Combinatorial and Convex Optimization for Probabilistic Models in Computer Vision

Pawan Kumar Mudigonda

Thesis submitted in partial fulfillment of the requirements of the award of

PhD

Oxford Brookes University

2008

Abstract

This thesis investigates the role of optimization in two areas of Computer Science: Computer Vision and Machine Learning. Specifically, we consider two well-known problems in Computer Vision, namely motion segmentation and object category specific image segmentation, and a fundamental problem in Machine Learning, known as maximum a posteriori (MAP) estimation of discrete probabilistic models.

In order to address the problem of motion segmentation, we propose a novel probabilistic model which is suitable for this application. Our model includes the effects of occlusion, lighting changes and motion blur. The segmentation of a given video is obtained by performing efficient inference on the probabilistic model in two stages: (i) In the first stage, an initial estimate of the model is obtained using a novel coarse-to-fine technique which reduces the time and memory required by the sum-product belief propagation algorithm; and (ii) In the second stage, the initial estimate is refined using the observation that the energy of the model can be efficiently reduced using the $\alpha\beta$-swap and α-expansion algorithms.

For object category specific image segmentation, we extend the probabilistic model used in previous approaches. Specifically, we incorporate it with an object category model which provides top-down information about the shape of the object. Given an image, its segmentation is determined using two new algorithmic contributions: (i) We propose efficient methods for obtaining samples of the object category models of our choice by matching them to the given image; and (ii) We make the (not obvious) observation that these samples can be quickly marginalized within the EM framework using one st-MINCUT operation. We compare our method with the state of the art approaches and demonstrate significant improvement.

Next, we present a theoretical analysis of previously proposed algorithms for MAP estimation which are based on convex relaxations. In particular, we show that a widely used linear programming (LP) relaxation strictly dominates (i.e. provides a better approximation than) some recently proposed Quadratic Programming (QP) and Second Order Cone Programming (SOCP) relaxations. We generalize this result to show that, despite the flexibility in the form of objective function and constraints offered by QP and SOCP, the LP relaxation dominates a large class of QP and SOCP relaxations. As a consequence of our analysis, we obtain two new SOCP relaxations which dominate the previous approaches.

Finally, we consider the problem of efficiently solving the new SOCP relaxations. To this end, we build upon the tree reweighted message passing framework. We propose convergent algorithms which iteratively optimize the Lagrangian dual of the SOCP relaxations. These algorithms allow us to empirically verify our theoretical analysis using both synthetic and real data.

Acknowledgements

I would like to thank my supervisors, Prof. Philip Torr and Prof. Andrew Zisserman, for their guidance and encouragement. This work would not have been possible without their contagious enthusiasm for research.

For having the patience to work with me and for numerous helpful discussions, I would like to thank my collaborators: Mukta Prasad, Pushmeet Kohli and Vladimir Kolmogorov. A special thanks to Mukta who excelled in her daunting role as my best friend (and agony aunt).

Members of the Brookes Vision Group, past and present, created a wonderful research environment. I am grateful to Pushmeet Kohli (my reliable soundboard for ideas), Carl Ek, Karteek Alahari (my two favourite Brookes chefs), Prem Fernando, Christophe Restif, Matthieu Bray, Samy Sun, Chris Russell, Jon Rihan, Lubor Ladicky and Srikumar Ramalingam.

I would also like to thank my sponsors: Oxford Brookes University, EPSRC, and PASCAL Network of Excellence.

I am indebted to my undergraduate advisors, Prof. C.V. Jawahar and Prof. P.J. Narayanan. The courses I studied and the projects I worked on under their supervision inspired me to pursue a PhD in Computer Vision.

Most importantly, I would like to thank my family for their support and invaluable advice.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Great strides have been made in the use of optimization. Models with thousands of variables and constraints are being solved regularly and the results are being applied routinely, but only by a relatively small number of organizations.
- George L. Nemhauser, *The Age of Optimization*, September, 1993.

The words of Nemhauser (among others) seem to have struck a chord with many researchers. In the past decade, the optimization literature has grown to form the backbone of several areas in Computer Science. This thesis represents our attempt to further highlight its scope and importance.

Optimization itself is a vast field, which spans the range from the simplest greedy algorithms (e.g. for determining the minimum spanning tree [52, 72]) to the most complex Interior Point methods (e.g. for truss design [5]) and beyond. Of particular interest to us is the application of optimization to the *Probabilistic Models* framework. The study and development of probablistic models, like optimization, has had a profound impact on many areas of Science, from Statistical Physics to Molecular Biology. The focus of this thesis is on two areas: Computer Vision and Machine Learning.

Computer Vision is concerned with the theory and technology for building artificial systems that obtain information from images. The image data can take many forms, such as a video sequence, views from multiple cameras, or multi-dimensional data from a medical scanner. Some recent real-world applications of Computer Vision include camera tracking for special effects in movies, motion capture for animation and biometric systems such as fingerprint recognition for security systems. Machine Learning deals with the design and development of algorithms and techniques that allow computers to *learn*. Its major focus is to extract information (make inference) from data automatically, by computational and statistical methods. Progress in this area has lead to several useful applications such as intelligent spam filters and handwritten text recognition for creating digital libraries[1]. Each of these two areas contributes immensely to the development of the other. While Machine Learning provides the means to solve Computer Vision problems, Computer Vision in turn opens up new avenues for Machine Learning to explore.

Our work in these areas can be divided into two parts. In the first part we look at two well-known problems in Computer Vision. We show how extending some previously known algorithms in Machine Learning (e.g. by improving their efficiency or developing some new observations about them) allows us to solve these problems accurately. In the second part we consider a fundamental problem of Machine Learning which plays a central role in many Computer Vision applications (including the ones considered in the first part). We present a theoretical analysis of a subclass of approaches for this problem, identifying the one that provides the best solution. Our analysis leads to some new and more

[1]Source of definitions: Wikipedia.

accurate methods which can be applied efficiently to tackle the hardest instances of the problem.

We begin by describing and motivating the above mentioned problems.

1.1. Optimization for Computer Vision

In this work, we are interested in the following two Computer Vision problems: (i) motion segmentation; and (ii) object category specific image segmentation. Below, we describe the two problems and motivate them by listing some of their applications. We also specify the difficulties that are inherent in these problems.

1.1.1 Motion Segmentation

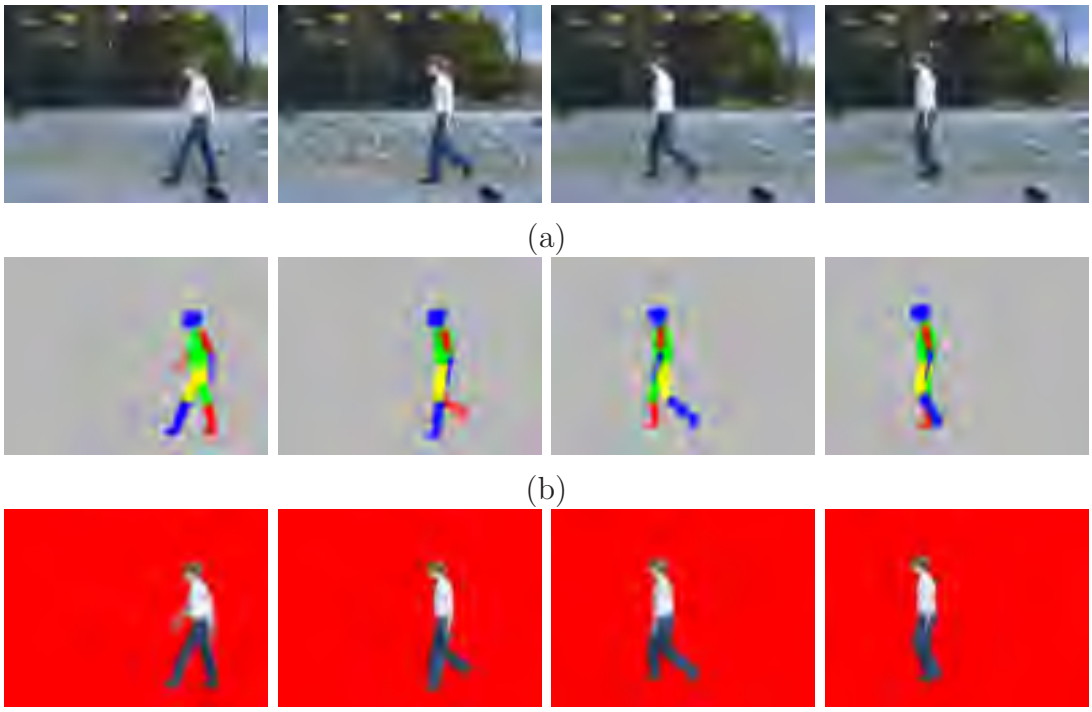
Given a video sequence of an object, we define the problem of motion segmentation as that of dividing the scene into rigidly moving segments in order to obtain a foreground-background segmentation. *This requires us to determine the number of segments and learn the shape, appearance and transformations of each segment in every frame of the video sequence.* For example, Fig. 1.1(a) shows a video sequence of a human walking. The shape and transformations of the learnt segments is shown in Fig. 1.1(b). The appearance of the segments is shown in Fig. 1.1(c).

Motivation: There are many potential applications of motion segmentation. For example,

- Understanding Videos – Given a video sequence, we may wish to understand the nature of the activities taking place in it. For example, for a surveillance video, we would like to determine whether the moving object is a car or a human, whether it is stationary or moving and which direction it is travelling in. The enormous amount of video data available to us makes it impossible to carry out such an analysis manually. Motion segmentation would provide an elegant and accurate solution for this problem.

- Driving Avatars for Animation – In the animation industry, avatars are currently driven using manual input which requires a lot of time and is prone to being inaccurate. The movements of a human or quadruped, learnt from a reliable motion segmentation algorithm, could be used as a time and cost efficient alternative for this task.

- Human Computer Interaction (HCI) – The current trend in HCI is to move away from conventional input devices such as mouse and keyboard, and



(a)

(b)

(c)

Figure 1.1: *An example of motion segmentation.* **(a)** *Four intermediate frames of an input video sequence of a person walking against a static background.* **(b)** *The shape of the rigidly moving segments (each shown in a different colour from the other) which constitute the entire visible body of the person. The transformations of the segments for all frames should also be learnt as shown.* **(c)** *Together with the appearance information, each frame of the video sequence can be segmented into foreground and background.*

towards more interactive and intuitive ways. A motion segmentation algorithm which can distinguish between different hand gestures would greatly facilitate this goal.

- Analyzing Motion for Medical Diagnosis – The diagnosis of several movement-related disorders such as Parkinson's disease and multiple sclerosis requires the computation of a large number of statistics related to the patient's gait. Motion segmentation can be used as an effective tool for obtaining these statistics.

Clearly, the problem of motion segmentation has great practical importance. However, it throws up a lot of challenges which have so far prevented researchers from solving it completely.

Challenges: A successful motion segmentation approach should address the following issues:

- Computer Vision – Parts of the scene may be occluded in one or more

frames of the video sequence (e.g. see Fig. 1.1 where the torso occludes an arm and the background). The appearance of the segments may also change due to lighting conditions and motion blur. A motion segmentation approach should be able to handle such changes and automatically determine the number of segments, together with their shape, appearance and transformations.

