithm and its empirical results in Figure 4.17.

While discrete energy ($\star$) could be addressed by many combinatorial optimization techniques (*e.g.* [54, 98, 83]) or their modifications, our empirical evaluation is focused on graph cut methods that we consider more promising due to optimality guarantees associated with them. The experiments in Figure 4.17 show that α++, an adaptive version of extended α-expansion in Section 4.3, generated better quality solutions faster than other methods. Standard α-expansion with a "best merge" heuristic α-BM [69] obtained better energy values in Figure 4.18 but it was also much slower. Comparing α-BM with α-SM ("simple merge" version of the same algorithm) suggests that α-BM benefits from adaptive new model proposals. In fact, α-BM is the only method in our tests that used adaptively generated new proposals



(a) rigid motion example 1

(b) rigid motion example 2

representative results at convergence

Figure 4.18: Rigid motion estimation examples (Vidal's data set [145]) comparing different algorithms minimizing energy ($\star$): α++, α+, α-SM, α-BM. Algorithm $\alpha\beta$+ generated solutions with similar energy values but it was much slower than the other methods. Thus, we chose not to show its energy curves for the rigid motion estimation examples above.

in addition to basic model re-estimation. Note that rigid motion models in Figure 4.18 have higher dimensionality than (planar) homographies in Figures 4.16–4.17. Generating label proposals adaptively could be a practical mechanism improving exploration of larger label spaces of higher-dimensions.

Our general practical observation is that often all tested algorithms α++, α+, $\alpha\beta$+, α-SM, α-BM generate comparable results. In most cases, however, it is easier to use α++ as it is fast, robust, and does not rely on extra merging heuristics. In higher dimensional model-fitting problems the combination of PEARL and α++ may further benefit from additional application-specific mechanisms adaptively generating new model proposals.

4.8 Discussion

The potential applications of our algorithm are nearly as broad as for α-expansion. Our new algorithm can be applied whenever observations are known *a priori* to be positively correlated, for example in space or in time, whereas classical mixture model algorithms (Section 4.5) are largely designed for i.i.d. data.

Our C++ code and MATLAB wrapper are available at `http://vision.csd.uwo.ca/code/`. Besides minimizing general energy ($\star$) with α-expansion, the code is further optimized in two important special cases:

1. when the energy reduces to (4.1) the solution is computed by the greedy UFL algorithm (Section 4.3.5), and

2. when only a small fraction of labels are feasible for any given data point (*e.g.* geometric models; labels localized to a patch) we support "sparse data costs" to dramatically speed up computation.[6]

Our new α-expansion code optionally uses a simple strategy to invest expansions mainly on 'successful' labels. This is often faster, but can be slower, so we suggest selecting an expansion scheme (adaptive vs. standard cycle) empirically for each application.

Our energy is quite general but this can be a disadvantage in terms of speed. The α-expansion step runs in polynomial time for fixed number of positive $H(L)$ terms, but higher-order label costs should be used sparingly. Even the set of per-label costs $\{H(l)\}_{l\in\mathcal{L}}$ slows down α-expansion by 40–60%, but this is still relatively fast for such difficult energies [139]. This slowdown may be because the Boykov-Kolmogorov maxflow algorithm [22] relies on heuristics that do not work well for large cliques, *i.e.* subgraphs of the kind in Figure 4.4. Even if faster algorithms can be developed, our implementation can test the merit of various energies before one invests time in specialized algorithms.

Category costs Our high-order label costs (on *subsets* of labels) seem to be novel, both in vision and in terms of the UFL problem, and can be thought of as a type of co-occurrence potential first proposed in [38]. A natural application is to group labels in a hierarchy of categories

[6] Sparse data costs were not used in our experiments.

and assign a *category cost* to each. This encourages labelings to use fewer categories or, equivalently, to avoid mixing labels from different categories (*e.g.* kitchen, office, street, beach) unless the local evidence is strong enough. With respect to object recognition/segmentation with co-occurrence, similar costs were independently developed by Ladický *et al.* [94]. We foresee applications of high-order label costs in motion and homography estimation.

Relation to Ladický *et al.* [94] The application in [94] is object recognition with co-occurrance statistics. They are motivated by the principle of *parsimony:* if several segmentations explain the image equally well, then the one that requires the fewest object labels should be preferred. They develop an extension to α-expansion that is equivalent to our earlier work [38], but they also consider energies outside the class of co-occurrence potentials that we defined, *i.e.* beyond independent costs for each subset of labels costs. However, their class of energies is not submodular with respect to expansion and so they apply a heuristic with no guarantee of finding an optimal expansion move for energies outside the class of energies that we defined.

Relation to P^n Potts [79] The P^n Potts potential $\psi_P(f_P)$ is defined on clique $P \subseteq \mathcal{P}$ as

$$\psi_P(f_P) \overset{\text{def}}{=} \begin{cases} \gamma_\alpha & \text{if } f_p = \alpha \quad \forall p \in P \\ \gamma_{\max} & \text{otherwise} \end{cases}$$

where $\gamma_\alpha \leq \gamma_{\max}$ for all $\alpha \in \mathcal{L}$. This potential encodes a label-specific reward $\gamma_{\max} - \gamma_\alpha$ for clique P taking label α in its entirety, and acts either as simple high-order regularization (all $\gamma_\alpha = \text{const}$) or as a form of high-order data cost (label-specific γ_α).

Let $\bar{\alpha}$ denote the set of all labels except α, *i.e.* the set $\mathcal{L} \setminus \{\alpha\}$. A regional label subset cost over clique P can encode the P^n Potts potential in energy ($\star$) as follows:

1. Set cost $H_P(\bar{\alpha}) := \gamma_{\max} - \gamma_\alpha$ for each $\alpha \in \mathcal{L}$.
2. Add constant $(1 - |\mathcal{L}|)\gamma_{\max} + \sum_\alpha \gamma_\alpha$ to the energy.

Each regional label cost $H_P(\bar{\alpha})$ is non-negative by definition of $\psi_P(\cdot)$, thus a P^n Potts potential can be expressed as a sum of high-order label costs.

The P^n Potts potential and its robust generalization [80] were designed to encourage consistent labelings over specific regions in an image. A special case of our potentials is very closely related to the robust variant: a basic per-label potential $H(l) \cdot \delta_l(f)$ can be expressed as a specific (concave) Robust P^n Potts potential. Besides significant conceptual and motivational differences, the main technical difference is that our construction makes no reference to a "dominant label." By constructing a two-label Robust P^n Potts potential at each dynamic clique $\mathcal{P}_L$ in our binary expansion step, we can encode an arbitrary concave penalty on the number of variables taking labels from a specific *subset* of labels. This generalizes our high-order potentials $\delta_L(\cdot)$ if needed.

Learning label costs We studied label costs in an unsupervised setting where parameters are chosen based on information criteria or tuned manually. It is important to note that energy ($\star$) and the α-expansion-based inference algorithm can be used in supervised settings as well. The label cost terms are included in energy ($\star$) linearly and can thus be learned by max-margin methods [143, 146]. This approach was recently used for CRF learning, *e.g.* [141].

Chapter 5

Energies with Hierarchical Costs

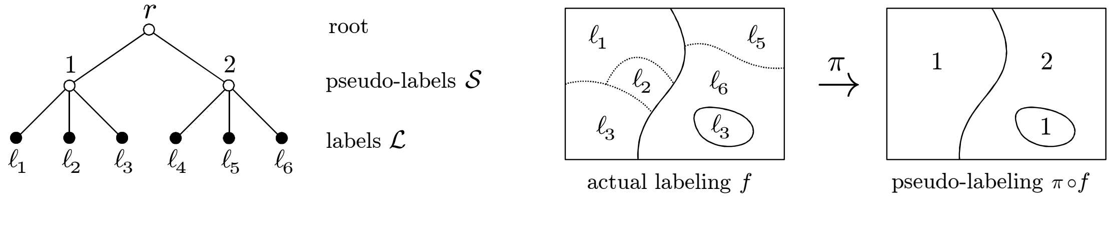
We present new theoretical results for an interesting class of energies that is closely related to that of Chapter 4. The key result is that by considering energies with a 'hierarchical' cost structure we can propose a new algorithm we call *hierarchical fusion* (h-fusion) that has better approximation guarantees than the classical α-expansion algorithm. It turns out that such energies can elegantly model problems in computer vision such as detecting multiple objects, motions, homographies, or repetitive patterns in image data. The improved theoretical guarantees are important because, in practice, α-expansion can easily get stuck in poor local minima for this useful class of energies.

We begin with a class of functionals $E(f)$ similar to the general form used in Chapter 4

$$E(f) \;=\; \sum_{p\in\mathcal{P}} D_p(f_p) \;+\; \sum_{pq\in\mathcal{N}} w_{pq}\cdot V(f_p,f_q) \;+\; \sum_{L\subseteq\mathcal{L}} H(L)\cdot\delta_L(f) \tag{5.1}$$

where $\mathcal{L}$ is the set of labels, weight $w_{pq} \geq 0$ scales the strength of V at edge pq, and $f : \mathcal{P} \to \mathcal{L}$ is a labeling. To describe an energy of the form (5.1) as having *hierarchical costs* we start by adding some *tree structure* defined over the label set. The root of the tree we denote by r, the leaves of the tree are the actual labels $\mathcal{L}$, and the remaining internal nodes (if any) belong to what we call the set of *pseudo-labels* $\mathcal{S}$. We express the tree structure by a child-to-parent map $\pi : \mathcal{L}\cup\mathcal{S} \to \mathcal{S}\cup\{r\}$, *i.e.* for any node $i \in \mathcal{L}\cup\mathcal{S}$ its parent node is given by $\pi(i)$. The functional $E(f)$ is still defined over labelings $f : \mathcal{P} \to \mathcal{L}$, but now there is some hierarchical *grouping* of these labels as illustrated by the example below.

Example 5.1. *A hierarchical grouping of six labels (left) into groups $\mathcal{S} = \{1, 2\}$ where, for example, $\pi(\ell_4) = 2$ and $\pi(2) = r$. At right is a possible labeling $f : \mathcal{P} \to \mathcal{L}$, and a coarser-level segmentation derived from grouping labels in f according to the tree.*


r
1
2
ℓ_1 ℓ_2 ℓ_3 ℓ_4 ℓ_5 ℓ_6
root
pseudo-labels $\mathcal{S}$
labels $\mathcal{L}$
ℓ_1 ℓ_5 ℓ_2 ℓ_6 ℓ_3 ℓ_3
actual labeling f
$\xrightarrow{\pi}$
1 2 1
pseudo-labeling $\pi\circ f$

Merely declaring the labels to be 'grouped' does not in itself change the energy $E(f)$, nor does is the α-expansion algorithm influenced by such label hierarchies. However, we will describe energies for which a 'good' tree can be defined so that our h-fusion algorithm is provably better than α-expansion. The methods we present are such that if we choose a flat tree ($\mathcal{S} = \{\}$) then all the results in this chapter reduce to those of Chapter 4 as a special case.

There are many problems for which the labels naturally form groups. In operations research, hierarchical variants of facility location problems have been studied [126, 138]. In computer vision, a recent trend is the use of *context* to resolve ambiguities in object recognition, *e.g.* [27, 94, 165]. The idea is that certain groups of labels are self-consistent because they appear in the same context, *e.g.* the *cup*, *keyboard*, *desk* labels all belong to the "office" context, while *sink*, *microwave*, *oven* labels all belong to the "kitchen" context. Or, in computer graphics, one may instead wish to automatically classify the faces of a 3D mesh into semantic parts for the benefit of artists and animators [76]. The part labels (*e.g.* arm, leg, tail, head, body) are naturally grouped by their context (humanoid, quadruped, chair). The illustration below helps to understand these two applications: object recognition on the left, mesh labeling on the right. Neither problem is fundamentally different from the other from an optimization point of view.

"office" context "street" context "horse" mesh "human" mesh
keyboard
coffee
desk
building
car
street
head
neck
torso
leg
tail
head
torso
upper arm
lower arm
hand
upper leg
lower leg
foot

The present chapter is essentially divided into two parts: smooth costs (Sections 5.1–5.4) and label costs (Section 5.5). The first four sections build concepts in hierarchical smooth costs only, omitting any mention of label costs for simplicity. The final section revisits the problem and introduces new definitions, theorems and pseudocode to handle hierarchical label costs. Section 5.1 defines a class of smooth cost potentials $V(\cdot,\cdot)$ that we call *hierarchical metric* (h-metric) potentials, while Section 5.2 discusses a simpler class we call *hierarchical Potts* (h-Potts) potentials. Section 5.3 then describes the steps of our h-fusion algorithm and shows that h-metrics fully characterize the class of smooth costs for which h-fusion is applicable. Section 5.4 derives approximation guarantees for h-fusion with respect to different choices of tree structure, and gives worst-case examples to show that our bounds are sufficiently tight. Section 5.5 introduces the class of *hierarchical* and *tree-structured* label costs, and generalizes our h-fusion algorithm and approximation bounds to incorporate such costs.

Related work The most similar work to ours, by far, is a recent paper by Kumar & Koller [92]. They use the concept of r-*hierarchically well-separated tree metrics* (r-HST metrics) [10] to arrive at an algorithm that is essentially the same as the h-fusion that we introduce in this chapter. They only consider smooth costs (no label costs) but their analysis is quite different, providing a number of very nice results on approximating arbitrary metrics (not just r-HST metrics). Our work was developed independently and, though our treatment of smooth costs is different and useful, our analysis of label costs is the most novel contribution at this point in time.

The main idea of an r-HST metric is as follows. Assume we are given a tree with distances $d(i,j)$ defined on each edge from child i to parent j. Further assume that we know the child-parent distance gets cheaper by a factor of r as we descend the tree, *i.e.* $\frac{d(i,j)}{d(k,i)} \geq r$ for some constant $r > 1$. The total distance between two leaf nodes α and β is the cumulative sum of edge distances along the path from α to β in the tree. If the 'costs' of a pairwise potential $V(\alpha,\beta)$ correspond to such a distance function for all α,β, then V is said to be an r-HST metric.

Our concept of an h-metric is expressed directly in terms of constraints on $V(\cdot,\cdot)$, not on edges or distances traversed in the tree. Furthermore, it is easy to find h-metrics for which there is no equivalent r-HST metric representation, but every r-HST metric is an h-metric. So, r-HST metrics are a special case of h-metrics.

5.1 Hierarchical Metrics (h-metrics)

From now until the beginning of Section 5.5 we concern ourselves with energies of the form

$$E(f) \;=\; \sum_{p\in\mathcal{P}} D_p(f_p) \;+\; \sum_{pq\in\mathcal{N}} w_{pq}\!\cdot\! V(f_p,f_q). \tag{5.2}$$

That is, we focus on energies with no label costs. This will help us to focus on the analysis of smooth costs and to present an initial version of the h-fusion algorithm that is much simpler.

We first introduce notation that is useful throughout the chapter when discussing trees. The following definitions are all with respect to some child-to-parent map $\pi : \mathcal{L} \cup \mathcal{S} \to \mathcal{S} \cup \{r\}$ and the tree that it defines. We use $\pi^n(i)$ to denote n applications of the parent function, as in $\pi(\cdots\pi(i))$.

Definition 5.2. *The set of children of node j is denoted by*

$$\mathcal{I}(j) = \{\, i \;:\; \pi(i) = j \,\}$$

Definition 5.3. *The set of all nodes in the subtree rooted at j is denoted by*

$$\mathrm{subtree}(j) = \{\, i \;:\; \pi^n(i) = j \;\textit{ for some }\; n \geq 0 \,\}$$

Definition 5.4. *The set of labels (leaves) belonging to the subtree of node i is*

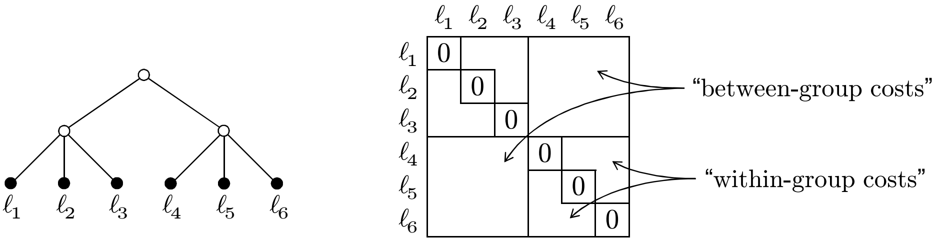
$$\mathcal{L}_i = \{\, \ell \in \mathcal{L} \;:\; \ell \in \mathrm{subtree}(i) \,\}$$

Definition 5.5. *The lowest common ancestor (lca) of nodes i, i' is*

$$\mathrm{lca}(i,i') = j \;\textit{ such that }\; j = \pi^n(i) = \pi^m(i') \;\textit{ for minimal }\; n,m.$$

With these definitions in mind, consider what it means to apply the tree structure shown in Example 5.1 over the labels. A hierarchical grouping of labels $\ell \in \mathcal{L}$ induces a grouping of the smooth cost values inside each $V(\ell,\ell')$ potential. Looking at the tree structure in Example 5.1

we can say that labels ℓ_1 and ℓ_2 are in the same group whereas ℓ_1 and ℓ_4 are from different groups; thus $V(\ell_1, \ell_2)$ can be interpreted as a "within-group cost" and $V(\ell_1, \ell_4)$ as a "between-group cost." The elements of V's smooth cost matrix are thus broken up into 'within-group' and 'between-group' entries. A simple example is given below, where the different regions of the $|\mathcal{L}| \times |\mathcal{L}|$ matrix are delineated by black lines and we assume $V(\alpha, \alpha) = 0$.



Note that the costs in each block need not be constant, though we consider this case in Section 5.2.

What follows next is the key definition of this section. Later in Section 5.3 we will show that, if a smooth cost potential V satisfies the constraint below with respect to some tree structure, the corresponding energy can be approximately minimized by our h-fusion algorithm. In fact we show that the characterization below is sufficient *and* necessary to be optimized by h-fusion.