- Machine Learning – Given a video, we are faced with the problem of learning the representation of the scene through motion segmentation. The representation should be chosen to handle the Computer Vision challenges mentioned above. However, such a representation would be complex and the problem of learning it (i.e. performing inference) from a video would not lend itself to naive methods like exhaustive search. For example, consider a video sequence of f frames, each of which is of size $l \times b$. Also, let us suppose that the orientation of a given segment in a frame can be defined using one of h possible rotation angles. For such a case, the shape of a segment would be determined using $O(2^{l \times b})$ operations by exhaustive search (which is 2^{153600} even for reasonable frame sizes, $l = 320$ and $b = 480$). Even if the shape of the segment has been provided, its transformations for the entire video sequence would require $O(lbfh)$ operations (i.e. over 75 million for $f = 50$ and $h = 10$). Clearly, a useful motion segmentation algorithm should cleverly employ efficient Machine Learning inference algorithms (such as those proposed in [15]) to overcome this deficiency.

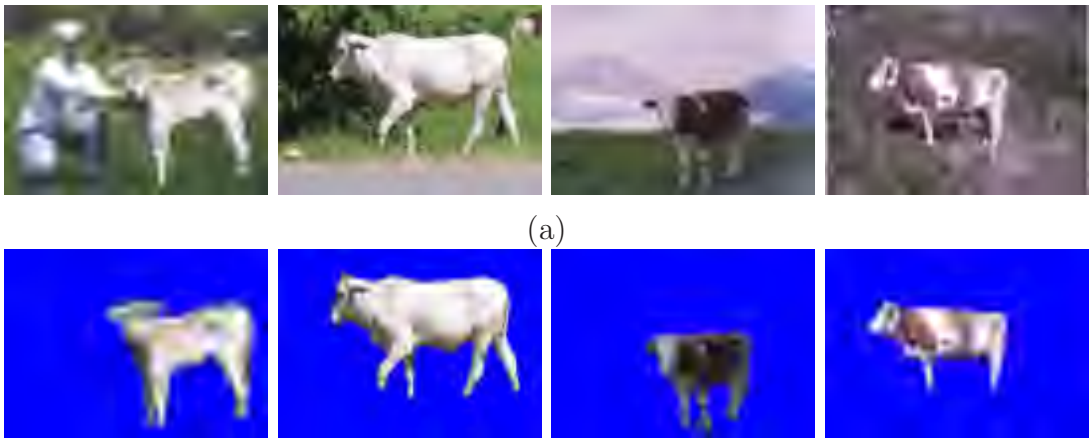
1.1.2 Object Category Specific Image Segmentation

Given an image containing an instance of an object category[1] of interest, our aim is to segment the pixels of the image into foreground (i.e the object) and background. *The segmentation obtained should be accurate and its shape should resemble that of the object.* For example, Fig. 1.2(a) shows some images of cows. Their corresponding segmentations are shown in Fig. 1.2(b).

Motivation: Object category specific image segmentation finds several applications, e.g.

- Automatic Photo Editing – Software such as Photoshop allows the user to segment objects (such as their pets) from images using manual interaction. However, the widespread availability of cameras has ensured that

[1]The term object category (or object class) refers to a collection of objects which can be described by a single noun, e.g. cows or horses. We use this term to imply a *visual* object category as opposed to a *functional* one.



(a)

(b)

Figure 1.2: *Examples of object category specific image segmentation.* **(a)** *Four images containing an instance of the object category 'cow'.* **(b)** *The segmentation of the images in (a) into foreground and background.*

even personal collections contain thousands of photographs. For such collections, manual interaction is no longer an attractive option and should to be substituted with automatic object category specific segmentation.

- Assisting Special Effects in Movies – Animated characters in movies are introduced by replacing proxy human actors with avatars. This requires the segmentation of a large number of frames (called rotoscoping), which is currently done manually. Automatic segmentation of proxy actors from frames would offer a more time and cost effective alternative.

- Improving Chroma-Keying – Imposing a synthetic background in TV video (e.g. weather telecast) is usually done using chroma-keying, where the broadcaster stands in front of a blue or green screen and a matte is pulled by background subtraction. However, chroma-keying can easily go wrong due to colour bleeding. The errors in segmentations can be greatly reduced by encouraging its shape to resemble that of a human.

- Interleaved Object Recognition and Segmentation – A good segmentation has been shown to greatly facilitate the localization of the object in the image. Object localization is the first step towards many applications such as surveillance and content based image and video retrieval.

Challenges: When addressing the problem of object category specific image segmentation, we must consider the following difficulties:

- Computer Vision – The shape and appearance of an object category varies greatly from one instance to the other. Further, articulated object category may also differ significantly in their poses in two images (see Fig. 1.2). Parts

of the object may also be occluded (i.e. either self-occlusion or occlusion from background). A good segmentation approach should be able to handle these variations and drive the shape of the segmentation to look like an object.

- Machine Learning – Similar to motion segmentation, the problem of object category specific image segmentation is also very large scale. For example, consider the task of detecting an instance of the object (required for segmentation) in a given image of size $l \times b$. If the object is made up of n parts, each of which can be in one of h poses, then an exhaustive search would detect the object in $O(n^h)$ time. Since the typical value of h is approximately $O(lb)$, exhaustive search would be computationally infeasible even for small image sizes such as $l = 320$ and $b = 480$. Similarly, the segmentation itself would require $O(2^{lb})$ time. A useful segmentation approach should overcome these deficiencies using efficient inference algorithms (such as those proposed in [23] for object detection and [13] for segmentation).

1.2. Optimization for Machine Learning

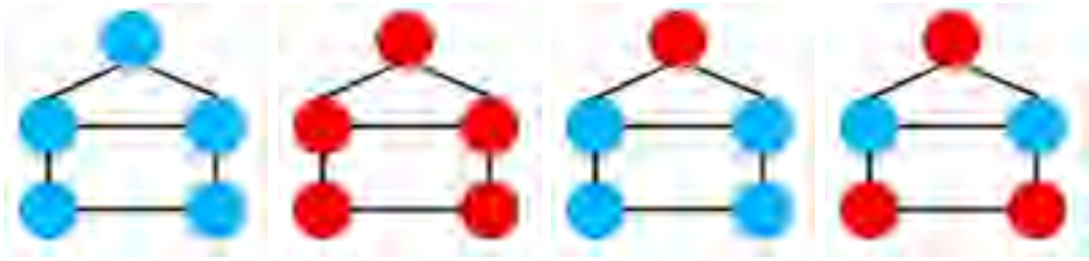
We now describe a fundamental problem in Machine Learning. We will postpone describing the notation associated with this problem to the next chapter, and provide a simpler graph theoretic formulation here as follows.

1.2.1 A Graph Theoretic Problem

Consider a graph $\mathcal{G} = \{\mathbf{V}, \mathbf{E}\}$, where $\mathbf{V} = \{V_0, V_1, \cdots, V_{n-1}\}$ is a set of n vertices and $\mathbf{E}$ is a set of edges. For such a graph, we can define a *colouring* as an assignment of one colour to each vertex from a given set of h possible colours. Associated with each colouring is a cost which consists of two types of terms (provided as input to the problem):

- A unary cost for assigning a colour i to V_a, for each vertex $V_a \in \mathbf{V}$.
- A pairwise cost for assigning colours i and j to V_a and V_b respectively, for each pair of vertices $(V_a, V_b) \in \mathbf{E}$.

As an example, consider the graph shown in Fig. 1.3 whose vertices can be coloured blue (with a unary cost of 1) or red (with a unary cost of 2). Suppose also that the pairwise cost for any pair of vertices $(V_a, V_b) \in \mathbf{E}$ is 0 if they are assigned the same colour, and 1 otherwise. For this case, Fig. 1.3 shows 4 of the 32 possible labellings, along with their costs. *The aim of the problem is to obtain the colouring with the minimum cost.*



Cost=5 Cost=10 Cost=8 Cost=12

Figure 1.3: *Four possible colourings of a graph consisting of five vertices and six edges. The total cost is shown below the corresponding colouring. Clearly, the first colouring has the minimum cost among of all 32 possible colourings, and hence is the solution of our problem.*

The above minimum cost colouring problem has a similar flavour to some of the fundamental problems in theoretical Computer Science, e.g. MAXCUT [31], multi-way cut [21], metric labelling [15, 42] and 0-extension [15, 39]. Further, it is well-known that solving a general instance of the minimum cost colouring problem (i.e. for a given graph $\mathcal{G}$ and some cost function) is extremely hard, especially for graphs with a large number of vertices. For this reason, computer scientists find this problem very interesting, and have devoted a vast amount of literature to developing approximate approaches for solving it. However, we note that the importance of the above problem goes beyond pure theoretical inquisitiveness. Specifically, it is very closely related to the problem of *maximum a posteriori* (MAP) estimation of a *probabilistic model*. For now, it is sufficient to note that MAP estimation plays a central role in many Computer Vision problems (including motion segmentation and object category specific segmentation). When we describe MAP estimation and its applications in detail, we hope the reader would easily be able find the analogy between the two problems.

1.2.2 How Should I Solve It?

As mentioned earlier, the minimum cost colouring problem (and the related MAP estimation problem) is, in general, extremely hard to solve exactly[1]. When we are faced with the task of solving it, one approach that we can adopt is to employ a previously proposed approximation algorithm. Some natural questions that arise now are, "Which algorithm would give me the best solution?" and "How quickly can I solve this algorithm?".

The task of answering the first question is made difficult by the presence of (seemingly) disconnected algorithms in the literature, which are hard to compare against each other (e.g. the min-sum BP and the st-MINCUT algorithms described

[1]We note however that some instances of this problem may offer themselves to computationally feasible methods, as will be seen in the next chapter.

in the next chapter). However, by restricting ourselves to approaches that share some common features, we can hope to make substantial progress in this direction.

For this work, we will consider a subclass of algorithms which *relax* (i.e. approximate) the MAP estimation problem to an easy-to-solve *Mathematical Optimization Problem*. Such algorithms are referred to as *relaxations*. Specifically, we will study the relaxations proposed in [17, 51, 55, 64, 75, 79, 101]. We can now simplify our earlier questions to, "Which relaxation would give me the best solution?" and "How quickly can I obtain the solution of this relaxation?". These two questions will be the focus of the second part of the thesis.

1.3. Contributions

Our contributions can be broadly divided in two areas, Computer Vision and Machine Learning.

1.3.1 Computer Vision

Motion Segmentation: We formulate the problem of motion segmentation using a novel probabilistic model. This model explicitly incorporates the effects of occlusion and changes in appearance due to lighting and motion blur. We present a method which automatically determines the number of rigidly moving segments present in a given video and obtains their shape, appearance and transformations for each frame. An initial estimate of the model is obtained by improving the efficiency of a well-known algorithm. In order to refine the initial estimate, we use the observation that the probability of the model can be increased using a computationally feasible approach.