Definition 5.6. *We say that pair (V, π) forms an h-metric potential if π defines an irreducible*[1] *tree and for every node $i \in \mathcal{L} \cup \mathcal{S}$ we have*

$$V(\alpha_1, \alpha_2) + V(\beta, \gamma) \le V(\alpha_1, \gamma) + V(\beta, \alpha_2) \quad \forall\, \alpha_1, \alpha_2 \in \mathcal{L}_i,\ \beta, \gamma \in \mathcal{L}_{\pi(i)} \setminus \mathcal{L}_i \tag{5.3}$$

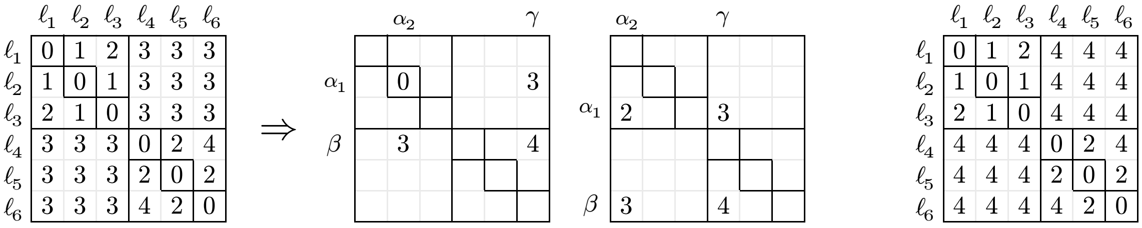
Note that for the special case of a 'flat' tree where $\mathcal{S} = \{\}$ then each set $\mathcal{L}_i$ in (5.3) contains only a single label $\{i\}$ and each $\mathcal{L}_{\pi(i)} = \mathcal{L}$. In that case we have $\alpha_1 = \alpha_2 = i$ and $\beta, \gamma \in \mathcal{L} \setminus \{i\}$, so the h-metric constraint reduces to the classic metric constraint for α-expansion of $V(\alpha, \alpha) + V(\beta, \gamma) \le V(\alpha, \gamma) + V(\beta, \alpha)$ given in [24].

Though the definition of an h-metric is important, the constraints involved can be more complex than for standard α-expansion. Figure 5.1 shows a concrete example of an h-metric (V, π) and suggests a way to visualize constraints (5.3) as a pattern in the matrix. However, we can express a condition that is simpler and yet sufficient for a pair (V, π) to form an h-metric. First we define two quantities that will be useful in subsequent proofs.

Definition 5.7. *Given a smooth cost potential V and tree structure defined by child-to-parent map π we define quantities $V_i^{\max}$ and $V_i^{\min}$ for each $i \in \mathcal{L} \cup \mathcal{S}$*

$$V_i^{\max} = \max_{\alpha, \beta \in \mathcal{L}_i} V(\alpha, \beta) \qquad V_i^{\min} = \min_{\substack{\gamma, \zeta \in \mathcal{L}_i \\ \mathrm{lca}(\gamma, \zeta) = i}} V(\gamma, \zeta).$$

[1] A tree is irreducible if all its internal nodes have at least two children, *i.e.* there are no 'redundant' parent nodes and so $\mathcal{L}_{\pi(i)} \setminus \mathcal{L}_i$ is never an empty set.



	ℓ_1	ℓ_2	ℓ_3	ℓ_4	ℓ_5	ℓ_6
ℓ_1	0	1	2	3	3	3
ℓ_2	1	0	1	3	3	3
ℓ_3	2	1	0	3	3	3
ℓ_4	3	3	3	0	2	4
ℓ_5	3	3	3	2	0	2
ℓ_6	3	3	3	4	2	0

(a) example matrix

(b) example constraint configurations

	ℓ_1	ℓ_2	ℓ_3	ℓ_4	ℓ_5	ℓ_6
ℓ_1	0	1	2	4	4	4
ℓ_2	1	0	1	4	4	4
ℓ_3	2	1	0	4	4	4
ℓ_4	4	4	4	0	2	4
ℓ_5	4	4	4	2	0	2
ℓ_6	4	4	4	4	2	0

(c) alternate matrix

Figure 5.1: Smooth cost matrix (a) is a concrete example of an h-metric when entries are grouped according to the tree structure π from Example 5.1. A valid h-metric must satisfy all constraints of the form (5.3), and (b) shows two such constraints satisfied: $0+4 \leq 3+3$ is indeed true, and $2+4 \leq 3+3$ as well. Standard α-expansion requires the former constraint, while only h-fusion on a tree such as π requires the latter constraint. The smooth cost matrix (c) can be easily checked as h-metric via Lemma 5.8, whereas (a) fails Lemma 5.8 yet is still an h-metric according to the full Definition 5.6.

In other words, $V_i^{\max}$ is the maximum cost for any pair of labels in the subtree of node i, and $V_i^{\min}$ is the minimum cost for two labels from *different* subtrees descended from i. For example, in Figure 5.1(a) one can see that the first parent node s_1 has $V_{s_1}^{\min} = 1$ and $V_{s_1}^{\max} = 2$. Similarly, for the root node r, this example has $V_r^{\min} = 3$ and $V_r^{\max} = 4$.

Lemma 5.8. *The following condition is sufficient for (V, π) to form an h-metric potential:*

$$V_j^{\max} - V_j^{\min} \leq V_j^{\min} - V_i^{\max} \qquad \forall\, i \in \mathcal{I}(j) \tag{5.4}$$

Proof. We must show that if (5.4) holds then so does (5.3). Since we know for any $\alpha_1, \alpha_2 \in \mathcal{L}_i$ and $\beta, \gamma \in \mathcal{L}_{\pi(i)} \setminus \mathcal{L}_i$ it follows that $\operatorname{lca}(\alpha_1, \gamma) = \operatorname{lca}(\beta, \alpha_2) = \pi(i)$. By minimizing over all such possible cases, each term on the right-hand side of (5.3) can be bounded from below by $V_{\pi(i)}^{\min} = V_j^{\min}$. Likewise, the two terms on the left-hand side of (5.3) can be bounded from above by $V_i^{\max}$ and $V_j^{\max}$ respectively. We therefore have that $V_i^{\max} + V_j^{\max} \leq 2V_j^{\min}$ implies inequality (5.3). Rearranging terms completes the proof. ■

Again, inequality (5.4) is sufficient but not necessary for (V, π) to be an h-metric, as shown by comparing Figures 5.1(a) and 5.1(c). Roughly speaking, Lemma 5.8 suggests two things: that if the within-group costs ($V_i^{\max}$) are cheaper than the between-group costs ($V_j^{\min}$) then (V, π) is more likely to form an h-metric, and that if the *range* of between-group costs is large then the maximal within-group costs should be cheaper by a corresponding amount.

5.2 Hierarchical Potts (h-Potts)

The *Potts* potential is now standard for energy-based models computer vision, even though it was originally proposed in statistical mechanics by R. B. Potts [119]. A pairwise Potts potential is of the form $V(\alpha, \beta) = w \cdot \delta(\alpha \neq \beta)$ where δ is 1 if the condition is true, 0 otherwise, and w is the cost that must be paid for $\alpha \neq \beta$. In computer vision the Potts potential is typically used

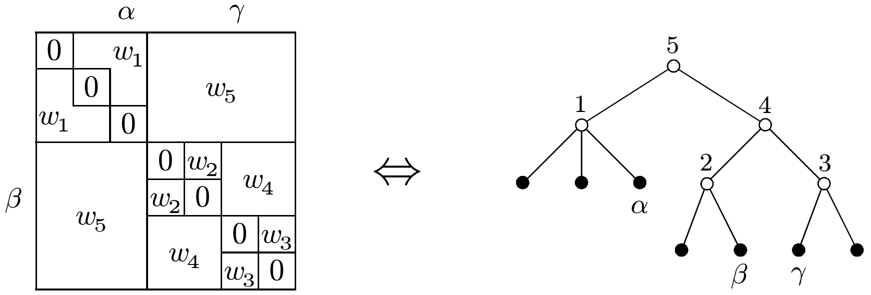


Figure 5.2: The smooth cost matrix for an h-Potts potential (left) comprises a number of regions with constant cost. The structure of these regions is determined by the tree structure π. In this example, $V(\alpha,\beta) = w_5$ and $V(\beta,\gamma) = w_4$. The tree structure imposes a nested hierarchy of Potts potentials, where the cost paid depends on the 'level' at which the discontinuity occurs.

in energies of the form

$$E(f) \;=\; \sum_{p\in\mathcal{P}} D_p(f_p) \;+\; \sum_{pq\in\mathcal{N}} w_{pq}\cdot\, \delta(f_p \neq f_q) \tag{5.5}$$

where weight $w_{pq} \geq 0$ scales the attraction strength between variables p and q. The larger the weight, the stronger the preference for choosing $f_p = f_q$ in the optimal labeling. Apart from preferring $f_p = f_q$ the Potts potential is agnostic about the remaining possibilities, *i.e.* the same cost w_{pq} is paid for all cases $f_p \neq f_q$.

We now introduce a natural class of smooth costs that we call *hierarchical Potts* (h-Potts) potentials. A pairwise h-Potts potential is defined by a pair (V,π) where the cost $V(\alpha,\beta)$ depends only on the lowest common ancestor of the labels α and β.

Definition 5.9. *We say that pair (V,π) forms an h-Potts potential if π is irreducible and V is defined by a set of constants $\{w_i\}_{i\in\mathcal{L}\cup\mathcal{S}\cup\{r\}}$ such that*

$$V(\alpha,\beta) = w_{\mathrm{lca}(\alpha,\beta)} \quad \forall\,\alpha,\beta\in\mathcal{L} \tag{5.6}$$

where $w_\ell = 0$ for all $\ell\in\mathcal{L}$.

Figure 5.2 shows the piecewise-constant structure of an h-Potts potential (V,π) for a particular tree. Just as for h-metrics, the most natural application of h-Potts potentials is when labels are somehow grouped. Indeed, any h-Potts potential that satisfies constraints (5.3) is also an h-metric potential, but for h-Potts we can express the constraints in a simpler manner. It is this simplicity, along with the ubiquity of Potts in computer vision, that makes h-Potts worth characterizing separately from h-metrics.

First, we characterize the circumstances under which standard α-expansion is applicable to an h-Potts potential (V,π). This establishes the full class of possible h-Potts potentials as a subclass of *metrics* as defined in Chapter 2. Recall that a pairwise potential V can be optimized by α-expansion if and only if it satisfies the metric constraint, restated here:

$$V(\alpha,\alpha) + V(\beta,\gamma) \leq V(\alpha,\gamma) + V(\beta,\alpha) \quad \forall\,\alpha,\beta,\gamma\in\mathcal{L}. \tag{5.7}$$

Theorem 5.10. *An h-Potts potential (V, π) is metric on $\mathcal{L}$ if and only if*

$$w_i \leq 2w_j \quad \forall\, i \in \mathrm{subtree}(j). \tag{5.8}$$

Proof. In the case of an h-Potts potential, the metric constraint is equivalent to

$$0 + w_{\mathrm{lca}(\beta,\gamma)} \leq w_{\mathrm{lca}(\alpha,\gamma)} + w_{\mathrm{lca}(\beta,\alpha)} \quad \forall\, \alpha, \beta, \gamma \in \mathcal{L} \tag{5.9}$$

Because π defines a tree structure, for every α, β, γ there exists $i, j \in \mathcal{S} \cup \{r\}$ such that, without loss of generality,

$$\begin{aligned} j &= \mathrm{lca}(\alpha,\gamma) = \mathrm{lca}(\beta,\alpha), \text{ and} \\ i &= \mathrm{lca}(\beta,\gamma) \text{ such that } i \in \mathrm{subtree}(j). \end{aligned} \tag{5.10}$$

We can assume this arrangement because (α, β, γ) form what is sometimes called a *rooted triple* and there can only be up to two unique lowest common ancestors among them. We assume ancestor i is in the sub-tree rooted at ancestor j, and possibly equal to j. For any particular (α, β, γ) and corresponding (i, j) inequality (5.9) is equivalent to $w_i \leq 2w_j$. Since π defines an irreducible tree, for each (i, j) there must exist corresponding sub-labels (α, β, γ) for which (5.9) holds. It follows that $w_i \leq 2w_j$ holds for all pairs (i, j) where $i \in \mathrm{subtree}(j)$ and completes the proof of (5.8). ■

The above result is useful when we (a) know that labels are grouped in some manner, and (b) wish to learn the ideal potentials from training data. Hierarchical Potts allows us to learn only a few coefficients $\{w_i\}$ rather than a full set of $|\mathcal{L}|^2$ smooth cost entries, while Theorem 5.10 gives us simple constraints for optimizing with α-expansion.

However, an h-Potts potential being a *metric* does not necessarily imply that it is an h-*metric*, which is necessary in order to apply our improved h-fusion algorithm. The following theorem gives a simple, sufficient condition for h-Potts potentials to be optimized by h-fusion. It says that, so long as discontinuity costs never decrease as we ascend the tree, then the energy can be optimized by h-fusion.

Theorem 5.11. *The following condition is sufficient for an h-Potts potential (V, π) to form an h-metric:*

$$w_i \leq w_j \quad \forall\, i \in \mathrm{subtree}(j). \tag{5.11}$$

Proof. For an h-Potts potential we have $V_j^{\min} = w_j$ for any node j. For an h-Potts potential that satisfies (5.11) we have in addition that $V_j^{\max} = \max_{i \in \mathrm{subtree}(j)} w_i = w_j$. Inequality (5.4) then becomes $w_j - w_j \leq w_j - w_i$ which is equivalent to $w_i \leq w_j$. Since by assumption (5.11) this holds for all $i \in \mathrm{subtree}(j)$ then clearly it holds for all $i \in \mathcal{I}(j)$ and so by Lemma 5.8 any such h-Potts potential is also an h-metric. ■

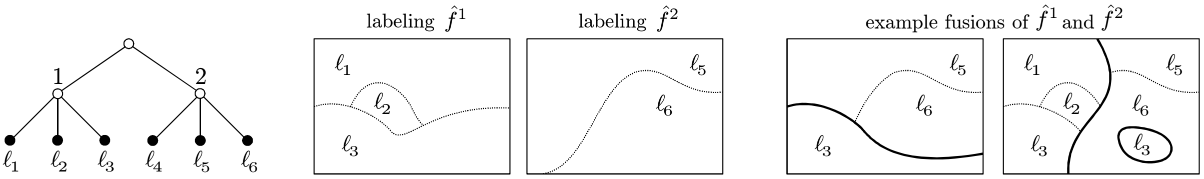


Figure 5.3: Given two complete labelings $\hat{f}^1$ and $\hat{f}^2$ shown above, here with $\hat{f}^1_p \in \{\ell_1, \ell_2, \ell_3\}$ and $\hat{f}^2_p \in \{\ell_4, \ell_5, \ell_6\}$, there are many ways to fuse or 'stitch' them together. A fusion is a new labeling f where each $f_p \in \{\hat{f}^1_p, \hat{f}^2_p\}$ (examples at right). Here we are fusing only two labelings, but in general any number of labelings may be combined via iteration.

5.3 Hierarchical Fusion with Smooth Costs

We now explain the steps of our *hierarchical fusion* (h-fusion) algorithm, a better alternative to standard α-expansion. Roughly speaking, the α-expansion procedure (Section 2.2.2) minimizes a multi-label energy by solving a sequence of special binary energies. The main idea of h-fusion is to create a sequence of special *multi-label* energies, each of which is solved by running α-expansion as a subroutine. These intermediate multi-label energies are designed to 'stitch' or to 'fuse' a particular set of existing labelings that were computed earlier, in a bottom-up fashion. As we will show in Section 5.4, our h-fusion procedure turns out to provide better optimality guarantees than α-expansion for energies with 'hierarchical' cost structure.

We call our algorithm h-fusion because our key multi-label energy at each step can be interpreted as performing *fusion moves*. The concept of a fusion move was introduced by Lempitsky *et al.* [100] as a powerful way to minimize energies $E(f)$ of the form (5.2). The main idea is that, given a two candidate labelings $\hat{f}^1$ and $\hat{f}^2$, a lower-energy labeling can be found by 'fusing' the best parts of each into a composite labeling. Figure 5.3 illustrates how new labelings are generated this way in our hierarchy. The key insight of [100] is that all fusion moves are of the form $\mathbf{x}\hat{f}^1 + \bar{\mathbf{x}}\hat{f}^2$ where $\mathbf{x}$ is a vector of binary variables, and that the *best* fusion move can be computed by minimizing an energy $E_{\text{fuse}}(\mathbf{x}) = E(\mathbf{x}\hat{f}^1 + (1-\mathbf{x})\hat{f}^2)$. Such a binary energy can also be expressed as

$$E_{\text{fuse}}(\mathbf{x}) = \sum_{p \in \mathcal{P}} D'_p(x_p) + \sum_{pq \in \mathcal{N}} V'_{pq}(x_p, x_q) \tag{5.12}$$

where $D'_p(x) = D_p(x\hat{f}^1_p + \bar{x}\hat{f}^2_p)$ and[2] $V'_{pq}(x,y) = w_{pq} \cdot V(x\hat{f}^1_p + \bar{x}\hat{f}^2_p, y\hat{f}^1_q + \bar{y}\hat{f}^2_q)$. In other words, each binary x_p selects between $\hat{f}^1_p$ and $\hat{f}^2_p$. Note that this is extremely similar to the binary energy at the heart of α-expansion. Indeed, an "expansion move on label α" can now be thought of as a fusion move where $\hat{f}^1$ is the current labeling and we simply set $\hat{f}^2 = (\alpha, \alpha, \ldots)$, *i.e.* a constant labeling. Lempitsky *et al.* even propose using fusion moves as a way to parallelize α-expansion by dividing the labels according to the number of CPUs.

However, the paper by Lempitsky *et al.* [100] assumes that the candidate labelings can be arbitrary and therefore the binary energy (5.12) is *non-submodular* in practice (see Chapter 2 for review). This makes each fusion step potentially $\mathcal{NP}$-hard and so it cannot be solved

[2] Note that here w_{pq} is just the weight of edge pq and should not be confused with an h-Potts coefficient w_i.