Object Category Specific Image Segmentation: We extend the previously used probabilistic model for image segmentation (which provides bottom-up information) by combining it with an object category model (which provides top-down information about the shape of the object). Samples of the object category model are found by efficiently matching it to the given image. The desired segmentation is obtained by using the observation that samples of the object category model can be quickly marginalized within the EM framework [30].

1.3.2 Machine Learning

Analysis of Relaxations: We present a theoretical analysis of some previously proposed relaxations. We clearly specify a criterion for comparing relaxations. Using this criterion, we show that the equivalent relaxations of [17, 51, 101], first presented in 1998 and closely related to the work of Schlesinger [79] in 1976,

provide a better approximation than the recently proposed relaxations of [55, 64, 75]. We extend this result by presenting a large class of relaxations which are guaranteed not to be better than the approach of [17, 51, 79, 101]. Based on this analysis, we propose two new and more accurate relaxations. It is worth noting that our results are applicable for all problems related to MAP estimation, e.g. MAXCUT, multi-way cut, metric labelling and 0-extension.

Efficient Algorithms for Relaxations: Our novel relaxations can be solved only for small probabilistic models using standard softwares for optimization (which is equivalent to considering a small number of vertices for the graph theoretic problem in § 1.2.1). However, in areas like Computer Vision, it is common to have very large probabilistic models (equivalent to hundreds of thousands of vertices). In order to address this deficiency, we develop novel iterative algorithms which are guaranteed to converge. These algorithms allow us to empirically verify our theoretical analysis of relaxations.

1.4. Outline of the Thesis

Chapter 2 describes the framework of probabilistic models and their role in Computer Vision. Specifically, it shows how the solution to many Computer Vision problems can be obtained by answering the queries that arise naturally in this framework (e.g. MAP estimation which is closely related to the minimum cost colouring problem described in § 1.2.1). We also provide details of some previously proposed algorithms for (approximately) obtaining the desired solution. In chapter 3 we present an alternative formulation of the MAP estimation problem, which allows us to specify the exact form of the some previous relaxations (that were briefly mentioned in § 1.2.2).

In chapters 4 and 5 we consider two problems of Computer Vision (i.e. motion segmentation and object category specific image segmentation respectively) and provide an overview of the relevant literature. We show how the two problems can be formulated within the framework of probabilistic models. In order to solve these problems, we propose some novel extensions of the algorithms described in chapter 2. We provide comparisons with the state of the art methods and demonstrate significant improvements.

In chapter 6 we present an analysis of previously proposed relaxations. In particular, we identify the relaxation which is best suited for the problem of MAP estimation. This analysis leads us to propose two new and more accurate relaxations. The question of how to solve these new relaxations efficiently is answered in chapter 7. Specifically, we present convergent iterative algorithms to obtain the solution of these relaxations. We test these algorithms using both synthetic and real data, and show that the empirical results conform with the

theoretical analysis of chapter 6.

1.5. Publications

Part of the work described here has previously appeared as the following publications.

- Chapter 4
  - M. Pawan Kumar, P.H.S. Torr and A. Zisserman. *Learning Layered Pictorial Structures from Video.* ICVGIP, 2004.
  - M. Pawan Kumar, P.H.S. Torr and A. Zisserman. *Learning Layered Motion Segmentations of Video.* ICCV, 2005.
  - M. Pawan Kumar, P.H.S. Torr and A. Zisserman. *Learning Layered Motion Segmentations of Video.* To appear in IJCV, 2007.
- Chapter 5
  - M. Pawan Kumar, P.H.S. Torr and A. Zisserman. *Extending Pictorial Structures for Object Recognition.* BMVC, 2004.
  - M. Pawan Kumar, P.H.S. Torr and A. Zisserman. OBJ CUT. CVPR, 2005.
  - M. Pawan Kumar, P.H.S. Torr and A. Zisserman. *An Object Category Specific MRF for Segmentation.* Towards Category-Level Object Recognition, 2006.
- Chapter 6
  - M. Pawan Kumar, V. Kolmogorov and P.H.S. Torr. *An Analysis of Convex Relaxations for MAP Estimation.* NIPS, 2007.
- Chapter 7
  - M. Pawan Kumar and P.H.S. Torr. *Efficiently Solving Convex Relaxations for MAP Estimation.* Oxford Brookes Technical Report, 2007.

Chapter 2

Probabilistic Models and Inference Algorithms

2.1. The Labelling Problem

Many tasks in Computer Vision can be viewed as *labelling problems*. Given some observed data $\mathbf{D}$, the objective of the labelling problem is to classify a set of random variables $\mathbf{v} = \{v_0, v_1, \cdots, v_{n-1}\}$ by assigning one label to each variable from the set $\mathbf{l} = \{l_0, l_1, \cdots, l_{h-1}\}$. Note that we have assumed a finite and discrete label set. Although this is not generally true, many applications use, or can be approximated using, a discrete set of labels. In this work, we will mainly focus on optimization based approaches for such applications.

A particular labelling of the random variables $\mathbf{v}$ can be represented by a function f whose domain corresponds to the indices of the random variables and range corresponds to the indices of the label set, i.e.

$$f : \{0, 1, \cdots, n-1\} \rightarrow \{0, 1, \cdots, h-1\}. \tag{2.1.1}$$

In other words, a labelling specifies that a random variable $v_a \in \mathbf{v}$ takes a label $l_{f(a)} \in \mathbf{l}$. In general we wish to obtain a labelling (or a set of labellings) which satisfies some criterion (e.g. it maximizes the joint probability of the labelling and the data).

The ability of the labelling problem to provide a unified framework for several applications makes it of fundamental importance to the community. In order to illustrate this, let us formulate some applications as labelling problems. We begin by considering the task of detecting an instance of an object class of interest (e.g. cows) in a given image $\mathbf{D}$. The object class is modelled using a set of parts. For example, a cow can be modelled using ten parts (the head, the torso and the eight half-limbs) as shown in Fig. 2.1(a). In order to model object detection as a labelling problem, we define the random variables $\mathbf{v}$ and the label set $\mathbf{l}$ as follows.

- Each random variable $v_a \in \mathbf{v}$ corresponds to one part of the object class model (e.g. the head or the torso).
- Each label $l_i = \{x_i, y_i, \phi_i, \rho_i\} \in \mathbf{l}$ represents a putative pose of the part in the image $\mathbf{D}$, where (x_i, y_i) is the location of the part in the image, ϕ_i is the orientation of the part and ρ_i is its scale.

Clearly, any labelling f defined using the above random variables and labels provides a localization of the object in the given image. The problem of object detection can then be viewed as obtaining a *desirable* labelling. Broadly speaking, a desirable labelling f should satisfy the following conditions:

- The appearance of an object part v_a should conform with the image data at pose $l_{f(a)}$.

(a) (b)

Figure 2.1: **(a)** *An example parts-based model for the object class 'cow'. The outlines show the shape of each part. The solid black lines show their orientation. The length of these black lines indicates the scale of each part. The solid grey lines which connect different parts indicate that the location of these parts should form a valid configuration.* **(b)** *Using the model shown in (a), the object is localized in a given image* $\mathbf{D}$*. The detected parts are shown overlaid on the image.*

- The boundary of an object part v_a in pose $l_{f(a)}$ should lie on the edges of the image.
- The parts form a valid spatial configuration (e.g. the head and the torso of a cow are adjacent or the legs of the cow are located beneath the torso).

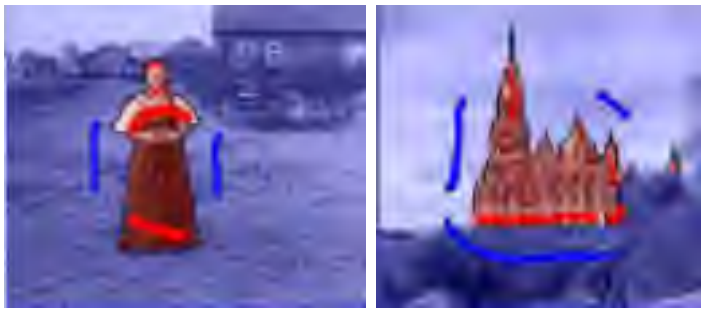
Fig. 2.1(b) shows a cow that has been detected in a given image by obtaining a desired labelling using the parts-based model in Fig. 2.1(a).

In the above application the number of random variables n is typically quite small (e.g. $n = 10$ for cows) while the number of labels h is large (e.g. the work described in [23] uses thousands of labels). In contrast our next example, namely interactive binary image segmentation [13], is defined using a large number of random variables and only 2 labels. In this application, the user provides a set of foreground and background seed pixels for a given image $\mathbf{D}$. These seed pixels can be used to learn a foreground and background appearance model. The aim of the application is then to partition the pixels of the image into foreground pixels and background pixels. This task can be formulated as a labelling problem over random variables $\mathbf{v}$ and label set $\mathbf{l}$ as follows.

- Each random variable $v_a \in \mathbf{l}$ corresponds to one pixel a of the given image $\mathbf{D}$.
- The label set consists of two labels, i.e. $\mathbf{l} = \{l_0, l_1\}$ where l_0 represents the foreground and l_1 represents the background.

A labelling f using the above random variables and labels specifies a segmentation of the image $\mathbf{D}$. Thus the task of interactive binary image segmentation can be seen as that of obtaining a desirable labelling, where a desirable labelling satisfies the following conditions.

- The labels $l_{f(a)}$ of variables v_a conform with the appearance models.



(a) (b)

Figure 2.2: *Two examples for interactive binary segmentation. The red and blue lines show the foreground and background seed pixels provided by the user. The blue region shows the pixels corresponding to the random variables which take the label l_1. The remaining pixels belong to the foreground. Image courtesy: Yuri Boykov.*

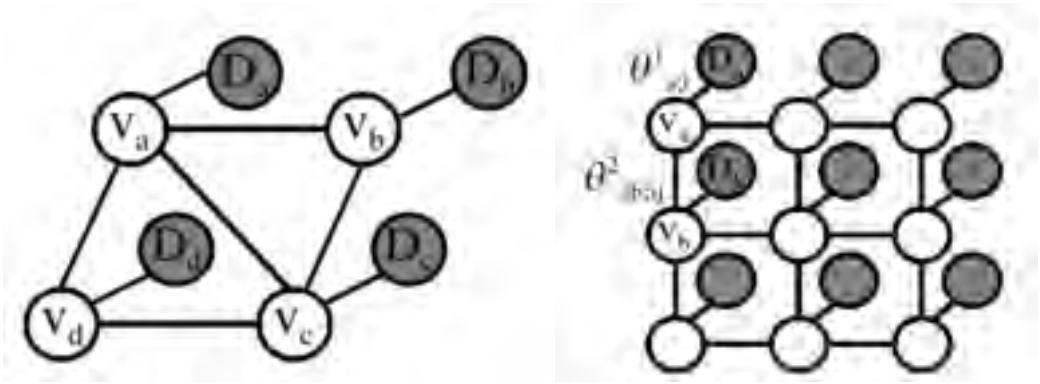
- The boundary of the resulting segmentation lies on image edges.
- The segmentation is smooth (i.e. it is spatially continuous and free of *speckles*).