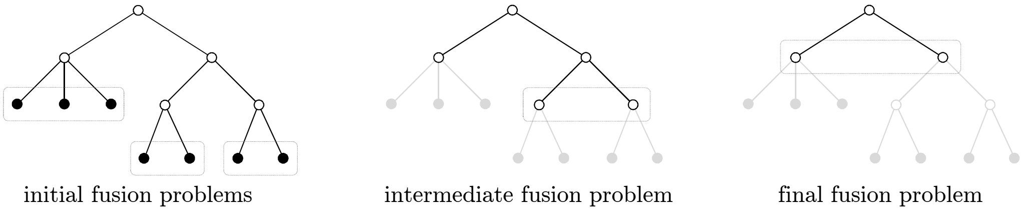


Figure 5.4: The hierarchical fusion process for a particular tree. The first sub-problems combine groups of actual labels (leaves) in a pure α-expansion procedure (left). The remaining sub-problems involve fusing the existing labelings in a bottom-up fashion throughout the tree (center) until finally the root labeling has been computed from its children (right). Each sub-problem is solved by invoking α-expansion on a special multi-label fusion energy that is based on the labelings involved.

by a graph cut, unlike for α-expansion. They therefore resort to a heuristic method known as QPBO [85, 124] which has no guarantee of finding a complete solution. In other words, the generality of their formulation precludes any approximation guarantees for the larger algorithm.

In contrast, we introduce h-fusion algorithm that uses an energy-driven criterion (good choice of tree structure) to determine a sequence of multi-label fusion energies. Furthermore, we introduced the concept of an h-metric to explicitly characterize the conditions for which h-fusion is guaranteed to have *submodular* binary moves. Submodularity allows the globally optimal fusion move to be computed by a single graph cut, just as for α-expansion. This will allow us in Section 5.4 to prove approximation guarantees for h-fusion that generalize—and improve upon—the well-known guarantees of α-expansion. Furthermore, we explain how to incorporate label costs into the fusion steps; to the best of our knowledge, we are the first to incorporate such high-order energy potentials into a fusion-based algorithm.

Our h-fusion algorithm is a simple recursive procedure that builds the final labeling in bottom-up fashion. There is one multi-label energy to solve for each internal node $j \in \mathcal{S} \cup \{r\}$ in the tree, and each of these "fusion energies" is solved (up to a local minimum) by running α-expansion. The local minima computed at one level of the tree are subsequently fused at the next-higher level of the tree until the root labeling has been computed. Figure 5.4 shows the order of sub-problems on a simple tree structure. The pseudocode for our recursive h-FUSION(E, π, j) procedure is given below, where $E = (D, V)$ is the original multi-label energy to be optimized, π defines the label hierarchy, and j is the tree node whose labeling is to be computed. The h-fusion process is initiated by calling h-FUSION(E, π, r) where r is the root of the label hierarchy.

```
$h$-FUSION$(E, \pi, j)$
1 if $j \in \mathcal{L}$
2     return $(j, j, \ldots)$                    // $j$ is an actual label, so return constant labeling
3 for $i \in \mathcal{I}(j)$
4     $\hat{f}^i := h$-FUSION$(E, \pi, i)$        // compute child labelings (parallel loop)
5 $E^j :=$ SETUPFUSION$(E, \pi, j)$
6 $\hat{g} := \alpha$-EXPANSION$(E^j)$            // compute map $\hat{g} : \mathcal{P} \to \mathcal{I}(j)$ that fuses the $\hat{f}^i$
7 $\hat{f}^j_p := \hat{f}^{\hat{g}_p}_p \quad \forall\, p \in \mathcal{P}$      // convert child indices to raw label indices $\mathcal{L}$
8 return $\hat{f}^j$
```

```
SETUPFUSION$(E, \pi, j)$ without label costs
$D'_p(i) := D_p(\hat{f}^i_p)$    $\forall\, p \in \mathcal{P}, i \in \mathcal{I}(j)$    // configure $\alpha$-expansion to index $\mathcal{I}(j)$
$V'_{pq}(i, i') := w_{pq} \cdot V(\hat{f}^i_p, \hat{f}^{i'}_q)$    $\forall\, pq \in \mathcal{N},\ i, i' \in \mathcal{I}(j)$
return $E' = (D', V')$
```

In h-fusion, the main subproblem (line **6**) calls α-expansion to fuse a set of labelings $\{\hat{f}^i\}_{i \in \mathcal{I}(j)}$. In this scenario α-expansion is selecting a child index from $\hat{g}_p \in \mathcal{I}(j)$ for each pixel—it does *not* select from actual labels $\mathcal{L}$ until the recursion reaches the bottom of the tree (Figure 5.4, left). This temporary child-index map $\hat{g}$ must therefore be converted into an actual labeling $\hat{f}^j$ associated with tree node j (line **7**).

The h-fusion process must terminate because there are $|\mathcal{S} \cup \{r\}|$ calls to α-expansion. The α-expansion algorithm is guaranteed to reduce the energy at each internal step and therefore terminates at a local minimum in finite time [24]. In practice, α-expansion is the basis of some of the fastest optimization algorithms for these kinds of energies [139, 4]. There is every reason to expect h-fusion to terminate as fast as standard α-expansion, if not faster. First, the total number of nodes at each level of the tree shrinks by at least a factor of two (often much more) as we ascend the hierarchy, so the total number of α-expansion invocations is $O(|\mathcal{L}|)$ and each invocation is a much smaller problem than running raw α-expansion. Second, since the label hierarchy has a tree structure, the for-loop (line **4**) can be executed in parallel. This allows for a number of simultaneous instances of α-expansion, each fusing a different subset of nodes, ensuring full utilization of CPU resources on modern multi-core systems[3].

We now arrive at a key result that precisely characterizes the class of smooth costs that h-fusion can handle.

Theorem 5.12. *The h-fusion algorithm is applicable for smooth cost potential V using label hierarchy π if and only if (V, π) forms an h-metric.*

[3] However, parallel max-flow/min-cut algorithms such as [35, 135, 104] are still of importance, even in our scenario. In a shared-memory architecture, if each thread frequently reads/writes to memory outside its cache (large working-set size) then bus contention inhibits parallelism. Each thread in a parallel max-flow algorithm has a smaller working-set size than do many threads each with their own full graph to handle. One should therefore consider a balanced approach when experimenting with parallel graph cut techniques.

Proof. As discussed in [24, 87], a metric V can be optimized by α-expansion if and only if it satisfies

$$V(\alpha,\alpha)+V(\beta,\gamma)\le V(\alpha,\gamma)+V(\beta,\alpha)\quad \forall\,\alpha\in\mathcal{L},\ \beta,\gamma\in\mathcal{L}\setminus\{\alpha\} \tag{5.13}$$

The above constraint is usually stated as holding for arbitrary $\alpha,\beta,\gamma\in\mathcal{L}$, rather than only for $\beta,\gamma\in\mathcal{L}\setminus\{\alpha\}$. Although this distinction does not matter for α-expansion, it matters for h-fusion and so we explicitly start from (5.13).

In the h-fusion case, each local fusion metric V'_{pq} on line **2** of SETUPFUSION must satisfy this constraint and so α-expansion can fuse a collection of labelings $\{\hat{f}^i\}_{i\in\mathcal{I}(j)}$ if and only if

$$V(\hat{f}^i_p,\hat{f}^i_q)+V(\hat{f}^{i'}_p,\hat{f}^{i''}_q)\le V(\hat{f}^i_p,\hat{f}^{i''}_q)+V(\hat{f}^{i'}_p,\hat{f}^i_q)\quad \forall\,i\in\mathcal{I}(j),\ i',i''\in\mathcal{I}(j)\setminus\{i\} \tag{5.14}$$

Note that $\hat{f}^i_p$ and $\hat{f}^i_q$ could each be any label in $\mathcal{L}_i$ and are *not* necessarily identical, unlike the α-expansion case. The constraints on the original metric V for h-fusion will therefore be more restrictive than for α-expansion. Since inequality (5.14) must hold for all possible labelings $\hat{f}^i, \hat{f}^{i'}$, and $\hat{f}^{i''}$ then it is equivalent to

$$V(\alpha_1,\alpha_2)+V(\beta,\gamma)\le V(\alpha_1,\gamma)+V(\beta,\alpha_2)\quad \forall\,\alpha_1,\alpha_2\in\mathcal{L}_i,\ \beta,\gamma\in\mathcal{L}_j\setminus\mathcal{L}_i \tag{5.15}$$

Since $j=\pi(i)$ then inequalities (5.15) are identical to (5.3), completing the proof. ■

Note that for any metric V we can choose a flat tree structure, *i.e.* $\pi(\ell)=r$ for all $\ell\in\mathcal{L}$, so that (V,π) forms an h-metric. Applying h-fusion in this degenerate case is identical to the standard α-expansion algorithm, so h-fusion inherits at least one basic approximation guarantee. However, for many potentials V a different choice of tree structure yields a better approximation guarantee for our h-fusion algorithm, as we shall prove in Section 5.4.

5.4 Approximation Bound of h-Fusion (without label costs)

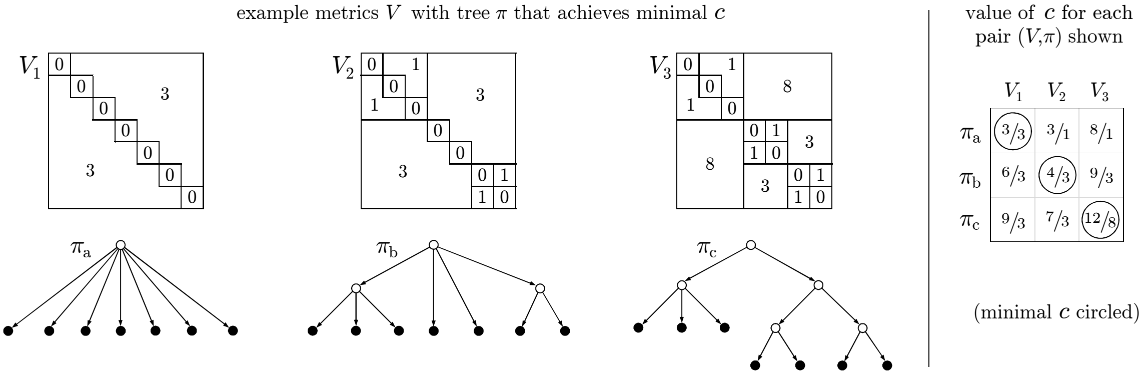
Our goal is to derive an optimality bound, and approximation guarantee, analogous to the ones for standard α-expansion derived in [24] and in Chapter 4. Like α-expansion's bound, the quality of our new bound is energy-dependent, and therefore instructs us on a case-by-case basis as to how much confidence we should place in the algorithm. Our improved bound generalizes that of α-expansion but is more complex to express. We begin by defining some quantities that are useful for understanding the nature of h-fusion.

Definition 5.13. *Given an h-metric (V,π) we define the additional quantities*

$$b_j=a_j+\max_{i\in\mathcal{I}(j)} b_i \qquad c=\max_{j\in\mathcal{S}\cup\{r\}}\frac{b_j}{d_j}$$

where we use shorthand $a_j=V^{\max}_j$ and $d_j=V^{\min}_j$ (see Definition 5.7).

The quantity c is most important because we will use it to bound the worst-case approximation error of the h-fusion algorithm.


example metrics V with tree π that achieves minimal c
V_1
V_2
V_3
π_a
π_b
π_c

value of c for each pair (V,π) shown

	V_1	V_2	V_3
π_a	$3/3$	$3/1$	$8/1$
π_b	$6/3$	$4/3$	$9/3$
π_c	$9/3$	$7/3$	$12/8$

(minimal c circled)

Figure 5.5: The top row shows three example smooth cost matrices. The first is a standard (flat) Potts potential with penalty $V_1(\ell, \ell') = 3$ for any $\ell \neq \ell'$ and so $c = 1$ for a flat tree π_a. Metrics V_2 and V_3 have varying penalties and so a flat tree yields $c = 5$ and $c = 8$ respectively. However, by applying h-fusion on tree structures π_b and π_c respectively (bottom row), we can achieve better c for these particular metrics. The table at right shows other values of c and demonstrates that the choice of tree is very important for achieving a good bound.

Observation 5.14. *If π defines a 'flat' tree, i.e. $\mathcal{S} = \{\}$ and so all $\pi(\ell) = r$, then constant c in Definition 5.13 reduces to the same constant used in the well-known α-expansion bound:*

$$c = \frac{\max_{\alpha,\beta\in\mathcal{L}} V(\alpha,\beta)}{\min_{\gamma\neq\zeta\in\mathcal{L}} V(\gamma,\zeta)}$$

As we will show, the main benefit of h-fusion is that, when the ratio c above is large for standard α-expansion, choosing a non-flat tree structure π can result in a much smaller constant and thereby a better approximation guarantee than attainable through α-expansion alone. The easiest way to understand how V and π affect bound c is by looking at a few numeric examples. Figure 5.5 examines specific values of c for various pairs of smooth cost matrices V and tree structures π. These examples suggests that for each V one wishes to optimize there exists an optimal choice of π to give the best approximation guarantee. Since for every metric V we can use a flat tree, then there always exists a tree for which h-fusion's guarantees are as good or better than those of α-expansion.

We now state the formal property of h-fusion that makes it an improvement over standard α-expansion, namely a better approximation guarantee when V affords an h-metric (V, π) with small coefficient c as suggested by Figure 5.5. In what follows we assume that energy (5.2) is configured so that $D_p \geq 0$, V is a metric [24], and thus $E(f) \geq 0$.

Theorem 5.15. *If f^* is a global minimum of energy* (5.1) *where all $H(L) = 0$, and $\hat{f}$ is a local minimum of h-fusion on a tree structure π, then*

$$E(\hat{f}) \;\leq\; 2cE(f^*) \tag{5.16}$$

where constant c is determined by V and π as in Definition 5.13.

Proof. Without loss of generality we assume that all weights are $w_{pq} = 1$. Consider any local minimum $\hat{f}^j$ computed by h-fusion at internal node $j \in \mathcal{S} \cup \{r\}$, and let us choose some child node $i \in \mathcal{I}(j)$. We first define a useful set of pixels with respect to a global optimum f^* as

$$\mathcal{P}_i \stackrel{\text{def}}{=} \{\, p : f^*_p \in \mathcal{L}_i \,\}.$$

This set contains all pixels assigned a label within subtree i, and so for any other child $i' \neq i$ we know that $\mathcal{P}_i \cap \mathcal{P}_{i'} = \emptyset$.

We can produce a labeling $\hat{f}^{j \otimes i}$ within one h-fusion move from local minimum $\hat{f}^j$ as follows:

$$\hat{f}^{j\otimes i}_p \stackrel{\text{def}}{=} \begin{cases} \hat{f}^i_p & \text{if } p \in \mathcal{P}_i \\ \hat{f}^j_p & \text{otherwise.} \end{cases}$$

Since each $\hat{f}^j$ is known to be a local optimum w.r.t. expansion moves for each $i \in \mathcal{I}(j)$ we know that

$$E(\hat{f}^j) \;\leq\; E(\hat{f}^{j\otimes i}). \tag{5.17}$$

The general strategy to use (5.17) for different i to build an inequality that is ultimately of the form $E(\hat{f}^j) \leq E(f^*) + \text{error}$. This will be achieved by breaking the energy terms in E into parts in such a way that a recursive inequality can be established. The recursive inequality will then be expanded until all terms can be bounded relative to $E(f^*)$.

Let $E(\cdot)|_{\mathcal{A}}$ denote a restriction of the summands of energy (5.2) to only the following terms:

$$E(f)|_{\mathcal{A}} \stackrel{\text{def}}{=} \sum_{p \in \mathcal{A}} D_p(f_p) \;+\; \sum_{pq \in \mathcal{A}} V(f_p, f_q).$$

We separate the unary and pairwise terms of $E(f)$ via interior, exterior, and boundary sets with respect to pixels $\mathcal{P}_i$:

$$\begin{aligned} \mathcal{A}_i &= \mathcal{P}_i && \cup \;\; \{\, pq \in \mathcal{N} \colon p, q \in \mathcal{P}_i \,\} \\ \overline{\mathcal{A}}_i &= \mathcal{P} \setminus \mathcal{P}_i && \cup \;\; \{\, pq \in \mathcal{N} \colon p, q \notin \mathcal{P}_i \,\} \\ \partial\mathcal{A}_i &= && \;\; \{\, pq \in \mathcal{N} \colon p \in \mathcal{P}_i, q \notin \mathcal{P}_i \,\}. \end{aligned}$$

The following facts now hold:

$$E(\hat{f}^{j\otimes i})|_{\mathcal{A}_i} = E(\hat{f}^i)|_{\mathcal{A}_i} \tag{5.18}$$

$$E(\hat{f}^{j\otimes i})|_{\overline{\mathcal{A}}_i} = E(\hat{f}^j)|_{\overline{\mathcal{A}}_i}. \tag{5.19}$$

Using (5.18) and (5.19) we can cancel out all the $\overline{\mathcal{A}}_i$ terms and rewrite (5.17) as

$$E(\hat{f}^j)|_{\mathcal{A}_i \cup \partial\mathcal{A}_i} \;\leq\; E(\hat{f}^i)|_{\mathcal{A}_i} + E(\hat{f}^{j\otimes i})|_{\partial\mathcal{A}_i} \tag{5.20}$$

For each $i \in \mathcal{I}(j)$ inequality (5.20) contains a subset of all the energy terms in $E(\hat{f}^j)|_{\mathcal{A}_j}$ pertaining to pixels $\mathcal{P}_i$. Let $\mathcal{I}^* = \{i \in \mathcal{I}(j) : \mathcal{P}_i \neq \emptyset\}$ be the set of children whose sub-trees contain a label used by f^*. If we sum inequality (5.20) over all $i \in \mathcal{I}^*$, the left-hand side will contain *all* the terms in $E(\hat{f}^j)|_{\mathcal{A}_j}$ (and more). Adding up all the left-hand sides we have

$$\sum_{i \in \mathcal{I}^*} E(\hat{f}^j)|_{\mathcal{A}_i \cup \partial\mathcal{A}_i} = E(\hat{f}^j)|_{\mathcal{A}_j \cup \partial\mathcal{A}_j} + \sum_{i \in \mathcal{I}^*} E(\hat{f}^j)|_{\partial\mathcal{A}_i \setminus \partial\mathcal{A}_j} \geq E(\hat{f}^j)|_{\mathcal{A}_j}. \tag{5.21}$$