Fig. 2.2 shows some examples of the image segmentations obtained by computing a desired labelling.

Note that in the above two applications, we have only provided a qualitative description of the desired labelling. While this is sufficient to show the importance of the labelling problem, it does not provide us with a concrete methodology to model Computer Vision applications. In order to address this issue, we now move on to a more quantitative description which specifies either (i) the joint probability of the labelling f and the data $\mathbf{D}$ (denoted by $\Pr(f, \mathbf{D})$); or (ii) the conditional probability of f given $\mathbf{D}$ (denoted by $\Pr(f|\mathbf{D})$). It is common practice to represent the above mentioned distributions using *probabilistic models* [7, 65]. In general, a particular distribution can be compactly represented using several different probabilistic models (e.g. *Bayesian networks* or *factor graphs*, see [114]). In this work, we use the *random field* models described in the next section.

2.2. Random Field Models

A random field model specifies a *neighbourhood* relationship between the variables $\mathbf{v}$. Let $\mathbf{N}_a \subset \{0, 1, \cdots, n-1\}$ represent the neighbourhood of a random variable v_a such that $b \in \mathbf{N}_a$ if, and only if, v_b is a neighbour of v_a. In this work, we are particularly interested in two types of random field models: (i) Markov random field; and (ii) conditional random field.



(a) (b)

Figure 2.3: *Examples of* MRF. *The unfilled circles denote the variables* $\mathbf{v}$ *while the filled circles represent the observed data* $\mathbf{D}$. **(a)** MRF *consisting of cliques of size two (formed by a variable and the corresponding observed node) and three (formed by three variables* $\{v_a, v_c, v_d\}$ *or* $\{v_a, v_b, v_d\}$*).* **(b)** *A pairwise* MRF *(i.e. with cliques of size two). The pairwise potentials are associated with the connections between neighbouring variables* v_a *and* v_b *while the unary potentials correspond to the connections between the variable* v_a *and the corresponding observed data* $\mathbf{D}_a$.

2.2.1 Markov Random Fields

A Markov random field (MRF) models the joint probability $\Pr(f, \mathbf{D})$ of the labelling f and the data $\mathbf{D}$. This joint probability is equal to the product of the likelihood and the prior probabilities corresponding to the labelling f, i.e.

$$\Pr(f, \mathbf{D}) = \Pr(\mathbf{D}|f) \Pr(f). \tag{2.2.1}$$

A random field modelling the above joint distribution is said to be *Markovian* if the prior distribution $\Pr(f)$ satisfies the following constraint [41]:

$$\Pr(f(a)|f(\{0, 1, \cdots, n-1\} - \{a\})) = \Pr(f(a)|f(\mathbf{N}_a)), \forall a, \tag{2.2.2}$$

where $f(\mathbf{S})$ for any set $\mathbf{S}$ is defined as $f(\mathbf{S}) = \{f(b)|b \in \mathbf{S}\}$. In other words, the prior probability of assigning a label to v_a depends only on the labelling of its neighbours defined by $\mathbf{N}_a$.

The Hammersley and Clifford theorem [6] allows us to represent the joint distribution $\Pr(f, \mathbf{D})$ of an MRF in closed form as follows. Let $\mathcal{C}$ be the set of cliques formed by the neighbourhood relationship of the MRF. Associated with each clique $\mathbf{C} \in \mathcal{C}$ is a potential $\theta_{\mathbf{C};f(\mathbf{C})}$ such that

$$\Pr(f, \mathbf{D}|\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \prod_{\mathbf{C} \in \mathcal{C}} \exp(-\theta_{\mathbf{C};f(\mathbf{C})}), \tag{2.2.3}$$

where for completeness we have introduced the term $\boldsymbol{\theta}$ in the LHS which denotes the parameters of the MRF (i.e. the clique potentials $\theta_{\mathbf{C};f(\mathbf{C})}$). The term $Z(\boldsymbol{\theta})$ is

a normalization constant (called the partition function) which ensures that the probabilities sum to one, i.e.

$$Z(\boldsymbol{\theta}) = \sum_{f} \prod_{\mathbf{C} \in \mathcal{C}} \exp(-\theta_{\mathbf{C};f(\mathbf{C})}). \tag{2.2.4}$$

The graphical model representation [7, 65] of an MRF is defined by undirected edges between two types of nodes: (i) observed nodes $\mathbf{D}_a$ which represent the data and are denoted by filled circles; and (ii) hidden nodes v_a, i.e. the random variables, which are denoted by unfilled circles. The undirected edges can also be classified into two types: (i) those that connect each random variable v_a to its corresponding observed node $\mathbf{D}_a$, thereby forming a clique of size 2; and (ii) those that connected v_a with its neighbours $\mathbf{N}_a$, thereby forming cliques of arbitrary size. Fig. 2.3 shows two examples of graphical models of MRF.

In this work, we are particularly interested in *pairwise* MRF (also known as *second-order* MRF[1]), i.e. MRF with cliques of size two. Although this seems restrictive, it can be shown that all probabilistic models which represent the joint distribution $\Pr(f, \mathbf{D})$ (including MRFs with arbitrary size cliques) have an equivalent pairwise MRF formulation [114].

In order to represent a pairwise MRF, we define a set $\mathcal{E}$ such that $(a, b) \in \mathcal{E}$ if, and only if, $b \in \mathbf{N}_a$, i.e. $\mathcal{E}$ consists of the set of all pairs of neighbouring variables. The distribution specified by a pairwise MRF defined on n random variables is then given by

$$\Pr(f, \mathbf{D}|\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \prod_{a} \exp(-\theta^1_{a;f(a)}) \prod_{(a,b)\in\mathcal{E}} \exp(-\theta^2_{ab;f(a)f(b)}), \tag{2.2.5}$$

where $\theta^1_{a;f(a)}$ and $\theta^2_{ab;f(a)f(b)}$ are called unary and pairwise potentials respectively. The superscripts 1 and 2 indicate that the unary potential depends on the label of one random variable at a time while the pairwise potential depends on the labels of two neighbouring random variables. The energy of a labelling f with respect to the MRF is defined as

$$Q(f, \mathbf{D}; \boldsymbol{\theta}) = \sum_{a} \theta^1_{a;f(a)} + \sum_{(a,b)\in\mathcal{E}} \theta^2_{ab;f(a)f(b)}. \tag{2.2.6}$$

Clearly the probability of a labelling can be written in terms of its energy as

$$\Pr(f, \mathbf{D}|\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \exp(-Q(f, \mathbf{D}; \boldsymbol{\theta})). \tag{2.2.7}$$

Fig. 2.3(b) shows an example graphical model of a pairwise MRF. Note that the terms $\theta^1_{a;f(a)}$ define the potentials for the cliques formed by an observed node

[1] Besag [6] used order for specifying the neighbourhood. However, using order for clique size is more appropriate due to the relationship of random fields with the work on Pseudo-Boolean Optimization [11] where clique size is related to the order of the pseudo-boolean polynomial than represents the MRF.

and a hidden node, i.e. they depend on the data. In contrast, $\theta^2_{ab;f(a)f(b)}$ are the potentials for cliques formed by joining two neighbouring hidden nodes and hence, they do not depend on the data.

Many applications in Computer Vision can be modelled using the MRF framework. In order to illustrate this, let us consider the two examples discussed in the previous section.

Example 1: For object class detection using a parts-based model, the neighbourhood relationship $\mathcal{E}$ is usually chosen to define a tree structured MRF[2] as in [23] or a complete graph[3] as in [25]. The unary potential $\theta^1_{a;f(a)}$ represents the negative log likelihood of the part v_a being assigned a label $l_{f(a)}$. For example, the unary potential can be measured as the chamfer distance between the shape exemplar of the part and the edge map of the image at pose $l_{f(a)}$. The pairwise potentials can be described using a non-regular Potts model[4] as follows:

$$\theta^2_{ab;f(a)f(b)} = \begin{cases} \kappa_1 & \text{if } l_{f(a)} \text{ and } l_{f(b)} \text{ form valid configuration,} \\ \kappa_2 & \text{otherwise,} \end{cases} \tag{2.2.8}$$

where $(a, b) \in \mathcal{E}$ and $\kappa_1 < \kappa_2$. From equation (2.2.5) we see that the pairwise potentials favour a labelling which specifies the poses of the parts to form a valid configuration (since this reduces the energy, thereby increasing the joint probability $\Pr(f, \mathbf{D}|\boldsymbol{\theta})$). Valid configurations can be defined strictly (e.g. distance between head and torso should be between 100 and 120 pixels) or leniently (e.g. head and torso should be adjacent to each other). Note that while the unary potentials depend on the observed data $\mathbf{D}$, the pairwise potentials are defined independent of the data. The above MRF model provides the probability of the object at any pose using equation (2.2.5).

Example 2: In the case of interactive binary image segmentation, the neighbourhood is usually specified to form a grid MRF (i.e. $\mathcal{E}$ defines a 4 or 8 neighbourhood). The unary potential $\theta^1_{a;f(a)}$ of a variable v_a taking a label $l_{f(a)}$ is the negative log likelihood of the RGB value $\mathbf{D}_a$ under the appearance models learnt from the seed pixels. For example, consider the image shown in Fig. 2.4(a) where the filled rectangles indicate the seed pixels provided by the user. If the foreground and background appearance models are represented by RGB histograms

[2]The neighbourhood relationship $\mathcal{E}$ is said to define a tree structured random field if the hidden nodes in the graphical model form a connected, acyclic graph.

[3]The neighbourhood relationship $\mathcal{E}$ is said to define a complete graph MRF if $(a, b) \in \mathcal{E}$ for all $v_a, v_b \in \mathbf{v}$, $a \neq b$.

[4]Note that in the Potts model the value of the pairwise potential $\theta^2_{ab;f(a)f(b)}$ is κ_1 if, and only if, $l_{f(a)} = l_{f(b)}$ and κ_2 otherwise. We use the term non-regular Potts model to mean any pairwise potential function defined over random variables v_a and v_b and $|\mathbf{l}| > 2$ labels which can take only one of two values depending on the labels $l_{f(a)}$ and $l_{f(b)}$.

(a) (b) (c)

Figure 2.4: **(a)** *An example image. The red and blue rectangles show the foreground and background seed pixels respectively which are provided by the user.* **(b)** *The ratio* $\frac{\theta_{a;1}}{\theta_{a;0}}$ *for each variable* v_a*. Brighter pixels indicate variables for which* $\theta_{a;0} < \theta_{a;1}$*, i.e. variables which are more likely to belong to the foreground.* **(c)** *The ratio (2.2.18) for each variable* v_a *specified by the pairwise potentials of the* CRF *formulation. The brighter pixels, which correspond to image edges, indicate the variables which are more likely to belong to the boundaries of the segmentation.*

($\mathcal{H}_f$ for foreground and $\mathcal{H}_b$ for background), then the ratio of the unary potentials

$$\frac{\theta^1_{a;1}}{\theta^1_{a;0}} = \frac{-\log p(\mathbf{D}_a|\mathbf{H}_b)}{-\log p(\mathbf{D}_a|\mathbf{H}_f)}, \tag{2.2.9}$$

for all variables v_a is shown in Fig. 2.4(b). Note that the brighter pixels indicate the random variables that are more likely to be assigned to the foreground. In order to encourage continuous segmentation, the pairwise potentials can be defined using an Ising model[5] as

$$\theta^2_{ab;f(a)f(b)} = \begin{cases} \kappa_1 & \text{if} \quad f(a) = f(b), \\ \kappa_2 & \text{otherwise,} \end{cases} \tag{2.2.10}$$

where $(a, b) \in \mathcal{E}$ and $\kappa_1 < \kappa_2$. In other words, the data independent pairwise potentials encourage two neighbouring variables v_a and v_b to take the same label.

2.2.2 Conditional Random Fields

Often in Computer Vision applications we would like to use pairwise potentials which are data dependent. For example, in interactive binary segmentation, if two neighbouring pixels a and b differ greatly in their RGB values then they are likely to belong to different segments. Hence, the energy function should not be heavily penalized for assigning them different labels. This can be achieved by defining the value of the pairwise potential $\theta^2_{ab;f(a)f(b)}$, where $f(a) \neq f(b)$, which

[5]The Ising model is a special case of the Potts model which is defined using $|\mathbf{l}| = 2$ labels such that $\theta^2_{ab;ii} < \theta^2_{ab;ij}$ where $i \neq j$.

depends on the difference between the RGB values $\mathbf{D}_a$ and $\mathbf{D}_b$ (i.e. the pairwise potentials are data dependent).

A probabilistic model which allows us to use data dependent pairwise potentials is the *conditional random field* (CRF) [56]. CRF is a discriminative framework (unlike the generative MRF framework [7]) which models the conditional probability $\Pr(f|\mathbf{D})$. The conditional distribution is assumed to satisfy the Markovian property, i.e.

$$\Pr(f(a)|f(\{0,1,\cdots,n-1\}-\{a\}),\mathbf{D}) = \Pr(f(a)|f(\mathbf{N}_a),\mathbf{D}). \tag{2.2.11}$$

Once again, we are mainly interested in the pairwise CRF model. According to the Hammersley-Clifford theorem [6], the distribution $\Pr(f|\mathbf{D})$ specified by a pairwise CRF can be written in closed form as

$$\Pr(f|\mathbf{D},\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})}\prod_a \exp(-\theta^1_{a;f(a)}) \prod_{(a,b)\in\mathcal{E}} \exp(-\theta^2_{ab;f(a)f(b)}), \tag{2.2.12}$$

where $\theta^1_{a;f(a)}$ and $\theta^2_{ab;f(a)f(b)}$ are the unary and pairwise potentials respectively which depend on the data. The partition function $Z(\boldsymbol{\theta})$ ensures that the probability distribution is normalized, i.e.

$$Z(\boldsymbol{\theta}) = \sum_f \prod_a \exp(-\theta^1_{a;f(a)}) \prod_{(a,b)\in\mathcal{E}} \exp(-\theta^2_{ab;f(a)f(b)}). \tag{2.2.13}$$

The parameter $\boldsymbol{\theta}$ of the CRF has been introduced in the LHS for completeness. Note that we have used similar notation for both MRF and CRF. However, the nature of the random field would be clear from context.

The energy of a labelling f for given CRF (denoted by $Q(f;\mathbf{D})$) is defined as

$$Q(f;\mathbf{D},\boldsymbol{\theta}) = \sum_a \theta^1_{a;f(a)} + \sum_{(a,b)\in\mathcal{E}} \theta^2_{ab;f(a)f(b)}. \tag{2.2.14}$$

Similar to MRF, the posterior probability of a labelling can be written in terms of its energy as

$$\Pr(f|\mathbf{D},\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})}\exp(-Q(f;\mathbf{D},\boldsymbol{\theta})). \tag{2.2.15}$$

Example: In order to illustrate the advantage of the CRF model over MRF, let us return to the case of interactive binary segmentation. Since the pairwise potentials are now data dependent, we can define them as follows:

$$\theta^2_{ab;f(a)f(b)} = \begin{cases} \kappa_1 & \text{if} \quad f(a)=f(b), \\ \kappa_2 - \gamma(a,b) & \text{otherwise,} \end{cases} \tag{2.2.16}$$

where $\kappa_1 < \kappa_2$ and the function $\gamma(\cdot,\cdot)$ is chosen to encourage the boundary of the segments to lie on image edges. Following [13], we choose

$$\gamma(a,b) = \lambda\left(1-\exp\left(\frac{-\Delta^2(a,b)}{2\sigma^2}\right)\frac{1}{dist(a,b)}\right). \tag{2.2.17}$$

Here, $\Delta(a,b)$ measures the difference in the RGB values $\mathbf{D}_a$ and $\mathbf{D}_b$ and $dist(a,b)$ is the Euclidean distance between pixels a and b. The parameter λ is the relative weight given to the term $\gamma(a,b)$ compared to the term κ_2. The parameter σ determines how much is subtracted from the pairwise potential $\theta^2_{ab;f(a)f(b)}$ when $f(a) \neq f(b)$, as $\gamma(a,b)$ is small when $\Delta(a,b) < \sigma$ and large when $\Delta(a,b) > \sigma$. In order to obtain good segmentations, σ should be sufficiently large to allow for variation in the RGB values within a segment. Fig. 2.4(c) shows the ratio

$$\frac{\sum_{b,(a,b)\in\mathcal{E}} \theta^2_{ab;00} + \theta^2_{ab;11}}{\sum_{b,(a,b)\in\mathcal{E}} \theta^2_{ab;01} + \theta^2_{ab;10}} \tag{2.2.18}$$

for all variables v_a using the following parameter setting: $\kappa_1 = 1$, $\kappa_2 = 2.2$, $\lambda = 1$ and $\sigma = 5$. Note that since $\theta^2_{ab;f(a)f(b)}$ is a constant when $f(a) = f(b)$ for all neighbouring variables v_a and v_b, the above ratio would be large for a variable v_a when $\theta^2_{ab;f(a)f(b)}$, $f(a) \neq f(b)$, has a small value for all $(a,b) \in \mathcal{E}$. In other words the brighter pixels in Fig. 2.4(c), which correspond to image edges, indicate the variables v_a which are more likely to lie on the segmentation boundary. Clearly, the pairwise potentials of CRF (in Fig. 2.4(c)) are more informative for the task of interactive binary image segmentation than the data independent pairwise potentials specified by the MRF formulation.

2.3. Inference

Models such as an MRF or a CRF can be used when performing *inference*, i.e. providing answers to probabilistic queries. We now describe three types of inference problems using a toy example for illustration. We assume the probabilistic model to be a CRF while noting that similar problems can be defined for the MRF framework.

2.3.1 Maximum A Posteriori Estimation

The problem of maximum a posteriori (MAP) estimation is to find the labelling with the maximum posterior probability. Formally, it requires us to find the labelling $\mathbf{f}^*$ such that

$$\begin{aligned} f^* &= \arg\max_f \Pr(f|\mathbf{D},\boldsymbol{\theta}) \\ &= \arg\max_f \frac{1}{Z(\boldsymbol{\theta})} \prod_a \exp(-\theta^1_{a;f(a)}) \prod_{(a,b)\in\mathcal{E}} \exp(-\theta^2_{ab;f(a)f(b)}). \end{aligned} \tag{2.3.1}$$

Note that in the above equation $Z(\boldsymbol{\theta})$ is a constant and can be ignored during the maximization. This implies that the MAP labelling can be obtained by minimizing

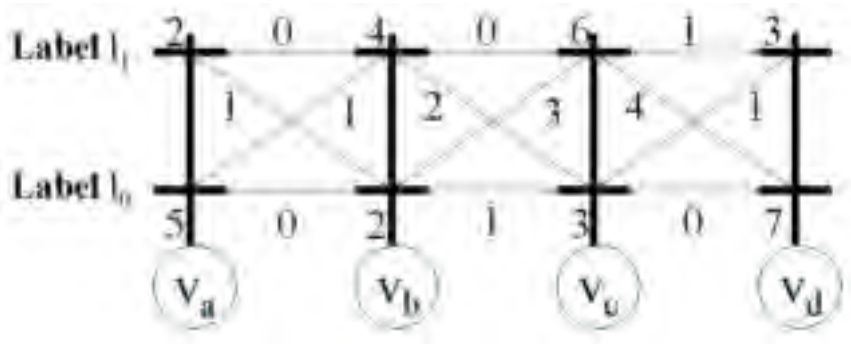


Figure 2.5: *An example* CRF *consisting of four variables (shown as circles) each of which can take one of two possible labels* l_0 *and* l_1 *(represented as a trellis on top of the variables). The unary and pairwise potentials corresponding to the* CRF *are also shown. For example,* $\theta^1_{a;0} = 5$ *and* $\theta^2_{bc;01} = 3$.

$f(a)$	$f(b)$	$f(c)$	$f(d)$	Energy	Probability
0	0	0	0	18	0.0048
0	0	0	1	15	0.0971
0	0	1	0	27	5.961e-07
0	0	1	1	20	0.0006
0	1	0	0	22	8.847e-05
0	1	0	1	19	0.0018
0	1	1	0	27	5.961e-07
0	1	1	1	20	0.0006
1	0	0	0	16	0.0357
1	0	0	1	**13**	**0.7169**
1	0	1	0	25	4.405e-06
1	0	1	1	18	0.0048
1	1	0	0	18	0.0048
1	1	0	1	15	0.0971
1	1	1	0	23	3.254e-05
1	1	1	1	16	0.0357

Table 2.1: *Energies and probabilities of all 16 possible labellings of the* CRF *shown in Fig. 2.5. The energies are computed using equation (2.2.14) and the probabilities are found using equation (2.2.15). The partition function* $Z(\boldsymbol{\theta})$ *required in equation (2.2.15) is computed using equation (2.2.13). For this* CRF *parameter* $\boldsymbol{\theta}$, $Z(\boldsymbol{\theta}) = 3.153e-06$.

the energy function $Q(\cdot;\mathbf{D})$, i.e.

$$\begin{aligned} f^* &= \arg\min_f Q(f;\mathbf{D},\boldsymbol{\theta}) \\ &= \arg\min_f \sum_a \theta^1_{a;f(a)} + \sum_{(a,b)\in\mathcal{E}} \theta^2_{ab;f(a)f(b)}. \end{aligned} \tag{2.3.2}$$

Fig. 2.5 shows an alternative graphical model representation of a CRF which consists of four random variables $\mathbf{v} = \{v_a, v_b, v_c, v_d\}$. Each of these random variables is represented by a hidden node (i.e. an unfilled circle) in the figure. The putative labels $\mathbf{l} = \{l_0, l_1\}$ for a random variable are shown as trellises (i.e. the vertical lines) on top of the corresponding hidden node. In other words, each branch of a trellis (shown as a horizontal line) represents one of the labels in the set $\mathbf{l}$. Note that the observed nodes are not shown for the sake of clarity of the figure. This representation allows us to compactly define the unary and pairwise potentials associated with the CRF. Specifically, the unary potential $\theta^1_{a;i}$ is shown next to the i^{th} branch of the trellis on top of the hidden node v_a. For example according to Fig. 2.5, $\theta^1_{a;0} = 5$ and $\theta^1_{a;1} = 2$. The pairwise potential $\theta^2_{ab;ij}$ is shown next to the edge joining the i^{th} and j^{th} branches of the trellises corresponding to hidden nodes v_a and v_b respectively. For example in Fig. 2.5, $\theta^2_{ab;00} = 0$ and $\theta^2_{ab;01} = 1$.