Using (5.21) and likewise adding up the right-hand sides of (5.20) we have

$$E(\hat{f}^j)|_{\mathcal{A}_j} \leq \sum_{i\in\mathcal{I}^*} E(\hat{f}^i)|_{\mathcal{A}_i} + E(\hat{f}^{j\otimes i})|_{\partial\mathcal{A}_i} \tag{5.22}$$

$$= \sum_{i\in\mathcal{I}^*} E(\hat{f}^i)|_{\mathcal{A}_i} + E(\hat{f}^{j\otimes i})|_{\partial\mathcal{A}_i\cap\partial\mathcal{A}_j} + E(\hat{f}^{j\otimes i})|_{\partial\mathcal{A}_i\setminus\partial\mathcal{A}_j} \tag{5.23}$$

$$= \sum_{i\in\mathcal{I}^*} E(\hat{f}^i)|_{\mathcal{A}_i} + \sum_{pq\in\partial\mathcal{A}_i\cap\partial\mathcal{A}_j} V(\hat{f}^i_p, \hat{f}^j_q) \;+ \sum_{pq\in\partial\mathcal{A}_i\setminus\partial\mathcal{A}_j} V(\hat{f}^i_p, \hat{f}^j_q) \tag{5.24}$$

The first important observation about (5.24) is that each term $E(\hat{f}^i)|_{\mathcal{A}_j}$ on the right-hand side can be substituted by recursively applying the inequality itself. We can recursively substitute, branching further and further down the tree, until the path finally stops at a leaf $\ell \in \mathcal{L}$ giving us base case $E(\hat{f}^\ell)|_{\mathcal{A}_\ell} = \sum_{p\in\mathcal{P}_\ell} D_p(f^*_p)$. The sets $\{\mathcal{P}_\ell\}_{\ell\in\mathcal{L}}$ must be disjoint and their union is $\mathcal{P}_j$ so expression (5.24), when fully expanded, becomes

$$= \sum_{p\in\mathcal{A}_j} D_p(f^*_p) \;+ \text{ pairwise terms of the form } V(\hat{f}^i_p, \hat{f}^{\pi(i)}_q). \tag{5.25}$$

The second observation about (5.24) is that each edge pq on an outer boundary $\partial\mathcal{A}_i \cap \partial\mathcal{A}_j$ appears once in the sum over $\mathcal{I}^*$ whereas each edge on an interior boundary $\partial\mathcal{A}_i \setminus \partial\mathcal{A}_j$ appears twice: once for $p \in \mathcal{A}_i$ and once for some $q \in \mathcal{A}_{i'}$. Example 5.16, located after this proof, attempts to visualize this double-counting of pairwise terms. By careful accounting we collect all the $V(\hat{f}^i_p, \hat{f}^{\pi(i)}_q)$ terms generated by the recursive substitution and express (5.24) as[4]

$$= \sum_{p\in\mathcal{A}_j} D_p(f^*_p) + \sum_{pq\in\mathcal{A}_j}\left(\sum_{i\in\mathcal{J}(f^*_p;f^*_q)} V(\hat{f}^i_p, \hat{f}^{\pi(i)}_q) \;+ \sum_{i\in\mathcal{J}(f^*_q;f^*_p)} V(\hat{f}^{\pi(i)}_p, \hat{f}^i_q)\right) \tag{5.26}$$

where we define $\mathcal{J}(\ell;\ell')$ to be the set of nodes along the path from a label $\ell \in \mathcal{L}$ up to, but not including, the lowest common ancestor of ℓ and ℓ', namely

$$\mathcal{J}(\ell;\ell') = \big\{\ell, \pi(\ell), \ldots, \pi^{n-1}(\ell)\big\} \quad \text{where} \quad \pi^n(\ell) = \text{lca}(\ell, \ell').$$

All that remains is to bound each $V(\hat{f}^i_p, \hat{f}^{\pi(i)}_q)$ in terms of $V(f^*_p, f^*_q)$ using the factors a_i, b_i, d_i described in Definition 5.13. For a particular edge pq shown in (5.26) we must have each $V(\hat{f}^i_p, \hat{f}^{\pi(i)}_q) \leq a_{\pi(i)}$ and so their sum is

$$\sum_{i\in\mathcal{J}(f^*_p;f^*_q)} V(\hat{f}^i_p, \hat{f}^{\pi(i)}_q) \;\leq\; a_{\pi(f^*_p)} + \cdots + a_{\text{lca}(f^*_p,f^*_q)} \;\leq\; b_{\text{lca}(f^*_p,f^*_q)}. \tag{5.27}$$

We also know that $V(f^*_p, f^*_q) \geq d_{\text{lca}(f^*_p,f^*_q)}$ so we can use ratio $\frac{b_{\text{lca}(f^*_p,f^*_q)}}{d_{\text{lca}(f^*_p,f^*_q)}}$ to bound the approximation error at each edge pq appearing in (5.26), giving upper-bound

$$\leq \sum_{p\in\mathcal{A}_j} D_p(f^*_p) + \sum_{pq\in\mathcal{A}_j}\left(2\,\frac{b_{\text{lca}(f^*_p,f^*_q)}}{d_{\text{lca}(f^*_p,f^*_q)}} V(f^*_p, f^*_q)\right). \tag{5.28}$$

[4] Due to our assumption that $V(\ell,\ell) = 0$ we can simply sum over all $pq \in \mathcal{A}_j$ instead of only where $f^*_p \neq f^*_q$.

If j is the root of the tree, then $\{p \in \mathcal{A}_j\} = \mathcal{P}$ and $\{pq \in \mathcal{A}_j\} = \mathcal{N}$. Using the fact that any ratio $\frac{b_i}{d_i}$ is bounded from above by quantity c (Definition 5.13) we arrive at

$$\leq \sum_{p\in\mathcal{P}} D_p(f^*_p) \;+\; 2c \sum_{pq\in\mathcal{N}} V(f^*_p, f^*_q) \tag{5.29}$$

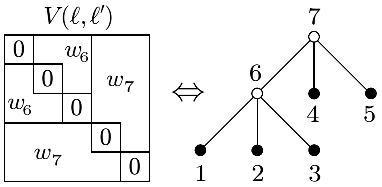
$$= E(f^*) \;+\; (2c-1) \sum_{pq\in\mathcal{N}} V(f^*_p, f^*_q) \tag{5.30}$$

$$\leq 2cE(f^*). \tag{5.31}$$

This completes the proof that $E(\hat{f}) \leq 2cE(f^*)$ where $\hat{f}$ is the labeling generated at the root of the tree. ■

Theorem 5.15 is important because it helps us to better understand the energy-dependent qualities that make h-fusion a strong algorithm. Again, the key coefficient is c and one can look at Figure 5.5 to see numerical values of c on a few examples. To better understand the idea of the proof itself, we provide the following example.

Example 5.16. *The following is meant to help understand how the proof of Theorem 5.15 works. An example h-Potts potential (V, π) is shown below, at left, for a 5-label energy. Suppose that we ran h-fusion on a small 5×5 pixel grid $\mathcal{P}$ with simple nearest-neighbour edges $\mathcal{N}$, and the result was local minimum $\hat{f}^7$ shown below (computed by fusing $\hat{f}^6$ and labels $4, 5$). If the global optimum is f^* shown at right, then clearly there is some approximation error.*



$\hat{f}^6$

3	3	3	3	3
3	3	3	3	3
3	3	3	3	3
3	3	3	3	3
3	3	3	3	3

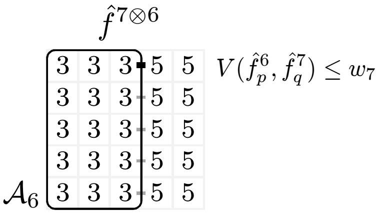
$\hat{f}^7$

5	5	5	5	5
5	5	5	5	5
5	5	5	5	5
5	5	5	5	5
5	5	5	5	5

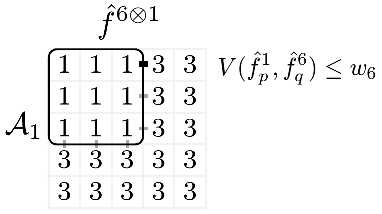
f^*

1	1	1	4	4
1	1	1	4	4
1	1	1	4	4
2	2	2	4	4
2	2	2	4	4

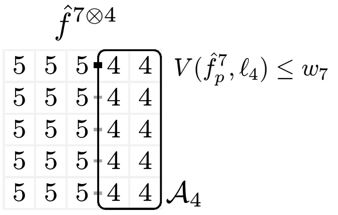
The key step is to understand how recursive application of (5.24) *can transform $\hat{f}^7$ into f^* while accumulating a bounded amount of error. Below (at left) we see each $pq \in \partial\mathcal{A}_6$ in the labeling $\hat{f}^{7\otimes6}$ can be bounded above by $V_7^{\max} = w_7$. Going deeper in the recursion (center) we see that each edge $pq \in \partial\mathcal{A}_1$ in $\hat{f}^{6\otimes1}$ can likewise be bounded by $V_6^{\max} = w_6$. Each such edge appears* again *when the other side is recursively descended, e.g. as part of $\partial\mathcal{A}_4$ (at right).*



3	3	3	5	5
3	3	3	5	5
3	3	3	5	5
3	3	3	5	5
3	3	3	5	5



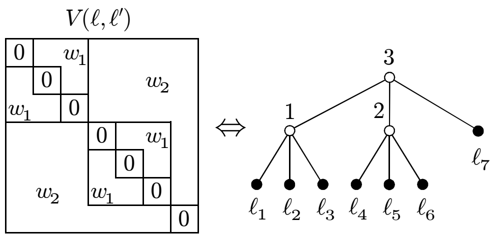
1	1	1	3	3
1	1	1	3	3
1	1	1	3	3
3	3	3	3	3
3	3	3	3	3



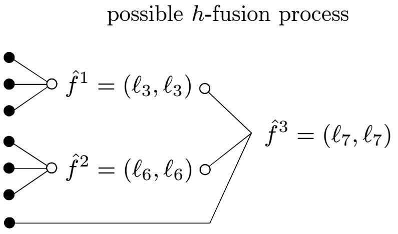
5	5	5	4	4
5	5	5	4	4
5	5	5	4	4
5	5	5	4	4
5	5	5	4	4

Now consider a worst-case example for h-fusion, where the local minima $\{\hat{f}^i\}$ are poor approximations all the way up the hierarchy. We will use this example to show that there are problem instances for which the optimality bound of Theorem 5.15 is tight.

Example 5.17. *Consider an energy with labels $\ell_1, \ldots, \ell_7$ and only two variables $\mathcal{P} = \{p, q\}$. We use h-Potts potential (V, π) shown below, at left. The h-fusion algorithm will first compute some labelings $\hat{f}^1, \hat{f}^2$ and then compute $\hat{f}^3$ as the solution. However, if the data costs $D_p(\cdot)$ and $D_q(\cdot)$ are of the form shown below, at right, then h-fusion can propagate approximation errors up the hierarchy. Specifically, $A \le w_1, B \le A + w_2$, and K is a large constant.*



data costs	p	q
ℓ_1	0	$2K$
ℓ_2	K	K
ℓ_3	A	$A{+}K$
ℓ_4	$2K$	0
ℓ_5	K	K
ℓ_6	$A{+}K$	A
ℓ_7	B	B


possible h-fusion process
$\hat{f}^1 = (\ell_3, \ell_3)$
$\hat{f}^2 = (\ell_6, \ell_6)$
$\hat{f}^3 = (\ell_7, \ell_7)$

To clarify, $\hat{f}^1 = (\ell_3, \ell_3)$ is a local minimum for α-expansion on the first three labels if $A \le w_1$ because neither $\hat{f}^1_p$ nor $\hat{f}^1_q$ wish to change to ℓ_1 or ℓ_2 respectively. Likewise from $\hat{f}^2$. The data costs are designed to incur the same problem when labelings $\hat{f}^1, \hat{f}^2$ and label ℓ_7 are fused to generate $\hat{f}^7$, i.e. that (ℓ_7, ℓ_7) is a local minimum if $B \le A + w_2$.

In Example 5.17 we can see that $E(\hat{f}^3) = 2B$ which we can assume to be $2(w_1 + w_2)$ in the worst case. It follows by inspection that $f^* = (\ell_1, \ell_4)$ is the optimal labeling and $E(f^*) = w_2$. We therefore have an approximation error of $2\frac{w_1+w_2}{w_2}$ which is equal to twice the constant c from Definition 5.13. The tree structure from Example 5.17 can be extended to arbitrary height k in a pattern that causes approximation error of $2\frac{w_1+w_2+\cdots w_k}{w_k} = 2c$ at the root labeling. This leads to an important observation about Theorem 5.15.

Observation 5.18. *Example 5.17 demonstrates a class of worst-case energies for which the approximation bound in Theorem 5.15 is tight.*

5.5 Hierarchical Fusion with Label Costs

By now we have characterized the class of hierarchical smooth costs (h-metrics) that can be optimized by our h-fusion algorithm. We now turn our attention to another important kind of energy term: *label costs*. Chapter 4 already demonstrated how label costs are useful in computer vision, especially in conjunction with smooth costs. In particular, the "category costs" mentioned in Section 4.8 are a kind of label *subset* cost that are useful for modeling a number of inference problems.

However, a major conclusion from that chapter (Theorem 4.1) was that the α-expansion algorithm has poor optimality guarantees for in the case of label *subset* costs. If the energy contains a cost $H(L)$ assigned to label subset $L \subseteq \mathcal{L}$, then the worst-case the approximation error can multiply by a factor of $|L|$. The worst-case example on page 50 shows precisely how local minima occur when $|L|$ is large. In fact our entire h-fusion algorithm was originally motivated by the need to handle label costs that were problematic for α-expansion. We will show that, when label subset costs are involved ($|L| > 1$), our h-fusion algorithm significantly improves upon the theoretical error bounds of α-expansion, *i.e.* upon Theorem 4.1.

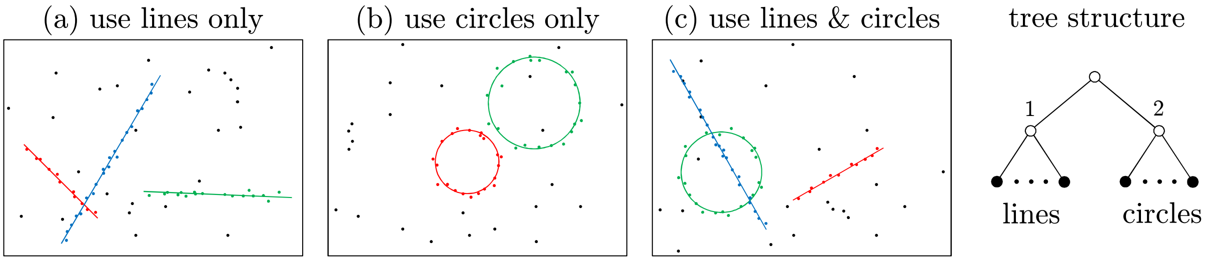


Figure 5.6: Result (a) is a straight-forward example of line-fitting via the PEARL procedure used in Chapter 4. The same procedure can be used for circle-fitting (b). When fitting a *mixture* of lines and circles (c) the random proposals naturally form two groups we can call 1 and 2. Adding costs $H(\mathcal{L}_1), H(\mathcal{L}_2) > 0$ causes the energy to prefer solutions that explain the data using only one of the two groups (unless the data strongly justifies both categories, *i.e. category costs* from Section 4.8).

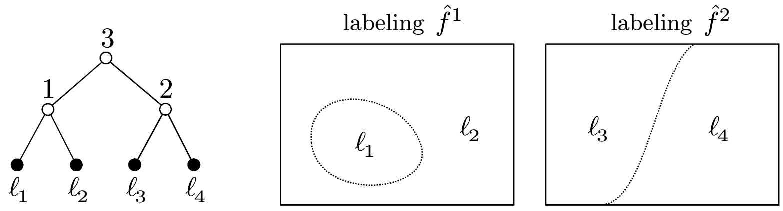
It is natural to ask, "in what sense are label subset costs 'hierarchical'?" The answer is that the subsets themselves may be *nested* in a way that corresponds to a hierarchy. In the simplest scenario, assume we have in mind a specific label hierarchy π with labels $\mathcal{L}$ and pseudo-labels $\mathcal{S}$. Associated with each tree node $i \in \mathcal{L} \cup \mathcal{S}$ is a one-time cost $H(\mathcal{L}_i)$ for using any label in its corresponding subtree[5]. Figure 5.6 shows a toy application of this sort, where we want basic per-label costs as well as group costs. In these simple scenarios, label costs are 'nested' in the sense that the sets $\{\mathcal{L}_i\}_{i \in \mathcal{I}(j)}$ are disjoint and their union forms $\mathcal{L}_j$. Such subsets form a hierarchy of label costs that precisely mirrors the tree structure of π. We will make the concept of hierarchical label costs more precise in Definitions 5.21 and 5.24.

We are now ready to generalize our h-fusion algorithm so that it can handle both h-metric smooth costs *and* hierarchical label costs. There are four main steps in this section:

1. understand how label costs affect the multi-label fusion energy,
2. determine what class of label costs can be optimized by h-fusion,
3. extend the pseudocode for SETUPFUSION from Section 5.3, and
4. generalize our approximation bound for h-fusion from Section 5.4.

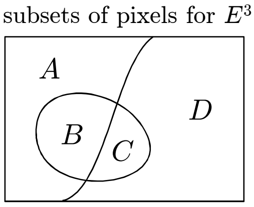
Ultimately we must extend the pseudocode for SETUPFUSION so that, when the fusion energy E' is configured, all relevant label costs in the original energy E are accounted for. The necessary changes and limitations can be understood from the following examples. (It may help to review *local label costs* $H_P(\cdot)$ from Section 4.3.4.)

Example 5.19. *Suppose we have the label hierarchy shown below, at left, along with two child labelings* $\hat{f}^1$ *and* $\hat{f}^2$*. We wish to compute the final labeling* $\hat{f}^3$.