Table 2.1 lists the energies and the probabilities of all possible labellings corresponding to this CRF. The values of the energies and the probabilities are computed using equations (2.2.14) and (2.2.15) respectively. Consider the labelling $\{0,1,1,0\}$ of the CRF in Fig. 2.5. The energy of this labelling is given by

$$\begin{aligned} Q(\{0,1,1,0\}|\mathbf{D},\boldsymbol{\theta}) = &\ \theta^1_{1;0} + \theta^1_{2;1} + \theta^1_{3;1} + \theta^1_{4;0} + \theta^2_{12;01} + \theta^2_{23;11} + \theta^2_{34;10}, \\ = &\ 5+4+6+7+1+0+4 = 27. \end{aligned} \tag{2.3.3}$$

The corresponding probability is computed as

$$\Pr(\{0,1,1,0\}|\mathbf{D},\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})}\exp(-27) = 5.961e-07, \tag{2.3.4}$$

where $Z(\boldsymbol{\theta})$ is given by equation (2.2.13). Clearly the labelling $f^* = \{1,0,0,1\}$ is the MAP estimate since it has the highest posterior probability (or equivalently it has the lowest energy).

2.3.2 Computing Max-Marginals

Consider a labelling f which assigns a particular label, say l_i, to a random variable v_a. The *max-marginal probability*[1] of this assignment is defined as the maximum posterior probability over all such labellings f (i.e. over all f which satisfy

[1]Max-marginal probabilities are called so since their form is similar to the marginal probabilities defined in the next section. We note, however, that max-marginal probabilities are not obtained using marginalization.

Label	Min-marginal energy	Max-marginal probability
$f(a) = 0$	15	0.0971
$f(a) = 1$	13	0.7169
$f(b) = 0$	13	0.7169
$f(b) = 1$	15	0.0971
$f(c) = 0$	13	0.7169
$f(c) = 1$	16	0.0357
$f(d) = 0$	16	0.0357
$f(d) = 1$	13	0.7169

Table 2.2: *Min-marginal energies and max-marginal probabilities for the* CRF *shown in Fig. 2.5 computed using equations (2.3.8) and (2.3.5) respectively.*

$f(a) = i$). We denote the max-marginal probability of assigning label l_i to v_a as $p_{a;i}(\mathbf{D}, \boldsymbol{\theta})$. The lower case 'p' is used to indicate the max-marginal probability is not technically a probability distribution as it is not necessarily normalized. In other words, for all particular variable v_a its max-marginal probabilities might not sum to one. Formally, $p_{a;i}(\mathbf{D}, \boldsymbol{\theta})$ is given by the following equation:

$$\begin{aligned} p_{a;i}(\mathbf{D}, \boldsymbol{\theta}) &= \max_{f, f(a)=i} \Pr(f|\mathbf{D}, \boldsymbol{\theta}) \\ &= \frac{1}{Z(\boldsymbol{\theta})} \max_{f, f(a)=i} \prod_{c} \exp(-\theta^1_{c;f(c)}) \prod_{(c,d)\in\mathcal{E}} \exp(-\theta^2_{cd;f(c)f(d)}). \end{aligned} \tag{2.3.5}$$

For example, the max-marginal $p_{a;1}(\mathbf{D}, \boldsymbol{\theta})$ for the CRF shown in Fig. 2.5 is the maximum of the probabilities of the last eight labellings in table 2.1, i.e.

$$\begin{aligned} p_{a;1}(\mathbf{D}, \boldsymbol{\theta}) = &\max\{0.0357, 0.7169, 4.405e-06, 0.0048, \\ &\quad 0.0048, 0.0971, 3.234e-05, 0.0357\} \\ = &\quad 0.7169. \end{aligned} \tag{2.3.6}$$

One can also define the max-marginal probability of assigning labels l_i and l_j to a pair of neighbouring random variables v_a and v_b. We denote this max-marginal probability as $p_{ab;ij}(\mathbf{D}, \boldsymbol{\theta})$. Once again, the max-marginal probabilities of random variables v_a and v_b need not be normalized. Similar to equation (2.3.5), the max-marginal probability $p_{ab;ij}(\mathbf{D}, \boldsymbol{\theta})$ is given by

$$\begin{aligned} p_{ab;ij}(\mathbf{D}, \boldsymbol{\theta}) &= \max_{f, f(a)=i, f(b)=j} \Pr(f|\mathbf{D}, \boldsymbol{\theta}) \\ &= \frac{1}{Z(\boldsymbol{\theta})} \max_{f, f(a)=i, f(b)=j} \prod_{c} \exp(-\theta^1_{c;f(c)}) \prod_{(c,d)\in\mathcal{E}} \exp(-\theta^2_{cd;f(c)f(d)}). \end{aligned} \tag{2.3.7}$$

In order to make the notation simpler, we will also denote the max-marginals $p_{a;i}(\mathbf{D}, \boldsymbol{\theta})$ and $p_{ab;ij}(\mathbf{D}, \boldsymbol{\theta})$ as $p_{a;i}$ and $p_{ab;ij}$ respectively whenever it is clear which CRF parameter $\boldsymbol{\theta}$ and data $\mathbf{D}$ are being used.

For every max-marginal probability, we can also define a min-marginal energy (or simply min-marginal) as

$$\begin{aligned} q_{a;i}(\mathbf{D},\boldsymbol{\theta}) &= \min_{f,f(a)=l_i} Q(f;\mathbf{D},\boldsymbol{\theta}) = \min_{f,f(a)=i} \sum_c \theta^1_{c;f(c)} + \sum_{(c,d)\in\mathcal{E}} \theta^2_{cd;f(c)f(d)}, \\ q_{ab;ij}(\mathbf{D},\boldsymbol{\theta}) &= \min_{f,f(a)=l_i,f(b)=l_j} Q(f;\mathbf{D},\boldsymbol{\theta}) \\ &= \min_{f,f(a)=i,f(b)=j} \sum_c \theta^1_{c;f(c)} + \sum_{(c,d)\in\mathcal{E}} \theta^2_{cd;f(c)f(d)}. \end{aligned} \tag{2.3.8}$$

Note that we use lower case 'q' so that the notation corresponds to the lower case 'p' used for max-marginal probabilities. For example, the min-marginal $q_{a;1}(\mathbf{D},\boldsymbol{\theta})$ for the CRF shown in Fig. 2.5 is

$$q_{a;1}(\mathbf{D},\boldsymbol{\theta}) = \min\{16, 13, 25, 18, 18, 15, 23, 16\} = 13. \tag{2.3.9}$$

Similar to max-marginals, we will also denote the min-marginal energies $q_{a;i}(\mathbf{D},\boldsymbol{\theta})$ and $q_{ab;ij}(\mathbf{D},\boldsymbol{\theta})$ as $q_{a;i}$ and $q_{ab;ij}$ respectively whenever it is clear which CRF and data are being used. Table 2.2 lists all the min-marginal energies and max-marginal probabilities corresponding to the variables of the CRF shown in Fig. 2.5.

Max-marginals (and equivalently min-marginals) are closely related to the problem of MAP estimation. Specifically, it can be shown that if a labelling f satisfies the following:

$$p_{a;f(a)}(\mathbf{D},\boldsymbol{\theta}) \geq p_{a;i}(\mathbf{D},\boldsymbol{\theta}), \forall v_a \in \mathbf{v}, l_i \in \mathbf{l}, \tag{2.3.10}$$

$$p_{ab;f(a)f(b)}(\mathbf{D},\boldsymbol{\theta}) \geq p_{ab;ij}(\mathbf{D},\boldsymbol{\theta}), \forall (a,b) \in \mathcal{E}, l_i, l_j \in \mathbf{l}, \tag{2.3.11}$$

then f is the MAP labelling of the CRF. In other words, the MAP estimate of a CRF can be obtained by choosing a labelling f which provides the maximum max-marginal (or equivalently the minimum min-marginals) for all $v_a \in \mathbf{v}$ and $(a,b) \in \mathcal{E}$. Further, max-marginals and min-marginals can also be used to compute the M most probable configurations of a CRF [113].

2.3.3 Computing Marginals

In many applications, one might wish to compute the *marginal probabilities*. Similar to max-marginal probabilities, these are defined for a particular assignment (say label l_i) of a random variable v_a. The marginal probability (or simply marginal) $P_{a;i}(\mathbf{D},\boldsymbol{\theta})$ of such an assignment is the sum of the posterior probabilities of all labelling f such that $f(a) = i$. Formally, the marginals are given by the following equation:

$$\begin{aligned} P_{a;i}(\mathbf{D},\boldsymbol{\theta}) &= \sum_{f,f(a)=i} \Pr(f|\mathbf{D},\boldsymbol{\theta}) \\ &= \frac{1}{Z(\boldsymbol{\theta})} \sum_{f,f(a)=i} \left(\prod_c \exp(-\theta^1_{c;f(c)}) \prod_{(c,d)\in\mathcal{E}} \exp(-\theta^2_{cd;f(c)f(d)}) \right). \end{aligned} \tag{2.3.12}$$

Label	Marginals
$f(a) = 0$	0.10502
$f(a) = 1$	0.89498
$f(b) = 0$	0.85991
$f(b) = 1$	0.14009
$f(c) = 0$	0.95814
$f(c) = 1$	0.04186
$f(d) = 0$	0.04548
$f(d) = 1$	0.95452

Table 2.3: *Marginals for the* CRF *shown in Fig. 2.5.*

For example, the marginal probability $P_{a;i}(\mathbf{D}, \boldsymbol{\theta})$ for the CRF shown in Fig. 2.5 can be computed as the sum of the probabilities of the last eight labellings in table 2.1, i.e.

$$\begin{aligned} P_{a;1}(\mathbf{D}, \boldsymbol{\theta}) &= 0.0357 + 0.7169 + 4.405e-06 + 0.0048 \\ &+ 0.0048 + 0.0971 + 3.234e-05 + 0.0357 \\ &= 0.89498. \end{aligned} \tag{2.3.13}$$

It can easily be verified that the marginals of a particular variable v_a are normalized. We choose to denote the marginal using upper case 'P' as they define true probability distributions. One can also define the marginal probability of a pair of neighbouring random variables v_a and v_b being assigned labels l_i and l_j respectively as

$$\begin{aligned} P_{ab;ij}(\mathbf{D}, \boldsymbol{\theta}) &= \sum_{f, f(a)=i, f(b)=j} \Pr(f|\mathbf{D}, \boldsymbol{\theta}) \qquad (2.3.14) \\ &= \frac{1}{Z(\boldsymbol{\theta})} \sum_{f, f(a)=i, f(b)=j} \left(\prod_{c} \exp(-\theta^1_{c;f(c)}) \prod_{(c,d)\in\mathcal{E}} \exp(-\theta^2_{cd;f(c)f(d)}) \right). \end{aligned}$$

Note that, unlike MAP estimation, ignoring the partition function $Z(\boldsymbol{\theta})$ would only compute the marginals up to a common scale factor. Table 2.3 lists all the marginal probabilities (computed using equation (2.3.13)) corresponding to the variables of the CRF in Fig. 2.5. In order to make the notation simpler, we will sometimes refer to the marginals $P_{a;i}(\mathbf{D}, \boldsymbol{\theta})$ and $P_{ab;ij}(\mathbf{D}, \boldsymbol{\theta})$ as $P_{a;i}$ and $P_{ab;ij}$ respectively when there is no ambiguity regarding which parameter $\boldsymbol{\theta}$ and data $\mathbf{D}$ is being employed.

Marginals are very useful when one wishes to *sample* from the distribution $\Pr(f|\mathbf{D})$. For example, see [23] for obtaining samples from the distribution of a tree structured MRF. Further, it can be shown that the labelling f^* where each label is given by

$$f^*(a) = \arg\max_{l_i \in \mathbf{l}} P_{a;i}(\mathbf{D}, \boldsymbol{\theta}) \tag{2.3.15}$$

provides us with the minimum mean squared error (MMSE) labelling [91].

Although the MAP estimate and the marginals can be easily computed for CRFs with small number of variables and labels (e.g. the one shown in Fig. 2.5), these inference problems are generally NP-hard[2]. However, given the central importance of probabilistic models in many areas of Computer Science, several approximate algorithms have been proposed in the literature. Next, we briefly describe some such approaches starting with algorithms for MAP estimation.

2.4. Algorithms for MAP Estimation

The problem of MAP estimation for a given pairwise MRF/CRF is central to obtaining the solution for many Computer Vision applications, e.g. interactive binary image segmentation, stereo reconstruction and image denoising [89]. Furthermore, MAP estimation is closely related to several important Combinatorial Optimization problems such as MAXCUT [31], multi-way cut [21], metric labelling [15, 42] and 0-extension [15, 39].

Although the MAP estimation problem is NP-hard in general, it is well-known that it can be solved exactly in polynomial time for certain restricted cases. For example, two such cases which are of interest to us are (i) tree structured MRF/CRF; and (ii) pairwise MRF/CRF with *submodular* energy functions. While the former case can be handled by a dynamic programming algorithm proposed in [70], the latter case has been shown to be equivalent to the *st*-MINCUT problem (which has several polynomial time algorithms) [33, 48, 78]. It seems natural, therefore, to extend these methods to obtain approximate MAP labellings for a general random field.

In this section, we will describe the approaches of [70] (for tree structured random fields) and [48] (for submodular energy functions), and explore some of their extensions that have been proposed in the literature. In the next chapter, we present an alternative formulation of MAP estimation which allows us to obtain approximate MAP labellings using *Convex Optimization*.

2.4.1 Min-Sum Belief Propagation

Min-sum belief propagation (min-sum BP) is an iterative *message passing* algorithm proposed by Pearl [70] to solve the MAP estimation problem. At each iteration, every random variable v_a passes messages to its neighbouring random variables (one message per neighbour). The message that v_a passes to its neighbour v_b (denoted by $\mathbf{m}_{ab}$) is a vector of length $h = |\mathbf{l}|$ whose j^{th} element is given

[2] NP-hard stands for *Non-deterministic Polynomial-time hard* and refers to a set of problems for which no polynomial time algorithms are known. An approximate solution to these problems is generally obtained by solving a related problem which has a polynomial time algorithm.

by

$$m_{ab;j} \longleftarrow \min_{l_i \in \mathbf{l}} \left(\theta^1_{a;i} + \theta^2_{ab;ij} + \sum_{(a,c)\in\mathcal{E}, c\neq b} m_{ca;i} \right) + \eta_1, \tag{2.4.1}$$

where η_1 is some constant which is used to prevent numerical overflow and underflow. As can be seen from the above equation, the messages are defined using the min and sum operations on the potentials and other messages. Hence, the name *min-sum* BP. All messages are usually initialized to 0. The algorithm is said to have converged when the rate of change of messages from one iteration to the next falls below a certain threshold.

Upon convergence, min-sum BP provides us with approximate min-marginals (called *beliefs*) for all random variables $(a, b) \in \mathcal{E}$ as

$$q'_{a;i} \longleftarrow \theta^1_{a;i} + \sum_{(a,c)\in\mathcal{E}} m_{ca;i} + \eta_2, \tag{2.4.2}$$

where $l_i \in \mathbf{l}$ and η_2 is some constant. It also provides us with beliefs for all pairs of random variables v_a and v_b, where $(a, b) \in \mathcal{E}$, as

$$q'_{ab;ij} \longleftarrow \theta^1_{a;i} + \theta^1_{ab;j} + \sum_{(a,c)\in\mathcal{E}, c\neq b} m_{ca;i} + \sum_{(a,c)\in\mathcal{E}, c\neq a} m_{cb;j} + \eta_2, \tag{2.4.3}$$

The notation q' indicates that the min-marginals obtained using min-sum BP need not necessarily be exact. The beliefs computed in this manner can be used to obtain an approximate MAP labelling f as

$$f(a) = \arg\min_{l_i \in \mathbf{l}} q'_{a;i}. \tag{2.4.4}$$

Min-sum BP was originally proposed for a tree structured random field where it is guaranteed to provide the exact min-marginals within two iterations [70]. Using the relationship between min-marginals and MAP estimation (discussed in section 2.3.2), it follows that min-sum BP also provides us the exact MAP labelling in two iterations. In the first iteration (which we call the forward pass), the messages are sent from the leaf nodes of the tree towards the root. In the second iteration (called the backward pass), the messages are sent in the opposite direction, i.e. starting from the root towards the leafs. Table 2.4 lists the messages and the beliefs computed by the min-sum BP algorithm for the tree structured CRF shown in Fig. 2.5 (assuming v_a to be the leaf and v_d to be the root). Note that the min-marginals correspond exactly with those listed in table 2.2. Further, the labelling $f = \{1, 0, 0, 1\}$ obtained using equation (2.4.4) provides the exact MAP estimate.

For a general random field (i.e. not necessarily tree structured), min-sum BP is not guaranteed to converge. However, in practice it often provides a good estimate of the MAP labelling, see e.g. [89]. Further, Weiss and Freeman [106] showed that upon convergence min-sum BP provides the optimum labelling within a single-cycle and tree neighbourhood. In other words, given the labelling f obtained

Forward Messages	Backward Messages
$m_{ab;0} \leftarrow 3$	$m_{dc;0} \leftarrow 4$
$m_{ab;1} \leftarrow 2$	$m_{dc;1} \leftarrow 4$
$m_{bc;0} \leftarrow 6$	$m_{cb;0} \leftarrow 8$
$m_{bc;1} \leftarrow 6$	$m_{cb;1} \leftarrow 9$
$m_{cd;0} \leftarrow 9$	$m_{ba;0} \leftarrow 10$
$m_{cd;1} \leftarrow 10$	$m_{ba;1} \leftarrow 11$

Label	Beliefs
$f(a) = 0$	$q'_{a;0} \leftarrow 15$
$f(a) = 1$	$q'_{a;1} \leftarrow 13$
$f(b) = 0$	$q'_{b;0} \leftarrow 13$
$f(b) = 1$	$q'_{b;1} \leftarrow 15$
$f(c) = 0$	$q'_{c;0} \leftarrow 13$
$f(c) = 1$	$q'_{c;1} \leftarrow 16$
$f(d) = 0$	$q'_{d;0} \leftarrow 16$
$f(d) = 1$	$q'_{d;1} \leftarrow 13$

Table 2.4: *Forward and backward pass messages computed using min-sum* BP *for the* CRF *shown in Fig. 2.5. The messages are computed using equation (2.4.1) with* $\eta_1 = 0$ *for both passes. The beliefs computed using equation (2.4.2) are also shown. Note that we use* $\eta_2 = 0$.

using min-sum BP, a labelling with higher probability (or lower energy) cannot be obtained by changing the labels of only those variables which form a tree or a single cycle.

2.4.2 Alternative Formulation of Min-Sum BP

We may sometimes find it convenient to use the alternative formulation of min-sum BP which was proposed in [46]. Before providing the details of this formulation, we must discuss the concept of *reparameterization*.

Reparameterization: Reparameterization (also known as equivalent transformations [79, 107]) plays a key role in many MAP estimation approaches (as will be seen in later sections). A parameter $\overline{\boldsymbol{\theta}}$ is called a reparameterization of $\boldsymbol{\theta}$ (denoted by $\overline{\boldsymbol{\theta}} \equiv \boldsymbol{\theta}$) if, and only if, the following holds true:

$$Q(f; \mathbf{D}, \overline{\boldsymbol{\theta}}) = Q(f; \mathbf{D}, \boldsymbol{\theta}), \forall f. \tag{2.4.5}$$

For example, it can easily be verified that the parameter $\overline{\boldsymbol{\theta}}$ given below is a reparameterization of $\boldsymbol{\theta}$:

$$\overline{\theta}^1_{a;i} = \theta^1_{a;i} + \textstyle\sum_{(a,b)\in\mathcal{E}} M_{ba;i}, \tag{2.4.6}$$

$$\overline{\theta}^2_{ab;ij} = \theta^2_{ab;ij} - M_{ba;i} - M_{ab;j}, \tag{2.4.7}$$

for all values of $M_{ba;i}$ and $M_{ab;j}$.

Min-Sum BP as Reparameterization: In the context of min-sum BP, the terms $M_{ba;i}$ and $M_{ab;j}$ which specify the reparameterization in equations (2.4.6) and (2.4.7) can be viewed as the i^{th} and the j^{th} elements of the messages $\mathbf{m}_{ba}$ and

$\mathbf{m}_{ab}$ respectively (i.e. $M_{ba;i} = m_{ba;i}$ and $M_{ab;j} = m_{ab;j}$). Formulated in this manner, min-sum BP boils down to a series of reparameterizations. In other words, instead of storing the original parameter $\boldsymbol{\theta}$ and the messages at each iteration, one can store the resulting reparameterized vector $\overline{\boldsymbol{\theta}}$ only. The algorithm is said to converge when the rate of change in the reparameterized vector $\overline{\boldsymbol{\theta}}$ falls below a certain threshold. The approximate MAP labelling f can then be obtained as

$$f(a) = \max_{l_i \in \mathbf{l}} \overline{\theta}^1_{a;i}. \tag{2.4.8}$$

The above formulation of min-sum BP retains the property of providing the exact MAP labelling for tree-structured random fields in two iterations. It is worth noting that at the end of the forward pass, the resulting reparameterized vector $\overline{\boldsymbol{\theta}}$ would provide the exact min-marginal for the root variable. In other words, if v_a is the root variable in a tree-structured random field, then

$$\overline{\theta}^1_{a;i} = q_{a;i}, \tag{2.4.9}$$

where $q_{a;i}$ is the exact min-marginal of v_a being assigned label l_i.