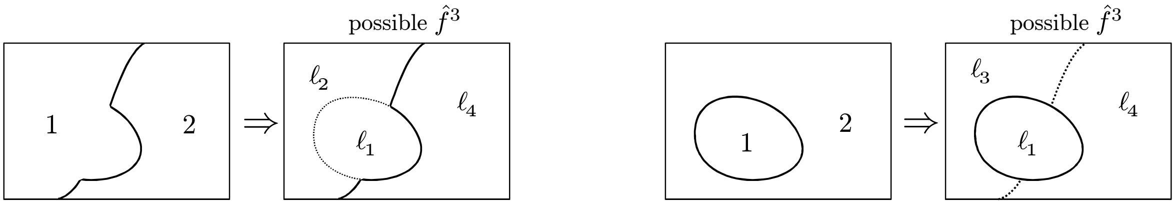
[5] Recall $\mathcal{L}_i = \{\ell \in \mathcal{L} : \ell \in \text{subtree}(i)\}$ is the set of labels (leaves) within the subtree of node i (Definition 5.4).

Assume label costs $H(L) > 0$ for label subsets $L \in \{\{\ell_1\}, \{\ell_2\}, \{\ell_3\}, \{\ell_4\}, \{\ell_1, \ell_2\}, \{\ell_3, \ell_4\}\}$. When computing $\hat{f}^3$, the multi-label fusion energy E^3 must correctly account for all $H(L)$ in the original energy. This is accomplished by using the following label cost terms to E^3, four of which are local to specific pixel subsets:



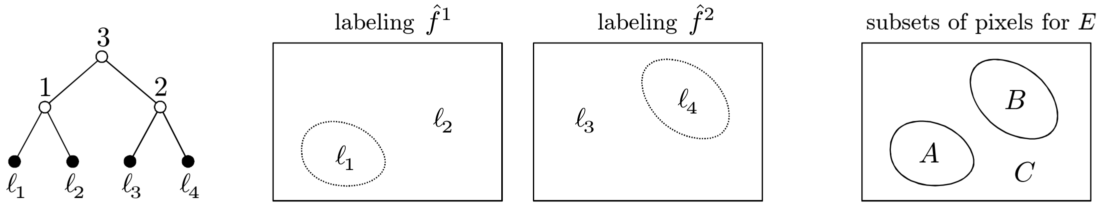
$$
\begin{aligned}
H'(1) &:= H(\{\ell_1, \ell_2\}) \qquad & H'_{B \cup C}(1) &:= H(\ell_1) \\
H'(2) &:= H(\{\ell_3, \ell_4\}) \qquad & H'_{A \cup D}(1) &:= H(\ell_2) \\
& & H'_{A \cup B}(2) &:= H(\ell_3) \\
& & H'_{C \cup D}(2) &:= H(\ell_4)
\end{aligned}
$$

The local label costs $H'_P(I)$ above are needed to ensure that the true label costs are correctly encoded. For example, the fusion labeling below at left should pay all label costs except $H(\ell_3)$, whereas the fusion labeling at right should pay all costs except $H(\ell_2)$.



The key observation of Example 5.19 is that many of the *global* label costs in the original energy become *local* label costs in the corresponding fusion energy. This occurs because, for any labeling $\hat{f}^i$ being fused, a particular label $\ell \in \mathcal{L}_i$ need only occupy a small portion of $\hat{f}^i$. The label cost $H(\ell)$ should *only* be paid if index i is chosen *for at least one of those specific pixels*—the remaining pixels cannot introduce label ℓ. The same is generally true for a label subset cost $H(L)$ for some $L \subseteq \mathcal{L}$, except the situation becomes more complex and there are potential problems. As the next example will show, it is possible that for some L the label cost $H(L)$ of the original energy *cannot* be converted to local label costs in the fusion energy. This has implications for the kinds of energies we can optimize with h-fusion.

Example 5.20. *Suppose we have the same label hierarchy as Example 5.19, but this time we have the child labelings $\hat{f}^1$ and $\hat{f}^2$ shown below.*



Assume the original energy contains a label subset cost $H(\{\ell_1, \ell_4\}) > 0$. This cost should be paid in fusion energy E^3 if any pixel in A is assigned pseudo-label 1 or *any pixel in B is assigned pseudo-label 2. Because each region is 'activated' by a different label in the fusion energy, this kind of potential is* not *the kind of label cost we have dealt with thus far.*

Example 5.20 begs the question: even though $H(\{\ell_1, \ell_4\})$ cannot be encoded by a label cost in the fusion energy, can it be encoded *anyway* and still be optimized by α-expansion? The answer is no, because the resulting energy potential is not submodular (Section 2.1.3) with respect to expansion moves. This can be seen in Example 5.20 by the fact that, if we let $E(i, i')$ denote the label cost of assigning index i to pixels A and index i' to pixels B, then

$$E(1,1) + E(2,2) = 2H(\{\ell_1, \ell_4\}) > H(\{\ell_1, \ell_4\}) = E(1,2) + E(2,1). \tag{5.32}$$

The above inequality means that a label cost $H(\{\ell_1, \ell_4\}) > 0$ in the original energy can cause a supermodular potential in the fusion energy, and thus expansion moves are not applicable. In fact because this example has only two labels, we can conclude that the $\alpha\beta$-swap algorithm is also inapplicable inside the h-fusion step. Therefore, we must be careful that our h-fusion algorithm is used only when situations resembling Example 5.20 are impossible. That is why we introduced the concept of *hierarchical label costs* (Definition 5.24), and the simpler special case of *tree-structured label costs* (Definition 5.21) below.

Definition 5.21. *Given a tree structure π we say that (H, π) forms "tree-structured label costs" if $H(L) > 0 \;\Rightarrow\; L \in \{\mathcal{L}_i\}_{i \in \mathcal{L} \cup \mathcal{S}}$.*

An energy $E(f)$ has *tree-structured label costs* if, with respect to some tree, the energy can be written as

$$E(f) = \sum_{p \in \mathcal{P}} D_p(f_p) \; + \sum_{pq \in \mathcal{N}} w_{pq} \cdot V(f_p, f_q) \; + \sum_{i \in \mathcal{L} \cup \mathcal{S}} H(\mathcal{L}_i) \cdot \delta_{\mathcal{L}_i}(f). \tag{5.33}$$

To motivate the above definition, we immediately state an approximation result concerning such energies.

Theorem 5.22. *Suppose f^* is a global minimum of an energy with h-metric (V, π) and tree-structured label costs (H, π). If $\hat{f}$ is a local minimum of h-fusion, then*

$$E(\hat{f}) \;\le\; 2cE(f^*) \;+ \sum_{i \in \mathcal{L} \cup \mathcal{S}} H(\mathcal{L}_i) \tag{5.34}$$

where constant c is determined by V and π (Definition 5.13).

The multiplicative coefficient of $2c$ in Theorem 5.22 is a substantial improvement over the analogous coefficient $(2c + c')$ for standard α-expansion (Theorem 4.1 on page 47). Recall that coefficient $c' = \max_{H(L)>0} |L| - 1$ and so, for an energy with tree-structured label costs, $c' = \max_{i \in \mathcal{I}(r)} |\mathcal{L}_i| - 1$ where r is the root of the tree. This coefficient can be very large and so α-expansion's optimality guarantees are poor in this case. Our h-fusion algorithm eliminates the factor of c' entirely and gives the best known approximation bounds for these kinds of hierarchical energies.

Instead of proving bound (5.34) directly, we will show that it follows from a more general theorem. Our h-fusion algorithm can actually handle a wider class of label costs that are still 'hierarchical' but not necessarily tree-structured.

Definition 5.23. *Given a tree structure π, its "hierarchical label subsets" is the set*

$$\mathcal{H} = \big\{ L : \; (L \cap \mathcal{L}_i = \emptyset \;\vee\; L \subset \mathcal{L}_i \;\vee\; L \supseteq \mathcal{L}_i) \;\forall\, i \in \mathcal{S} \cup \{r\} \big\} \tag{5.35}$$

An equivalent definition is to say that a subset $L \subseteq \mathcal{L}$ belongs to $\mathcal{H}$ if and only if $L = \cup_{i \in I} \mathcal{L}_i$ for some (possibly empty) set of sibling nodes I in the tree. For example, let us assume $\mathcal{L} = \{\ell_1, \ldots, \ell_6\}$ and the label hierarchy contains three simple groups $S = \{7, 8, 9\}$ with $\mathcal{L}_7 = \{\ell_1, \ell_2\}, \mathcal{L}_8 = \{\ell_3, \ell_4\}, \mathcal{L}_9 = \{\ell_5, \ell_6\}$. The hierarchical subsets are

$$\begin{aligned} \mathcal{H} = \big\{ \{\}, &\{\ell_1\}, \{\ell_2\}, \{\ell_3\}, \{\ell_4\}, \{\ell_5\}, \{\ell_6\}, \\ &\{\ell_1, \ell_2\}, \{\ell_3, \ell_4\}, \{\ell_5, \ell_6\}, \\ &\{\ell_1, \ell_2, \ell_3, \ell_4\}, \{\ell_1, \ell_2, \ell_5, \ell_6\}, \{\ell_3, \ell_4, \ell_5, \ell_6\} \\ &\{\ell_1, \ell_2, \ell_3, \ell_4, \ell_5, \ell_6\} \big\}. \end{aligned} \tag{5.36}$$

The sets of cardinality four in (5.36) are not tree-structured label costs, but they are hierarchical in a sense that we can optimize. Note that sets like $\{\ell_2, \ell_3\}$ and $\{\ell_1, \ell_2, \ell_3\}$ are not in $\mathcal{H}$ because they cannot be generated from a union of siblings in the particular hierarchy chosen.

Definition 5.24. *Given a tree structure π we say that (H, π) form "hierarchical label costs" if $H(L) > 0 \;\Rightarrow\; L \in \mathcal{H}$.*

As we shall see, if an energy $E(f)$ has *hierarchical label costs* with respect to some tree, then $E(f)$ can be minimized by h-fusion on that tree. On the other hand, if the label costs are not hierarchical (*e.g.* arbitrary subsets $L \subseteq \mathcal{L}$) then there are cases where h-fusion can only be applied with a 'flat' tree and is therefore equivalent to standard α-expansion.

Observation 5.25. *Tree-structured label costs are a sub-class of hierarchical label costs. This follows from the fact that $\{\mathcal{L}_i\}_{i \in \mathcal{L} \cup \mathcal{S}} \subseteq \mathcal{H}$ for any tree structure, as can be seen by collecting only those elements of* (5.35) *for which* $|I| = 1$.

We are finally ready to generalize the pseudocode for SETUPFUSION (Section 5.3) to handle label costs. Keep in mind that the particular steps on lines **4–6** are intended to be a correct encoding only for hierarchical label costs, and would not always be correct for arbitrary $L \subseteq \mathcal{L}$.

```
SETUPFUSION($E, \pi, j$) with label costs
1  $D'_p(i) := D_p(\hat{f}_p^i)$                        $\forall\, p \in \mathcal{P}, i \in \mathcal{I}(j)$
2  $V'_{pq}(i, i') := w_{pq} \cdot V(\hat{f}_p^i, \hat{f}_q^{i'})$   $\forall\, pq \in \mathcal{N},\; i, i' \in \mathcal{I}(j)$
3  for each $L \in \mathcal{H}$ such that $H(L) > 0$
4      $I := \{\, i \in \mathcal{I}(j) \,:\, L \cap \mathcal{L}_i \neq \emptyset \,\}$        // indices of labelings containing $L$
5      $P := \{\, p \in \mathcal{P} \,:\, \exists \hat{f}_p^i \in L, i \in I \,\}$        // pixels where some labeling is in $L$
6      $H'_P(I) := H(L)$                            // convert $H(L)$ into regional label cost for $E'$
7  end
8  return $E' = (D', V', H')$
```

Note that on line **3** that for any $L \supseteq \mathcal{L}_j$ the corresponding I is the full set of indices $\mathcal{I}(j)$, so these label costs are always paid for the subproblem at node j; we can optionally omit these label costs, and the fusion energy at node j will still be correct up to an additive constant.

Theorem 5.26. *The h-fusion algorithm is correct with hierarchical label costs.*

Proof. We must prove that the multi-label fusion energy E' created by SETUPFUSION is consistent with the original energy E. Let $g : \mathcal{P} \to \mathcal{I}(j)$ be the labeling of E' that indexes the local minima $\{\hat{f}^i\}$, and let $f(g)$ be the corresponding labeling of E defined by $f_p(g_p) = \hat{f}_p^{g_p}$. We must confirm that $E'(g) = E(f(g))$ for all possible configurations of g.

The correctness of D' and V' are self-evident. Given a particular label subset L with $H(L) > 0$, the correctness of H' reduces to showing that $\delta_L(f(g)) = \delta_I(g_P)$ where indices I and pixels P are as defined on lines **4** and **5**. In other words, we must have

$$\exists \hat{f}_p^{g_p} \in L,\ p \in \mathcal{P} \quad \text{if and only if} \quad \exists g_p \in I,\ p \in P \tag{5.37}$$

where we know that $\hat{f}_p^{g_p} \in \mathcal{L}_{g_p}$. Because we assume hierarchical label costs (Definition 5.23), each L belongs to one of four cases.

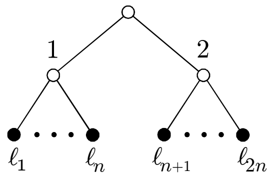
1. If $L \cap \mathcal{L}_i = \emptyset$ for all $i \in \mathcal{I}(j)$, then we know that any $\hat{f}_p^{g_p} \notin L$. Therefore setting $I = \emptyset$ ensures $\delta_L(f(g)) = \delta_\emptyset(g_P) = 0$ which is correct.
2. If $L \subset \mathcal{L}_i$ for some $i \in \mathcal{I}(j)$, then $I = \{i\}$ and $P = \{p : \hat{f}_p^i \in L\}$. Clearly if $g_p \neq i$ then $\hat{f}_p^{g_p} \notin L$ and so $f(g)$ contains a label in L if and only if $g_p = i$ for some $p \in P$. Therefore $\delta_L(f(g)) = \delta_i(g_P)$ holds in this case.
3. If $\mathcal{L}_i \subseteq L \subset \mathcal{L}_j$ for some $i \in \mathcal{I}(j)$, then clearly $P = \mathcal{P}$. By the hierarchical label costs assumption we must also have $L = \cup_{i \in I} \mathcal{L}_i$, and so $\delta_L(f(g)) = \delta_I(g)$ holds in this case.
4. If $L \supseteq \mathcal{L}_j$ then $H(L)$ can be added to the energy as a constant or simply ignored.

■

Looking at the proof of Theorem 5.26 we can see that the special structure of hierarchical label costs is especially needed for the third case. If it were possible to have a subset $L \supset \mathcal{L}_i$ that could *not* be written as a union of i's siblings, then problematic cases like Example 5.20 would be possible. The resulting fusion energy could not be minimized by α-expansion because the internal binary steps would be non-submodular and potentially NP-hard.

To understand the kind of situation where h-fusion is clearly a good idea, consider the problem instance in the example below. It is modeled after the simplest application of label subset costs, such as the scenario in Figure 5.6.

Example 5.27. *Suppose we have variables $\mathcal{P} = \{p_1, \ldots, p_n\}$ and labels $\mathcal{L} = \{\ell_1, \ldots, \ell_{2n}\}$. The labels are grouped into $\mathcal{L}_1$ and $\mathcal{L}_2$ as shown at left, and the data costs $D_p(\cdot)$ are shown in a table where $a > 0$. We assume label costs $H(\mathcal{L}_1) = H(\mathcal{L}_2) = h$ for some constant $h > 0$, and assume no smooth costs at all.*



data costs

	p_1	$\cdots$	p_n	label costs
ℓ_1	a	$\cdots$	a	h
$\vdots$	$\vdots$	$\ddots$	$\vdots$	
ℓ_n	a	$\cdots$	a	
ℓ_{n+1}	0	$\cdots$	∞	h
$\vdots$	$\vdots$	$\ddots$	$\vdots$	
ℓ_{2n}	∞	$\cdots$	0	

$\hat{f} = (\ell_1, \ldots, \ell_1)$ for α-expansion

$f^* = (\ell_{n+1}, \ldots, \ell_{2n})$

So long as $a \leq h$ the α-expansion algorithm will consider $\hat{f}$ shown above to be a local minimum. Since $E(\hat{f}) = na \leq nh$ and $E(f^*) = h$, the approximation error for α-expansion is linear in n. The h-fusion algorithm will compute intermediate labelings $\hat{f}^1 = \hat{f}$ and $\hat{f}^2 = f^*$, then fuse them to return $\hat{f}^3 = f^*$ as the (globally optimal) final result.

We now focus our attention to the issue of theoretical approximation bounds. Example 5.27 showed a simple case with tree-structured label costs where h-fusion is a clear improvement over α-expansion. It turns out that the cardinality of I on line **4** of SETUPFUSION is an important quantity in the approximation bound for h-fusion. In general, the smaller the quantity $|I|$ when $I \subset \mathcal{I}(j)$, the better the approximation guarantees. We define a more general version of constant c' to incorporate this quantity, so that we may use it to express the bounds of h-fusion in our main Theorem 5.30.

Definition 5.28. *We let c' denote the maximum cardinality of any index set I (*SETUPFUSION*, line **4**) that is a strict subset of $\mathcal{I}(j)$, minus 1. That is, the constant*

$$c' = \max_{H(L)>0} |I(L)| - 1 \;\; \textit{where} \;\; I(L) = I \subset \mathcal{I}(j) \;\; \textit{for some } j \textit{ such that } \bigcup_{i \in I} \mathcal{L}_i = L$$

For example, suppose we again have $\mathcal{L} = \{\ell_1, \ldots, \ell_6\}$ and label groups $\mathcal{L}_7 = \{\ell_1, \ell_2\}, \mathcal{L}_8 = \{\ell_3, \ell_4\}, \mathcal{L}_9 = \{\ell_5, \ell_6\}$ used to generate (5.36). If our only label cost $H(L) > 0$ were on subset $L = \{\ell_5, \ell_6\}$ then because $I(L) = \{9\}$ we are left with coefficient $c' = 0$. If instead we have a label cost on any subset $L \in \{\{\ell_1, \ell_2, \ell_3, \ell_4\}, \{\ell_1, \ell_2, \ell_5, \ell_6\}, \{\ell_3, \ell_4, \ell_5, \ell_6\}\}$ we have in each case $I(L) \in \{\{7, 8\}, \{7, 9\}, \{8, 9\}\}$ yielding coefficient $c' = 1$. As we shall see, the approximation bounds of h-fusion are much stronger for energies where c' is small.

Observation 5.29. *For a 'flat' tree structure π, the coefficient c' reduces to $\max_{H(L)>0} |L| - 1$ which is exactly the coefficient used in the approximation bound of standard α-expansion.*

We can now state a generalized approximation bound for h-fusion when

Theorem 5.30. *Suppose f^* is a global minimum of an energy with h-metric (V, π) and hierarchical label costs (H, π). If $\hat{f}$ is a local minimum of h-fusion, then*

$$E(\hat{f}) \;\leq\; (2c + c')E(f^*) \;+\; \sum_{L \in \mathcal{H}} H(L) \tag{5.38}$$

where c (Definition 5.13) and c' (Definition 5.28) are constants.

Proof. We generalize the proof of Theorem 5.15. Again, without loss of generality assume uniform weights $w_{pq} = 1$. With respect to nodes i, j where $i \in \mathcal{I}(j)$, we repeat the following definitions verbatim. First, pixel set $\mathcal{P}_i$ based on optimal labeling f^*

$$\mathcal{P}_i \overset{\text{def}}{=} \{\, p : f^*_p \in \mathcal{L}_i \,\}.$$

Then, a fusion move $\hat{f}^{j \otimes i}$ based on two local minima $\hat{f}^j$ and $\hat{f}^i$

$$\hat{f}^{j \otimes i}_p \overset{\text{def}}{=} \begin{cases} \hat{f}^i_p & \text{if } p \in \mathcal{P}_i \\ \hat{f}^j_p & \text{otherwise.} \end{cases}$$

The restriction of energy (5.1) to a subset of energy terms $\mathcal{A}$

$$E(f)|_{\mathcal{A}} \overset{\text{def}}{=} \sum_{p \in \mathcal{A}} D_p(f_p) \; + \sum_{pq \in \mathcal{A}} V(f_p, f_q).$$

Finally, a partition of all energy terms into the following sets

$$\begin{array}{rlcl} \mathcal{A}_i = & \mathcal{P}_i & \cup & \{\, pq \in \mathcal{N} \colon p, q \in \mathcal{P}_i \,\} \\ \overline{\mathcal{A}}_i = & \mathcal{P} \setminus \mathcal{P}_i & \cup & \{\, pq \in \mathcal{N} \colon p, q \notin \mathcal{P}_i \,\} \\ \partial\mathcal{A}_i = & & & \{\, pq \in \mathcal{N} \colon p \in \mathcal{P}_i, q \notin \mathcal{P}_i \,\}. \end{array}$$

Recall that the main strategy of the proof is to start from local minimum $\hat{f}^j$ and recursively transform it into f^* via a set of expansion moves (fusion moves). Each move contributes an 'error' of sorts, but we can bound the total error by careful use of the following fact: each labeling $\hat{f}^j$ is a local minimum w.r.t. expansion moves, and so for all $i \in \mathcal{I}(j)$ we must have

$$E(\hat{f}^j) \;\le\; E(\hat{f}^{j \otimes i}). \tag{5.39}$$

The new element in this proof is the possibility of label costs in the energy E. Let $E_H(f)$ denote the total label cost incurred by a labeling f, *i.e.* the sum of label cost terms. We can bound the label cost $E_H(\hat{f}^{j\otimes i})$ of our fused labeling by

$$E_H(\hat{f}^{j\otimes i}) \le E_H(\hat{f}^j) \; + \sum_{\substack{L \subseteq \mathcal{L} \setminus \hat{\mathcal{L}}_j \\ L \cap \hat{\mathcal{L}}_i \neq \emptyset}} H(L) \tag{5.40}$$

where $\hat{\mathcal{L}}_j$ and $\hat{\mathcal{L}}_i$ are the sets of unique labels appearing in $\hat{f}^j$ and $\hat{f}^i$ respectively.

Looking at the key inequality (5.39), recall from Theorem 5.15 that we can break the energy terms on each side into parts based on sets $\mathcal{A}_i, \overline{\mathcal{A}}_i$, and $\partial\mathcal{A}_i$. Because $E(\hat{f}^{j\otimes i})|_{\overline{\mathcal{A}}_i} = E(\hat{f}^j)|_{\overline{\mathcal{A}}_i}$ these terms cancel out, and we can substitute $E(\hat{f}^{j\otimes i})|_{\mathcal{A}_i} = E(\hat{f}^i)|_{\mathcal{A}_i}$. Along with bound (5.40) and canceling the $E_H(\hat{f}^j)$ terms we can now rewrite (5.39) as

$$E(\hat{f}^j)|_{\mathcal{A}_i \cup \partial\mathcal{A}_i} \;\le\; E(\hat{f}^i)|_{\mathcal{A}_i} + E(\hat{f}^{j\otimes i})|_{\partial\mathcal{A}_i} + \sum_{\substack{L \subseteq \mathcal{L} \setminus \hat{\mathcal{L}}_j \\ L \cap \hat{\mathcal{L}}_i \neq \emptyset}} H(L) \tag{5.41}$$

Again, let $\mathcal{I}^* = \{i \in \mathcal{I}(j) : \mathcal{P}_i \neq \emptyset\}$ be the set of child nodes that contain a label used by f^* in their subtree. We sum inequality (5.41) over all $i \in \mathcal{I}^*$ to arrive at a recursive expression, this time incorporating errors incurred by label costs. The key observation is that a particular label cost $H(L)$ appears once on the right-hand side for each element in the set $\mathcal{I}^*_L = \{i \in \mathcal{I}^* : L \cap \hat{\mathcal{L}}_i \neq \emptyset\}$. The sum of inequalities (5.41) thus implies

$$E(\hat{f}^j)|_{\mathcal{A}_j} \;\le\; \left(\sum_{i \in \mathcal{I}^*} E(\hat{f}^i)|_{\mathcal{A}_i} + E(\hat{f}^{j\otimes i})|_{\partial\mathcal{A}_i} \right) + \sum_{L \subseteq \mathcal{L} \setminus \hat{\mathcal{L}}_j} H(L) \cdot |\mathcal{I}^*_L| \tag{5.42}$$

where the quantity in parentheses is identical to that of Theorem 5.15. The above inequality can be recursively expanded for each $E(\hat{f}^i)|_{\mathcal{A}_i}$ until the recursion stops at a label used by f^*. We already know that, after recursive substitution, the quantity in parentheses is bounded by

$$\sum_{p\in\mathcal{P}} D_p(f^*_p) \;+\; 2c \sum_{pq\in\mathcal{N}} V(f^*_p, f^*_q)$$

However, we must now argue for a bound on the total label cost accumulated by recursive application of (5.42). The central question is whether a particular subset L that appears in (5.42) with $|\mathcal{I}^*_L| > 0$ for node j can appear *again* when we recursively substitute the children $i \in \mathcal{I}^*$. If the answer were 'yes' then each label cost $H(L)$ could appear more than $|\mathcal{I}^*_L|$ total times by the end of recursive expansion, leading to a worse bound. Fortunately, Lemma 5.31 proves that this is not the case; each L appearing in the sum for j and child i (5.41) can never reappear in the sums for i or its children.

If we let $\mathcal{H}^*$ denote the set of all subsets L generated by recursive substitution of (5.42), we can thereby write

$$E(\hat{f}^j)|_{\mathcal{A}_j} \;\le\; \sum_{p\in\mathcal{P}} D_p(f^*_p) \;+\; 2c \sum_{pq\in\mathcal{N}} V(f^*_p, f^*_q) + \sum_{L\in\mathcal{H}^*} H(L)\cdot|\mathcal{I}^*_L| \tag{5.43}$$

From now on we assume j is the root of the tree structure, and so $\hat{f}^j = \hat{f}$, *i.e.* the final labeling output by h-fusion. Note that the left-hand side of (5.43) is still $E(\hat{f}^j)|_{\mathcal{A}_j}$ which does *not* include the label costs incurred by $\hat{f}^j$. By adding $E_H(\hat{f}^j)$ to both sides we have $E(\hat{f}^j)|_{\mathcal{A}_j} + E_H(\hat{f}^j) = E(\hat{f})$ on the left side, giving a new inequality below.

$$E(\hat{f}) \;\le\; \sum_{p\in\mathcal{P}} D_p(f^*_p) \;+\; 2c \sum_{pq\in\mathcal{N}} V(f^*_p, f^*_q) + E_H(\hat{f}) + \sum_{L\in\mathcal{H}^*} H(L)\cdot|\mathcal{I}^*_L| \tag{5.44}$$

$$= \; E(f^*) + (2c-1)\sum_{pq\in\mathcal{N}} V(f^*_p, f^*_q) + E_H(\hat{f}) - E_H(f^*) + \sum_{L\in\mathcal{H}^*} H(L)\cdot|\mathcal{I}^*_L| \tag{5.45}$$

All that is left is to re-group the summands in the last three terms (the label cost terms) in a way that proves our theorem. First we rewrite the three sums more explicitly, using $\hat{\mathcal{L}}$ and $\mathcal{L}^*$ to denote the unique labels used by $\hat{f} = \hat{f}^j$ and f^* respectively.

$$\sum_{\substack{L\in\mathcal{H}\\ L\cap\hat{\mathcal{L}}\neq\emptyset}} H(L) \;-\; \sum_{\substack{L\in\mathcal{H}\\ L\cap\mathcal{L}^*\neq\emptyset}} H(L) \;+ \sum_{L\in\mathcal{H}^*} H(L)\cdot|\mathcal{I}^*_L|$$

$$= \; \sum_{\substack{L\in\mathcal{H}\\ L\cap\mathcal{L}^*=\emptyset\\ L\cap\hat{\mathcal{L}}\neq\emptyset}} H(L) \;-\; \sum_{\substack{L\in\mathcal{H}\\ L\cap\mathcal{L}^*\neq\emptyset\\ L\cap\hat{\mathcal{L}}=\emptyset}} H(L) \;+ \sum_{L\in\mathcal{H}^*} H(L)\cdot|\mathcal{I}^*_L| \tag{5.46}$$

First note that if $|\mathcal{I}^*_L| > 1$ then this means $L \supset \mathcal{L}_i$ for some $\mathcal{L}_i \cap \mathcal{L}^* \neq \emptyset$ and so $L \cap \mathcal{L}^* \neq \emptyset$ also. We can break the last sum in (5.46) into two parts based on whether $L \cap \mathcal{L}^* \neq \emptyset$.

$$= \; \sum_{\substack{L\in\mathcal{H}\\ L\cap\mathcal{L}^*=\emptyset\\ L\cap\hat{\mathcal{L}}\neq\emptyset}} H(L) \;+\; \sum_{\substack{L\in\mathcal{H}^*\\ L\cap\mathcal{L}^*=\emptyset}} H(L) \;-\; \sum_{\substack{L\in\mathcal{H}\\ L\cap\mathcal{L}^*\neq\emptyset\\ L\cap\hat{\mathcal{L}}=\emptyset}} H(L) \;+\; \sum_{\substack{L\in\mathcal{H}^*\\ L\cap\mathcal{L}^*\neq\emptyset}} H(L)\cdot|\mathcal{I}^*_L| \tag{5.47}$$

We can also show that $L \in \mathcal{H}^* \;\Rightarrow\; L \cap \hat{\mathcal{L}} = \emptyset$ as follows. If $L \in \mathcal{H}^*$ then there must be some node i such that $L \cap \hat{\mathcal{L}}_i = \emptyset$ and $L \subset \mathcal{L}_i$. We know from (5.60) in Lemma 5.31 that $\hat{\mathcal{L}} \cap \mathcal{L}_i \subseteq \hat{\mathcal{L}}_i$, so this implies $\emptyset = L \cap \hat{\mathcal{L}}_i \supseteq L \cap (\hat{\mathcal{L}} \cap \mathcal{L}_i) = L \cap \hat{\mathcal{L}}$. This means the two leftmost sums of (5.47) have disjoint L and can be bounded by simply $\sum_{L \in \mathcal{H}} H(L)$. It furthermore implies that, for every L appearing in the rightmost sum of (5.47), the same L must appear in the negative sum. Putting these together we have upper bound on label costs

$$\leq \sum_{L \in \mathcal{H}} H(L) \;+ \sum_{\substack{L \in \mathcal{H}^* \\ L \cap \mathcal{L}^* \neq \emptyset}} H(L) \cdot (|\mathcal{I}_L^*| - 1) \tag{5.48}$$

$$\leq \sum_{L \in \mathcal{H}} H(L) \;+ c' \cdot \sum_{\substack{L \in \mathcal{H}^* \\ L \cap \mathcal{L}^* \neq \emptyset}} H(L) \tag{5.49}$$

$$\leq \sum_{L \in \mathcal{H}} H(L) \;+ c' E_H(f^*) \tag{5.50}$$

We can therefore revise bound (5.45) to

$$E(\hat{f}) \;\leq\; E(f^*) + (2c-1) \sum_{pq \in \mathcal{N}} V(f_p^*, f_q^*) + c' E_H(f^*) \;+ \sum_{L \in \mathcal{H}} H(L) \tag{5.51}$$

$$\leq\; E(f^*) + (2c-1) E(f^*) + c' E(f^*) + \sum_{L \in \mathcal{H}} H(L) \tag{5.52}$$

$$\leq\; (2c + c') E(f^*) + \sum_{L \in \mathcal{H}} H(L) \tag{5.53}$$

Thus completing the proof. ■

Lemma 5.31. *If label subset* L *appears in the summand of* (5.41) *for node* j *and child* i*, then* L *does not appear in the summands of* (5.41) *for any* $k \in \mathrm{subtree}(i)$.

Proof. To be clear, let us restate the claim more formally. Let $\mathcal{H}^{j \otimes i}$ denote all subsets L appearing in the label cost summands of (5.41) when applied to node j and child i, *i.e.*

$$\mathcal{H}^{j \otimes i} \overset{\text{def}}{=} \{\, L \,:\, L \cap \hat{\mathcal{L}}_j = \emptyset,\, L \cap \hat{\mathcal{L}}_i \neq \emptyset \,\} \tag{5.54}$$

We must prove that $L \in \mathcal{H}^{j \otimes i} \;\Rightarrow\; L \notin \mathcal{H}^{k \otimes l}$ for any $k \in \mathrm{subtree}(i)$ and $l \in \mathcal{I}(k)$.

First note that for each $L \in \mathcal{H}^{j \otimes i}$ we have

$$L \cap \hat{\mathcal{L}}_j = \emptyset \;\Rightarrow\; L \not\supseteq \mathcal{L}_j \tag{5.55}$$

$$L \cap \hat{\mathcal{L}}_i \neq \emptyset \;\Rightarrow\; L \cap \mathcal{L}_i \neq \emptyset \tag{5.56}$$

By the hierarchical label cost assumption (Definition 5.23) we can use (5.55) and (5.56) to conclude that $L \in \mathcal{H}^{j \otimes i} \;\Rightarrow\; L \subset \mathcal{L}_j$.

Now consider the set $\mathcal{H}^{j \otimes i} \cap \mathcal{H}^{k \otimes l}$. By the definition (5.54) an element L of this joint set must satisfy at least the following conditions:

$$L \cap \hat{\mathcal{L}}_i \neq \emptyset \tag{5.57}$$

$$L \cap \hat{\mathcal{L}}_k = \emptyset \tag{5.58}$$

$$L \subset \mathcal{L}_k. \tag{5.59}$$

However, no subset L can satisfy all three conditions, as we now show. In the h-fusion algorithm, if $\hat{f}^i$ contains a label $\ell \in \mathcal{L}_k$, then $\hat{f}^k$ *must* contain ℓ as well—after all, there is no other way that a label in $\mathcal{L}_k$ could have propagated up to $\hat{f}^i$. This relation can be restated as

$$\hat{\mathcal{L}}_i \cap \mathcal{L}_k \subseteq \hat{\mathcal{L}}_k \quad \forall\, k \in \text{subtree}(i) \tag{5.60}$$

Starting from (5.57) we can say

$$\begin{aligned} & L \cap \hat{\mathcal{L}}_i \neq \emptyset & \\ \Rightarrow\ & L \cap (\hat{\mathcal{L}}_i \cap \mathcal{L}_k) \neq \emptyset & \text{by (5.59)} \\ \Rightarrow\ & L \cap (\hat{\mathcal{L}}_k) \neq \emptyset & \text{by (5.60)} \end{aligned}$$

which contradicts requirement (5.58). Thus $\mathcal{H}^{j\otimes i} \cap \mathcal{H}^{k\otimes l} = \emptyset$ for all $k \in \text{subtree}(i)$ and our proof is complete. ■

Theorem 5.30 is a general result for *hierarchical label costs* (Definition 5.24), but recall that *tree-structured label costs* (Definition 5.21) are a simple special case. We can now prove Theorem 5.22 simply by proving that constant $c' = 0$ for tree-structured label costs.

Proof of Theorem 5.22. Start with the definition of c' (Definition 5.28), *i.e.* given some hierarchical label costs (H, π)

$$c' = \max_{H(L)>0} |I(L)| - 1 \;\; \text{where} \;\; I(L) = I \subset \mathcal{I}(j) \;\; \text{for some } j \text{ such that } \bigcup_{i \in I} \mathcal{L}_i = L$$

It is sufficient to show that if (H, π) forms tree-structured label costs then $|I(\mathcal{L}_i)| = 1$ for all nodes $i \in \mathcal{L} \cup \mathcal{S}$. Clearly we must have $I(\mathcal{L}_i) = \{i\}$ because $\emptyset \subset \mathcal{L}_i \subset \mathcal{L}_j$ and the $\mathcal{L}_i$ are disjoint sets; no $\mathcal{L}_i$ can be written as the union of other siblings. Therefore $|I(L)| = 1$ for all L such that $H(L) > 0$. ■

5.6 Discussion

The main results of this chapter are a characterization of hierarchical costs (h-metrics and *hierarchical subsets*), the h-fusion algorithm itself, and a significant improvement on the approximation bound of α-expansion. These results are theoretical, but we foresee a number of applications for such energies.

Applications of hierarchical costs Computer vision is full of problems for which hierarchical costs are natural. The most obvious is using hierarchical context (*e.g.* [27]) for image segmentation, where in theory we could group the labels into some appropriate context as depicted at the outset of this chapter. This is a very rudimentary form of context but can be integrated with segmentation via an energy with hierarchical V and H terms.

In vision it is also common to assign labels that have geometric meaning, such as depths (*e.g.* [24, 95]), homographies or motions (*e.g.* [14, 70]). For example, Isack and Boykov [70] start with a set of observations (points, matches, *etc.*) and use random sampling to generate

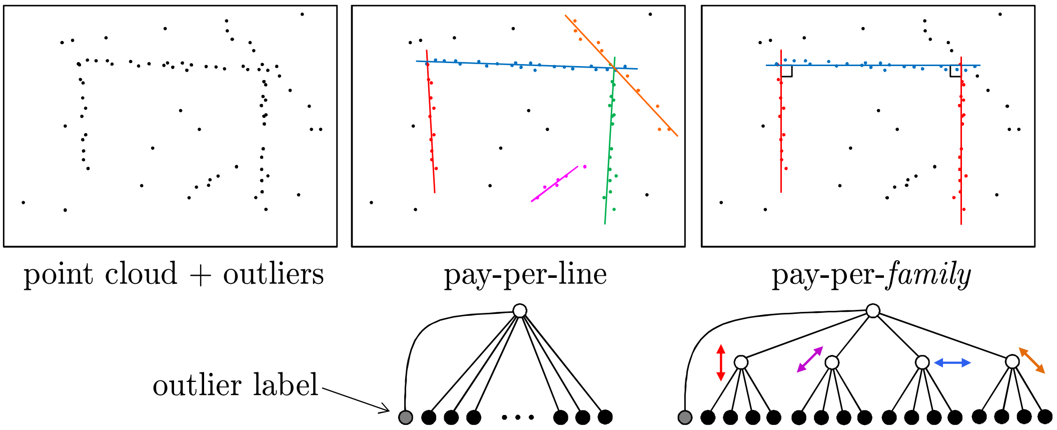


Figure 5.7: A visual depiction of how hierarchical line fitting might work with label costs. Each label corresponds to a possible line, and the points want to be labeled by a line that passes nearby. A label cost $H(\ell)$ discourages a line from being used unless there are enough points—otherwise the points take the outlier label (constant penalty per point). However, if we group lines into families of orientation, we could add group costs $H(\{\ell_k, \ldots\})$ and encourage solutions that use a few families of parallel lines.

hundreds of candidate geometric models, much the way RANSAC does [46]. They formulate the model fitting problem as a *labeling* problem where each label represents a candidate model. They find a labeling that corresponds to a good configurations of models, and do this by minimizing an energy. However, there are many situations where geometric models fall into a natural hierarchy. Figure 5.7 is a hypothetical example to illustrate this point. Analogous hierarchical relationships exist between, for example, a fundamental matrix (a rigid motion) and the family of homographies (families of correspondences) compatible with that fundamental matrix [63].

Furthermore, hierarchical costs can be useful for detecting patterns, for compression, and for learning a database of inter-dependent patches from images [59].

Generalizing facility location In the optimization and operations research communities, *uncapacitated facility location* (UFL) is a well-studied problem [134]. The UFL problem assigns a 'facility' to serve each client such that the cost to clients is balanced against the cost of 'opening' facilities. UFL is connected to our energy because if we let $\mathcal{L}$ denote the facilities and $\mathcal{P}$ denote the clients then every problem instance can be expressed as minimizing an energy of the form

$$E(f) = D(f) + \sum_{\ell \in \mathcal{L}} H(\ell)\delta_\ell(f). \tag{5.61}$$

In vision, the UFL problem has recently been applied to motion segmentation by Li [101] and by Lazic *et al.* [97].

There exist variants of UFL that allow for a hierarchy of facilities, *e.g.* [138, 126]. This generalization allows for more realistic modeling of complex interdependencies between facilities themselves. Some of these works derive constant-factor approximation bounds for hierarchical facility location, *e.g.* [77], but all such works assume *metric* client costs where the costs $D_p(\cdot)$ are computed as distances from a particular center. Without this assumption, Feige's hardness result still holds. Strategies for optimizing hierarchical UFL include linear programming relaxation, primal-dual algorithms and, very recently, message passing algorithms [55].

We can encode a kind of hierarchical facility cost with our framework as follows. Suppose facilities ℓ_1 and ℓ_2 require the services of facility ℓ_3, which costs 50 to open. A label cost $H(\{\ell_1, \ell_2, \ell_3\}) := 50$ correctly accounts for the shared dependency of ℓ_1 and ℓ_2 on ℓ_3. If we furthermore have a facility ℓ_4 that depends on both ℓ_3 and some facility ℓ_5 (cost 80), then our label costs should instead be $H(\{\ell_1, \ell_2, \ell_3, \ell_4\}) := 50$ and $H(\{\ell_4, \ell_5\}) := 80$.

Furthermore, our h-fusion algorithm can handle smooth costs V, which to the best of our knowledge are novel for UFL. In the UFL setting, $V(f_p, f_q)$ can encode an explicit preference that clients p and q be serviced by the same facility. When clients are social, there are many scenarios where such a preference makes sense. When client costs D are metric (*e.g.* Euclidean distance) then this preference is implicitly encoded in D. However, when the client costs are not metric, such as clients connected by an irregular network despite being physically close, then our smooth costs V may be useful for modeling such problems.

Improving our bound Recall that minimizing label costs is NP-hard by reduction from SET-COVER. Due to a hardness result by Feige [43], we know that a $\ln |\mathcal{P}|$-approximation is the best possible SET-COVER in polynomial-time. However, Hochbaum [64] gave a simple greedy algorithm for SET-COVER and proved that it yields precisely a $\ln |\mathcal{P}|$-approximation, the best possible according to Feige. If label costs are arbitrary in (5.61), then α-expansion's bound is also arbitrarily bad. So, there is a huge gap between what α-expansion can achieve on (5.61) versus what Hochbaum's greedy algorithm can guarantee. For hierarchical label costs alone, it is possible to use Hochbaum's algorithm as a subroutine within h-fusion (rather than using α-expansion). One may ask if h-fusion would inherit better approximation guarantees in that case. When smooth costs V are incorporated, it is not clear how Hochbaum's approach could be usefully applied, yet such a "best of both worlds" solution would be valuable.

Bibliography

[1] Ashraf M. Abdelbar and Sandra M. Hedetniemi. Approximating MAPs for belief networks is NP-hard and other theorems. *Artificial Intelligence*, 102(1):21–38, 1998.

[2] Ravindra K. Ahuja, Özlem Ergun, James B. Orlin, and Abraham P. Punnen. A survey of very large-scale neighborhood search techniques. *Discrete Applied Mathematics*, 123(1–3):75–202, 2002.

[3] Hirotugu Akaike. A new look at statistical model identification. *IEEE Transactions on Automatic Control*, 19:716–723, 1974.

[4] Karteek Alahari, Pushmeet Kohli, and Philip H.S. Torr. Dynamic Hybrid Algorithms for MAP Inference in Discrete MRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 32:1846–1857, 2010.

[5] Ben Appleton and Hugues Talbot. Globally Minimal Surfaces by Continuous Maximal Flows. *IEEE Transactions on Pattern Recognition and Machine Intelligence (TPAMI)*, 28(1):106–118, 2006.

[6] Ismail Ben Ayed and Amar Mitiche. A Region Merging Prior for Variational Level Set Image Segmentation. *IEEE Transactions on Image Processing (TIP)*, 17(12):2301–2311, 2008.

[7] Dj. A. Babayev. Comments on the note of Frieze. *Mathematical Programming*, 7(1):249–252, December 1974.

[8] Francis Bach. Structured Sparsity-Inducing Norms through Submodular Functions. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

[9] Olga Barinova, Victor Lempitsky, and Pushmeet Kohli. On the Detection of Multiple Object Instances using Hough Transforms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2010.

[10] Yair Bartal. On approximating arbitrary metrics by tree metrics. In *ACM Symposium on Theory of Computing (STOC)*, 1998.

[11] Dhruv Batra, A. C. Gallagher, Devi Parikh, and Tsuhan Chen. Beyond trees: MRF inference via outer-planar decomposition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2010.

[12] Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[13] Julian Besag. On the Statistical Analysis of Dirty Pictures. *Journal of the Royal Statistical Society B*, 48(3):259–302, 1986.

[14] Stan Birchfield and Carlo Tomasi. Multiway cut for stereo and motion with slanted surfaces. In *International Conference on Computer Vision (ICCV)*, 1999.

[15] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, August 2006.

[16] Andrew Blake and Andrew Zisserman. *Visual Reconstruction*. MIT Press, Cambridge, MA, 1987.

[17] Avrim Blum and Shuchi Chawla. Learning from labeled and unlabeled data using graph mincuts. In *International Conference on Machine Learning (ICML)*, June 2001.

[18] Endre Boros and Peter L. Hammer. Pseudo-boolean optimization. *Discrete Applied Mathematics*, 123(1–3):155–225, 2002.

[19] Yuri Boykov and Marie-Pierre Jolly. Interactive Organ Segmentation Using Graph Cuts. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI), LNCS 1935*, October 2000.

[20] Yuri Boykov and Marie-Pierre Jolly. Interactive Graph Cuts for Optimal Boundary and Region Segmentation of Objects in N-D Images. In *International Conference on Computer Vision (ICCV)*, 2001.

[21] Yuri Boykov and Vladimir Kolmogorov. Computing Geodesics and Minimal Surfaces via Graph Cuts. In *International Conference on Computer Vision (ICCV)*, 2003.

[22] Yuri Boykov and Vladimir Kolmogorov. An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision. *IEEE Transactions on Pattern Recognition and Machine Intelligence (TPAMI)*, 29(9):1124–1137, 2004.

[23] Yuri Boykov, Olga Veksler, and Ramin Zabih. Markov Random Fields with Efficient Approximations. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, June 1998.

[24] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast Approximate Energy Minimization via Graph Cuts. *IEEE Transactions on Pattern Recognition and Machine Intelligence (TPAMI)*, 23(11):1222–1239, 2001.

[25] Thomas Brox and Joachim Weickert. Level set based segmentation of multiple objects. In *Pattern Recognition*, volume 3175 of *LNCS*, pages 415–423, 2004.

[26] Kenneth P. Burnham and David R. Anderson. *Model Selection and Multimodel Inference*. Springer, 2002.

[27] Myung Jin Choi, Joseph J. Lim, Antonio Torralba, and Alan S. Willsky. Exploiting Hierarchical Context on a Large Database of Object Categories. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2010.

[28] Thomas H. Cormen, Charles E. Leiserson, Robert L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, 2001.

[29] G. Cornuejols, M. L. Fisher, and G. L. Nemhauser. Location of Bank Accounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms. *Management Science*, 23(8):789–810, 1977.

[30] G. Cornuejols, G. L. Nemhauser, and L. A. Wolsey. The Uncapacitated Facility Location Problem. Technical Report 605, Operations Research, Cornell University, August 1983.

[31] William Cunningham and Lawrence Tang. Optimal 3-Terminal Cuts and Linear Programming. In *Integer Programming and Combinatorial Optimization*, volume 1610 of *LNCS*, pages 114–125. 1999.

[32] William H. Cunningham. Minimum cuts, modular functions, and matroid polyhedra. *Networks*, 15:205–215, 1985.

[33] E. Dahlhaus, D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, and M. Yannakakis. The Complexity of Multiterminal Cuts. *SIAM Journal on Computing*, 23(4):864–894, 1994.

[34] Andrew Delong. Quantifier elimination as investigative tool for proofs: Application to graph cuts. CS829 project, University of Western Ontario, April 2007.

[35] Andrew Delong and Yuri Boykov. A Scalable Graph-Cut Algorithm for N-D Grids. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2008.

[36] Andrew Delong and Yuri Boykov. Globally Optimal Segmentation of Multi-Region Objects. In *International Conference on Computer Vision (ICCV)*, October 2009.

[37] Andrew Delong, Lena Gorelick, Olga Veksler, and Yuri Boykov. Minimizing Energies with Hierarchical Costs. *International Journal of Computer Vision (IJCV)*, 2011 (submitted).

[38] Andrew Delong, Anton Osokin, Hossam N. Isack, and Yuri Boykov. Fast Approximate Energy Minimization with Label Costs. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2010.

[39] Andrew Delong, Anton Osokin, Hossam N. Isack, and Yuri Boykov. Fast Approximate Energy Minimization with Label Costs. *International Journal of Computer Vision (IJCV)*, 2011. (in press).

[40] 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*, 39(1):1–38, 1977.

[41] E. A. Dinic. Algorithm for solution of a problem of maximum ow in networks with power estimation. *Soviet Math. Dokl*, 11:1266–1280, 1970.

[42] Olivier Duchenne, Jean-Yves Audibert, Renaud Keriven, Jean Ponce, and Florent Segonne. Segmentation by transduction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2008.

[43] Uriel Feige. A Threshold of $\ln n$ for Approximating Set Cover. *Journal of the ACM*, 45(4):634–652, 1998.

[44] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial Structures for Object Recognition. *International Journal of Computer Vision (IJCV)*, 61(1):55–79, 2005.

[45] Mario A. T. Figueiredo and Anil K. Jain. Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 24(3):381–396, 2002.

[46] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[47] L. Ford and D. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.

[48] James R. Foulds, Nicholas Navaroli, Padhraic Smyth, and Alexander T. Ihler. Revisiting MAP Estimation, Message Passing and Perfect Graphs. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[49] Daniel Freedman and Petros Drineas. Energy minimization via graph cuts: settling what is possible. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[50] Daniel Freedman and Tao Zhang. Interactive Graph Cut Based Segmentation With Shape Priors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[51] Alan M. Frieze. A cost function property for plant location problems. *Mathematical Programming*, 7(1):245–248, December 1974.

[52] Satoru Fujishige. *Submodular functions and optimization*. Elsevier, 2005.

[53] D. Geiger, A. Gupta, L. A. Costa, and J. Vlontzos. Dynamic-programming for detecting, tracking, and matching deformable contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 17(3):294–302, March 1995.

[54] Stuart Geman and Donald Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[55] Inmar E. Givoni, Clement Chung, and Brendan J. Frey. Hierarchical Affinity Propagation. In *Uncertainty in Artificial Intelligence (UAI)*, July 2011.

[56] Amir Globerson and Tommi Jaakkola. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

[57] Andrew V. Goldberg, Sagi Hed, Haim Kaplan, Robert E. Tarjan, and Renato F. Werneck. Maximum Flows by Incremental Breadth-First Search. In *Algorithms ESA*, 2011.

[58] Andrew V. Goldberg and Robert E. Tarjan. A new approach to the maximum-ow problem. *Journal of the Association for Computing Machinery (JACM)*, 35(4):921–940, 1988.

[59] Lena Gorelick, Andrew Delong, Olga Veksler, and Yuri Boykov. Recursive MDL via Graph Cuts: Application to Segmentation. In *International Conference on Computer Vision (ICCV)*, November 2011.

[60] Stephen Gould, Fernando Amat, and Daphne Koller. Alphabet SOUP: A Framework for Approximate Energy Minimization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2009.

[61] Leo Grady. Multilabel Random Walker Image Segmentation Using Prior Models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2005.

[62] D. Greig, B. Porteous, and A. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society B*, 51(2):271–279, 1989.

[63] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[64] Dorit S. Hochbaum. Heuristics for the fixed cost median problem. *Mathematical Programming*, 22(1):148–162, 1982.

[65] Dorit S. Hochbaum. An efficient algorithm for image segmentation, Markov random fields and related problems. *Journal of the ACM (JACM)*, 48:686–701, 2001.

[66] Derek Hoiem, Alexei A. Efros, and Martal Hebert. Recovering Surface Layout from an Image. *International Journal of Computer Vision (IJCV)*, 75(1), October 2007.

[67] Derek Hoiem, Carsten Rother, and John Winn. 3D LayoutCRF for Multi-View Object Class Recognition and Segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[68] Hossam N. Isack. Spatially Coherent Multi-Model Fitting. Master's thesis, University of Western Ontario, London, Canada, April 2009.

[69] Hossam N. Isack and Yuri Boykov. Energy-based Geometric Multi-Model Fitting. Technical Report 735, University of Western Ontario, March 2010. (Submitted to IJCV).

[70] Hossam N. Isack and Yuri Boykov. Energy-based Geometric Multi-Model Fitting. *International Journal of Computer Vision (IJCV)*, 2011. (in press).

[71] Hiroshi Ishikawa. Exact Optimization for Markov Random Fields with Convex Priors. *IEEE Transactions on Pattern Recognition and Machine Intelligence (TPAMI)*, 25(10):1333–1336, 2003.

[72] Tony Jebara. Map estimation, message passing, and perfect graphs. In *Uncertainty in Artificial Intelligence (UAI)*, 2009.

[73] Stefanie Jegelka and Jeff Bilmes. Approximation Bounds for Inference using Cooperative Cut. In *International Conference on Machine Learning (ICML)*, 2011.

[74] Michael I. Jordan. *Learning in Graphical Models*. MIT Press, 1999.

[75] Kyomin Jung, Pushmeet Kohli, and Devavrat Shah. Local Rules for Global MAP: When Do They Work? In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

[76] Evangelos Kalogerakis, Aaron Hertzmann, and Karan Singh. Learning 3d mesh segmentation and labeling. In *ACM SIGGRAPH*, 2010.

[77] Erez Kantor and David Peleg. Approximate hierarchical facility location and applications to the bounded depth Steiner tree and range assignment problems. *Journal of Discrete Algorithms*, 7(3):341–362, 2009.

[78] Jon Kleinberg and Éva Tardos. Approximation Algorithms for Classification Problems with Pairwise Relationships: Metric Labeling and Markov Random Fields. *Journal of the ACM*, 49(5), 2002.

[79] Pushmeet Kohli, M. Pawan Kumar, and Philip H. S. Torr. $\mathcal{P}^3$ & Beyond: Solving Energies with Higher Order Cliques. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[80] Pushmeet Kohli, L'ubor Ladický, and Philip H. S. Torr. Robust Higher Order Potentials for Enforcing Label Consistency. *International Journal on Computer Vision (IJCV)*, 82(3):302–324, 2009.

[81] Pushmeet Kohli, Jonathan Rihan, Matthieu Bray, and Philip H. S. Torr. Simultaneous Segmentation and Pose Estimation of Humans Using Dynamic Graph Cuts. *International Journal of Computer Vision (IJCV)*, 79(3):285–298, Sept 2008.

[82] Pushmeet Kohli and Philip H. S. Torr. Dynamic Graph Cuts for Efficient Inference in Markov Random Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29:2079–2088, 2007.

[83] Vladimir Kolmogorov. Convergent Tree-Reweighted Message Passing for Energy Minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(10):1568–1583, October 2006.

[84] Vladimir Kolmogorov, Yuri Boykov, and Carsten Rother. Applications of Parametric Maxflow in Computer Vision. In *International Conference on Computer Vision (ICCV)*, November 2007.

[85] Vladimir Kolmogorov and Carsten Rother. Minimizing non-submodular functions with graph cuts—a review. *IEEE Transactions on Pattern Recognition and Machine Intelligence (TPAMI)*, 29(7), 2007.

[86] Vladimir Kolmogorov and Akiyoshi Shioura. New Algorithms for Convex Cost Tension Problem with Application to Computer Vision. *Discrete Optimization*, 6(4):378–393, November 2009.

[87] Vladimir Kolmogorov and Ramin Zabih. What Energy Functions Can Be Optimized via Graph Cuts. *IEEE Transactions on Pattern Recognition and Machine Intelligence (TPAMI)*, 26(2):147–159, 2004.

[88] Vladimir Kolmogorov, Ramin Zabih, and Steven Gortler. Generalized Multi-camera Scene Reconstruction Using Graph Cuts. In *Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, 2003.

[89] Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. Mrf energy minimization and beyond via dual decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 33:531–552, 2011.

[90] Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47:498–519, 2001.

[91] A. A. Kuehn and M. J. Hamburger. A Heuristic Program for Locating Warehouses. *Management Science*, 9(4):643–666, 1963.

[92] M. Pawan Kumar and Daphne Koller. MAP estimation of semi-metric MRFs via hierarchical graph cuts. In *Conference on Uncertainty in Artificial Intelligence*, pages 313–320, June 2009.

[93] M. Pawan Kumar and Philip H. S. Torr. Improved Moves for Truncated Convex Models. In *Advances in Neural Information Processing Systems (NIPS)*, volume 22, 2008.

[94] L'ubor Ladický, Chris Russell, Pushmeet Kohli, and Philip H. S. Torr. Graph Cut based Inference with Co-occurrence Statistics. In *European Conference on Computer Vision (ECCV)*, September 2010.

[95] L'ubor Ladický, Paul Sturgess, Chris Russell, Sunando Sengupta, Yalin Bastanlar, William Clocksin, and Philip H. S. Torr. Joint Optimisation for Object Class Segmentation and Dense Stereo Reconstruction. In *British Machine Vision Conference (BMVC)*, 2010.

[96] Nevena Lazic, Brendan J. Frey, and Parham Aarabi. Solving the Uncapacitated Facility Location Problem Using Message Passing Algorithms. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

[97] Nevena Lazic, Inmar Givoni, Brendan J. Frey, and Parham Aarabi. FLoSS: Facility Location for Subspace Segmentation. In *International Conference on Computer Vision (ICCV)*, 2009.

[98] Yvan G. Leclerc. Constructing simple stable descriptions for image partitioning. *International Journal of Computer Vision (IJCV)*, 3(1):73–102, May 1989.

[99] Victor Lempitsky, Yuri Boykov, and Denis Ivanov. Oriented visibility for multiview reconstruction. In *European Conference on Computer Vision*, May 2006.

[100] Victor Lempitsky, Carsten Rother, Stephan Roth, and Andrew Blake. Fusion moves for markov random field optimization. *IEEE Transactions on Pattern Analysis and Machine Inference (TPAMI)*, 32:1392–1405, August 2010.

[101] Hongdong Li. Two-view Motion Segmentation from Linear Programming Relaxation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[102] Kang Li, Xiaodong Wu, Danny Z. Chen, and Milan Sonka. Optimal Surface Segmentation in Volumetric Images—A Graph-Theoretic Approach. *IEEE Transactions on Pattern Recognition and Machine Intelligence (TPAMI)*, 28(1), 2006.

[103] Stan Z. Li. *Markov Random Field Modeling in Image Analysis*. Springer, 1994.

[104] Jiangyu Liu and Jian Sun. Parallel graph-cuts by adaptive bottom-up merging. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2010.

[105] Xiaoqing Liu, Olga Veksler, and Jagath Samarabandu. Graph Cut with Ordering Constraints on Labels and its Applications. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2008.

[106] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal on Computer Vision (IJCV)*, 60:91–110, 2004.

[107] David J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[108] G. Miller and J. Naor. Flows in planar graphs with multiple sources and sinks. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 1991.

[109] T.J. Mitchell and J.J. Beauchamp. Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032, 1988.

[110] David Mumford and Jayant Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on Pure and Applied Mathematics*, 42(5):577–685, July 1989.

[111] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions – I. *Mathematical Programming*, 14(1):265–294, 1978.

[112] Carl Olsson, Martin Byröd, Niels Christian Overgaard, and Fredrik Kahl. Extending Continuous Cuts: Anisotropic Metrics and Expansion Moves. In *International Conference on Computer Vision (ICCV)*, October 2009.

[113] Anton Osokin, Dmitry Vetrov, and Vladimir Kolmogorov. Submodular Decomposition Framework for Inference in Associative Markov Networks with Global Constraints. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2011.

[114] Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, 1982.

[115] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[116] J. C. Picard and H. D. Ratliff. Minimum cuts and related problems. *Networks*, 5:357–370, 1974.

[117] Thomas Pock, Antonin Chambolle, Horst Bischof, and Daniel Cremers. A Convex Relaxation Approach for Computing Minimal Partitions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2009.

[118] Thomas Pock, Thomas Schoenemann, Gottfried Graber, Horst Bischof, and Daniel Cremers. A Convex Formulation of Continuous Multi-Label Problems. In *European Conference on Computer Vision (ECCV)*, October 2008.

[119] Renfrey B. Potts. Some generalized order-disorder transformations. *Mathematical Proceedings of the Cambridge Philosophical Society*, 48:106–109, 1952.

[120] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[121] Srikumar Ramalingam, Pushmeet Kohli, Karteek Alahari, and Philip H. S. Torr. Exact Inference in Multi-label CRFs with Higher Order Cliques. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2008.

[122] Sashank J. Reddi, Sunita Sarawagi, and Sundar Vishwanathan. MAP estimation in Binary MRFs via Bipartite Multi-cuts. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

[123] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. GrabCut: Interactive Foreground Extraction using Iterated Graph Cuts. In *ACM SIGGRAPH*, 2004.

[124] Carsten Rother, Vladimir Kolmogorov, Victor Lempitsky, and Martin Szummer. Optimizing Binary MRFs via Extended Roof Duality. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2007.

[125] H. Rusinek, Y. Boykov, M. Kaur, S. Wang, L. Bokacheva, J. Sajous, A. Huang, S. Heller, and V. Lee. Performance of an automated segmentation algorithm for 3D MR renography. *Magnetic Resonance in Medicine*, 2006.

[126] Güvenç Sahin and Haldun Süral. A review of hierarchical facility location models. *Computers and Operations Research*, 34(8):2310–2331, 2007.

[127] Dmitri Schlesinger. Exact Solution of Permuted Submodular MinSum Problems. In *Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, 2007.

[128] Frank R. Schmidt, Eno Töppe, and Daniel Cremers. Efficient planar graph cuts with applications in computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Miami, Florida, June 2009.

[129] Nicol N. Schraudolph and Dmitry Kamenetsky. Efficient exact inference in planar ising models. In *Advances in Neural Information Processing Systems (NIPS)*, 2008.

[130] Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, 1986.

[131] Emre Sefer and Carl Kingsford. Metric Labeling and Semi-metric Embedding for Protein Annotation Prediction. In *Research in Computational Molecular Biology*, 2011.

[132] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22(8):888–905, August 2000.

[133] Y. Shimony. Finding the MAPs for belief networks is NP-hard. *Artificial Intelligence*, 68(2):399–410, 1994.

[134] David B. Shmoys, Éva Tardos, and Karen Aardal. Approximation algorithms for facility location problems (extended abstract). In *ACM Symposium on Theory of Computing (STOC)*, pages 265–274, 1998.

[135] Petter Strandmark and Fredrik Kahl. Parallel and distributed graph cuts by dual decomposition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2010.

[136] Minghe Sun. A Tabu Search Heuristic for the Uncapacitated Facility Location Problem. In *Metaheuristic Optimization via Memory and Evolution*, volume 30, pages 191–211. Springer US, 2005.

[137] Kah Kay Sung and Tomaso Poggio. Example based learning for view-based human face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 20:39–51, 1995.

[138] Zoya Svitkina and Éva Tardos. Facility location with hierarchical facility costs. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2006.

[139] Richard Szeliski, Ramin Zabih, Daniel Scharstein, Olga Veksler, Vladimir Kolmogorov, Aseem Agarwala, Marshall Tappen, and Carsten Rother. A comparative study of energy minimization methods for Markov random fields with smoothness-based priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(6):1068–1080, June 2008.

[140] Martin Szummer and Tommi Jaakkola. Partially labeled classication with markov random walks. In *Advances in Neural Information Processing Systems (NIPS)*, December 2001.

[141] Martin Szummer, Pushmeet Kohli, and Derek Hoiem. Learning CRFs using Graph Cuts. In *European Conference on Computer Vision (ECCV)*, 2008.

[142] Daniel Tarlow, Dhruv Batra, Pushmeet Kohli, and Vladimir Kolmogorov. Dynamic Tree Block Coordinate Ascent. In *International Conference on Machine Learning (ICML)*, 2011.

[143] Ben Taskar, Vassil Chatalbashev, and Daphne Koller. Learning Associative Markov Networks. In *International Conference on Machine Learning (ICML)*, 2004.

[144] Philip H. S. Torr. Geometric Motion Segmentation and Model Selection. *Philosophical Transactions of the Royal Society A*, pages 1321–1340, 1998.

[145] Roberto Tron and Rene Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[146] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research (JMLR)*, 6(2):1453–1484, 2006.

[147] Naonori Ueda, Ryohei Nakano, Zoubin Ghahramani, and Geoffrey E. Hinton. SMEM Algorithm for Mixture Models. *Neural Computation*, 12(9):2109–2128, 2000.

[148] Johannes Ulén, Petter Strandmark, and Fredrik Kahl. Optimization for Multi-Region Segmentation of Cardiac MRI. In *MICCAI Workshop on Statistical Atlases and Computational Models of the Heart: Imaging and Modelling Challenges*, 2011.

[149] A. Vazquez, A. Flammini, A. Maritan, and A. Vespignani. Global protein function prediction from protein-protein interaction networks. *Nature Biotechnology*, 6, 2003.

[150] Olga Veksler. Stereo Correspondence by Dynamic Programming on a Tree. In *IEEE Confeence on Computer Vision and Pattern Recognition (CVPR)*, June 2005.

[151] Olga Veksler. Graph Cut Based Optimization for MRFs with Truncated Convex Priors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2007.

[152] Olga Veksler. Star Shape Prior for Graph-Cut Image Segmentation. In *European Conference on Computer Vision (ECCV)*, 2008.

[153] Sara Vicente, Vladimir Kolmogorov, and Carsten Rother. Graph cut based image segmentation with connectivity priors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2008.

[154] George Vogiatzis, Philip H. S. Torr, and Roberto Cipolla. Multi-view stereo via volumetric graph-cuts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2005.

[155] Nhat Vu and B.S. Manjunath. Shape Prior Segmentation of Multiple Objects with Graph Cuts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[156] Stanislav Živný, David A. Cohen, and Peter G. Jeavons. The Expressive Power of Binary Submodular Functions. *Discrete Applied Mathematics*, 2009.

[157] Martin J. Wainwright, Tommi S. Jaakkola, and Alan S. Willsky. Tree-reweighted belief propagation and approximate ML estimation by pseudo-moment matching. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2003.

[158] Tomáš Werner. A Linear Programming Approach to Max-sum Problem: A Review. *IEEE Transactions on Pattern Recognition and Machine Intelligence (TPAMI)*, 29(7):1165–1179, July 2007.

[159] Tomáš Werner. High-arity Interactions, Polyhedral Relaxations, and Cutting Plane Algorithm for Soft Constraint Optimisation (MAP-MRF). In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2008.

[160] Oliver J. Woodford, Carsten Rother, and Vladimir Kolmogorov. A Global Perspective on MAP Inference for Low-Level Vision. In *International Conference on Computer Vision (ICCV)*, October 2009.

[161] Chen Yanover and Yair Weiss. Approximate inference and protein folding. In *Advances in Neural Information Processing Systems (NIPS)*, 2002.

[162] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. *Understanding belief propagation and its generalizations*, pages 239–269. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[163] Jing Yuan and Yuri Boykov. TV-Based Multi-Label Image Segmentation with Label Cost Prior. In *British Machine Vision Conference (BMVC)*, Sept 2010.

[164] Ramin Zabih and Vladimir Kolmogorov. Spatially Coherent Clustering with Graph Cuts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2004.

[165] Quan Zhou, Tianfu Wu, Wenyu Liu, and Song-Chun Zhu. Scene Parsing by Data-Driven Cluster Sampling. *International Journal of Computer Vision (IJCV)*, 2011. under review.

[166] Song-Chun Zhu and Alan L. Yuille. Region competition: unifying snakes, region growing, and Bayes/MDL for multiband image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 18(9):884–900, 1996.

[167] Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *International Conference on Machine Learning (ICML)*, August 2003.

[168] M. Zuliani, C. S. Kenney, and B. S. Manjunath. The multiRANSAC algorithm and its application to detect planar homographies. In *International Conference on Image Processing (ICIP)*, 2005.

Curriculum Vitae

ANDREW DELONG

EDUCATION

2011 PhD in Computer Science, University of Western Ontario.

Thesis: *Advances in Graph-Cut Optimization: Multi-Surface Models, Label Costs, and Hierarchical Costs*
Advisor: Yuri Boykov

2006 MSc in Computer Science, University of Western Ontario.

Thesis: *A Scalable Max-Flow/Min-Cut Algorithm for Sparse Graphs*
Advisor: Yuri Boykov

2003 BMath in Honours Co-op Computer Science (with distinction), University of Waterloo.

AWARDS and DISTINCTIONS

2010–2011 Ontario Graduate Scholarship in Science and Technology (OGSST).

2008 Faculty of Science Graduate Student Teaching Award, Univ. of Western Ontario.

2007–2010 National Sciences and Engineering Research Council Postgraduate Scholarship.

2006–2007 Ontario Graduate Scholarship (OGS).

1998–2003 Term Dean's Honour List four times (10th percentile), University of Waterloo.

REFEREED PUBLICATIONS

8. A. Delong, L. Gorelick, O. Veksler, and Y. Boykov. Minimizing Energies with Hierarchical Costs. *International Journal of Computer Vision (IJCV)*, submitted October 2011.

7. L. Gorelick, A. Delong, O. Veksler, and Y. Boykov. Recursive MDL via Graph Cuts: Application to Segmentation. In *International Conference on Computer Vision (ICCV)*, November 2011.

6. A. Delong, L. Gorelick, F. R. Schmidt, O. Veksler, and Y. Boykov. Interactive Segmentation with Super-Labels. In *Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, July 2011. **(oral)**
5. A. Delong, A. Osokin, H. N. Isack, and Y. Boykov. Fast Approximate Energy Minimization with Label Costs. *International Journal of Computer Vision (IJCV)*, 2011 (in press).
4. A. Delong, A. Osokin, H. N. Isack, and Y. Boykov. Fast Approximate Energy Minimization with Label Costs. In *Computer Vision and Pattern Recognition (CVPR)*, June 2010.
3. A. Delong and Y. Boykov. Globally Optimal Segmentation of Multi-Region Objects. In *International Conference on Computer Vision (ICCV)*, October 2009. **(oral)**
2. A. Delong and Y. Boykov. A Scalable Graph-Cut Algorithm for N-D Grids. In *Computer Vision and Pattern Recognition (CVPR)*, June 2008.
1. Y. Boykov, V. Kolmogorov, D. Cremers, and A. Delong. An Integral Solution to Surface Evolution PDEs via Geo-Cuts. In *European Conf. on Computer Vision (ECCV)*, LNCS 3953, May 2006.

TEACHING EXPERIENCE (as full-time instructor)

- Introduction to Computer Graphics (Fall 2011)
- Fundamentals of Computer Science II (Fall 2010)

RELEVANT ACTIVITIES

- **Invited talks**: (2009) Lund University, (2010) University of Toronto, (2011) École Normal Superiéure, University of Oxford, ESIEE-Paris, University of Heidelberg, Technical University of Munich, University of Toronto.
- **Journal referee**: IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), SIAM Journal on Imaging Sciences (SIIMS), Image and Vision Computing (IMAVIS), SPIE Journal of Electronic Imaging (JEI), Pattern Recognition.
- **Conference referee**: IEEE International Conference on Computer Vision (ICCV), European Conference on Computer Vision (ECCV), IEEE Computer Vision and Pattern Recognition (CVPR), Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR), Eurographics.

PATENTS

1. A. Delong, Y. Boykov, D. Yu. “Region Based Push-Relabel Algorithm for Efficient Computation of Maximum Flow”. United States Patent App. #11/685,815 (to be issued, confirmation #5292).