2.4.3 Graph Cuts

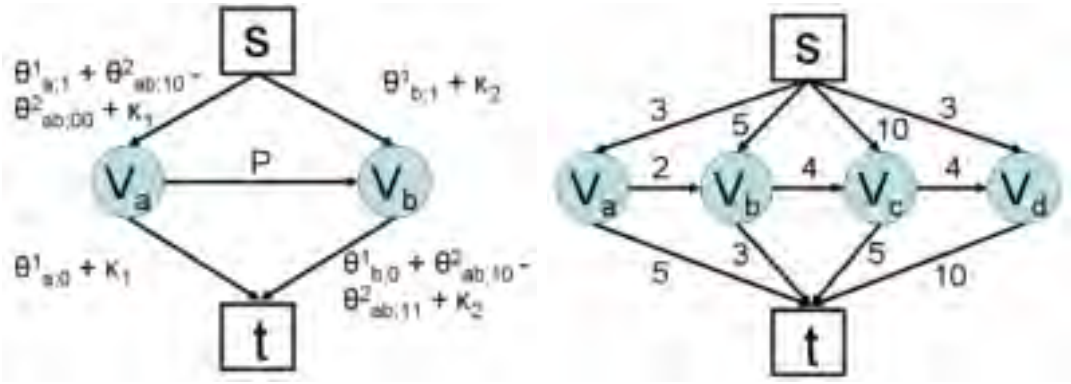
Algorithms based on graph cuts (more specifically st-MINCUT) are a popular tool for MAP estimation in Computer Vision due to their speed and accuracy. Below, we describe some such algorithms which are used in subsequent chapters. Before we begin, we need the following definitions.

*The st-*MINCUT *problem:* This problem is defined using a positively weighted directed graph $\mathcal{G} = \{\mathbf{V}_G \bigcup \{s, t\}, \mathbf{E}_G, w\}$. Here $\mathbf{V}_G$ denotes the set of vertices and $\mathbf{E}_G$ denotes the set of directed edges. The function $w : \mathbf{E}_G \rightarrow \mathbb{R}^+$ specifies the weights of the edges (where $\mathbb{R}^+$ denotes the set of non-negative real numbers)[1]. The vertices s and t are special vertices called the terminals such that there are no incoming edge to s (the source) and there are no outgoing edges from t (the sink). Given such a graph $\mathcal{G}$, a *cut* is defined as a partitioning of the vertices into two disjoint sets $\mathbf{V}^0_G$ and $\mathbf{V}^1_G$ such that $s \in \mathbf{V}^0_G$ and $t \in \mathbf{V}^1_G$. The cost of a cut (denoted by $C(\mathbf{V}^0_G, \mathbf{V}^1_G)$) is given by

$$C(\mathbf{V}^0_G, \mathbf{V}^1_G) = \sum_{V_a \in \mathbf{V}^0_G, V_b \in \mathbf{V}^1_G} w(V_a, V_b), \tag{2.4.10}$$

where $w(V_a, V_b)$ is the weight of edge from vertex V_a to V_b. The problem of st-MINCUT is to find the cut with the minimum cost. Several polynomial time algorithms exist for solving the st-MINCUT problem, e.g. see [14].

[1] To be precise, the st-MINCUT problem is defined for graphs whose weights are non-negative integers. However, since real numbers are quantized for storage in a computer, we will assume the weights to be non-negative real numbers.



(a) (b)

Figure 2.6: **(a)** *The graph construction for obtaining the* MAP *estimation of a random field defined on two variables* $\mathbf{v} = \{v_a, v_b\}$*. Here* $P = \theta^2_{ab;01} + \theta^2_{ab;10} - \theta^2_{ab;00} - \theta^2_{ab;11}$*. Note that if the submodularity condition (2.4.12) is satisfied, then* $P \geq 0$*. The terms* $\kappa_1 = \min\{0, |\theta^1_{a;0}|, |\theta^1_{a;1} + \theta^2_{ab;10} - \theta^2_{ab;00}|\}$ *and* $\kappa_2 = \min\{0, |\theta^1_{b;1}|, |\theta^1_{b;0} + \theta^2_{ab;10} - \theta^2_{ab;11}|\}$ *ensure that the remaining edges also have non-negative weights. The st-*MINCUT *of this graph (and hence, the* MAP *estimate of the corresponding random field) can be obtained in polynomial time.* **(b)** *The graph construction in (a), together with the additivity theorem of [48], is used to obtain the graph construction for the random field in Fig. 2.5.*

Submodular Energy Functions: Several equivalent definitions of submodular energy functions exist in the literature. Here, we will use the definition provided in [78, 82]. The energy function $Q(\cdot; \mathbf{D}, \boldsymbol{\theta})$ of a CRF (or $Q(\cdot, \mathbf{D}; \boldsymbol{\theta})$ for an MRF) defined over variables $\mathbf{v}$ which take a label from set $\mathbf{l}$ is said to be submodular if the following condition is satisfied for all $(a, b) \in \mathcal{E}$:

$$\theta^2_{ab;ii'} + \theta^2_{ab;jj'} \leq \theta^2_{ab;ij'} + \theta^2_{ab;ji'}, \forall i \leq j, i' \leq j', \tag{2.4.11}$$

where $l_i, l_j, l_{i'}, l_{j'} \in \mathbf{l}$.

We are now ready to describe the st-MINCUT based algorithm which provides the exact MAP estimate for binary submodular energy functions.

Binary Submodular Energy Functions: The problem of obtaining the MAP estimate of a submodular energy functions (for pairwise random fields) can be mapped on to an equivalent st-MINCUT problem [78]. In this work, we are particularly interested in binary submodular functions which are defined using a label set $\mathbf{l} = \{l_0, l_1\}$. In this case, the submodularity condition defined above reduces to

$$\theta^2_{ab;00} + \theta^2_{ab;11} \leq \theta^2_{ab;01} + \theta^2_{ab;10}, \forall (a, b) \in \mathcal{E}. \tag{2.4.12}$$

In order to illustrate the equivalence of MAP estimation for binary submodular functions and st-MINCUT, let us consider a simple example where $\mathbf{v} = \{v_a, v_b\}$. To obtain the MAP labelling of this random field, we define an st-MINCUT problem over a graph $\mathcal{G} = \{\{V_a, V_b\} \bigcup \{s, t\}, \mathbf{E}_G, w\}$.

For every cut $(\mathbf{V}_G^0, \mathbf{V}_G^1)$ of the graph $\mathcal{G}$, we can define a labelling f such that

$$f(a) = \begin{cases} 0 & \text{if} \quad V_a \in \mathbf{V}_G^0, \\ 1 & \text{otherwise.} \end{cases}$$

We would like to ensure that for every labelling f defined in this manner, the following holds true:

$$Q(f; \mathbf{D}, \boldsymbol{\theta}) = C(\mathbf{V}_G^0, \mathbf{V}_G^1) + \kappa, \tag{2.4.13}$$

where κ is some constant which is independent of the labelling f. If we can construct such a graph, it would follow that the labelling f^* corresponding to the st-MINCUT would provide the MAP estimate of the random field. A method to obtain the desired graph construction was proposed in [48]. Below, we provide a brief description of the graph construction. For details, we refer the reader to [48].

Recall that the st-MINCUT problem is defined using weights $w : \mathbf{E}_G \rightarrow \mathbb{R}^+$. Hence the graph $\mathcal{G}$, which we wish to construct, should contain only non-negative edges. We achieve this in two stages. In the first stage, we reparameterize the original parameter $\boldsymbol{\theta}$. As will be seen, this reparameterization allows us to construct the desired graph $\mathcal{G}$ (i.e. one that does not contain any negative edges and satisfies equation (2.4.13)) in the second stage.

We begin by defining the following reparameterization of the original parameter $\boldsymbol{\theta}$:

$$\overline{\theta}^1_{a;0} = \theta^1_{a;0} - \theta^2_{ab;10} + \theta^2_{ab;00} + \theta^2_{ab;11}, \tag{2.4.14}$$

$$\overline{\theta}^1_{a;1} = \theta^1_{a;1} + \theta^2_{ab;11}, \tag{2.4.15}$$

$$\overline{\theta}^1_{b;0} = \theta^1_{b;0} + \theta^2_{ab;10} - \theta^2_{ab;11}, \tag{2.4.16}$$

$$\overline{\theta}^1_{b;1} = \theta^1_{b;1}, \tag{2.4.17}$$

$$\overline{\theta}^2_{ab;00} = \overline{\theta}^2_{ab;10} = \overline{\theta}^2_{ab;11} = 0, \tag{2.4.18}$$

$$\overline{\theta}^2_{ab;01} = \theta^2_{ab;01} + \theta^2_{ab;10} - \theta^2_{ab;00} - \theta^2_{ab;11}. \tag{2.4.19}$$

It can be easily verified that the parameter $\overline{\boldsymbol{\theta}}$ defined above is a valid reparameterization of $\boldsymbol{\theta}$ since it satisfies equation (2.4.5). For example, consider the energy of the labelling $\{1, 1\}$ (i.e. $f(a) = 1$ and $f(b) = 1$) defined by $\overline{\boldsymbol{\theta}}$, i.e.

$$\begin{aligned} Q(\{1,1\}; \mathbf{D}, \overline{\boldsymbol{\theta}}) &= \overline{\theta}^1_{a;1} + \overline{\theta}^1_{b;1} + \overline{\theta}^2_{ab;11}, \\ &= \theta^1_{a;1} + \theta^1_{b;1} + \theta^2_{ab;11}, \\ &= Q(\{1,1\}; \mathbf{D}, \boldsymbol{\theta}). \end{aligned} \tag{2.4.20}$$

Note that some of the unary potentials defined above may be negative. This would lead to negative edges in the graph construction using the method of [48]. To avoid this problem (i.e. the negative edges), we consider the effect of adding a constant to all the unary potentials $\theta^1_{a;i}$ corresponding to a variable v_a. Since

every labelling f has to assign one and only one label to v_a, such an addition of constant to $\theta^1_{a;i}$ will result in adding the same constant to all labellings f. Hence, this operation would not change the MAP estimate of a given random field (since the constant is independent of the labelling f).

In our graph construction, we choose to add the terms $\theta^2_{ab;10} - \theta^2_{ab;00} - \theta^2_{ab;11} + \kappa_1$ and κ_2 to the unary potentials corresponding to v_a and v_b respectively (i.e. to $\overline{\theta}^1_{a;i}$ and $\overline{\theta}^1_{b;i}$ respectively). This would result in the following parameter $\hat{\boldsymbol{\theta}}$:

$$\hat{\theta}^1_{a;0} = \theta^1_{a;0} + \kappa_1, \tag{2.4.21}$$

$$\hat{\theta}^1_{a;1} = \theta^1_{a;1} + \theta^2_{ab;10} - \theta^2_{ab;00} + \kappa_1, \tag{2.4.22}$$

$$\hat{\theta}^1_{b;0} = \theta^1_{b;0} + \theta^2_{ab;10} - \theta^2_{ab;11} + \kappa_2, \tag{2.4.23}$$

$$\hat{\theta}^1_{b;1} = \theta^1_{b;1} + \kappa_2, \tag{2.4.24}$$

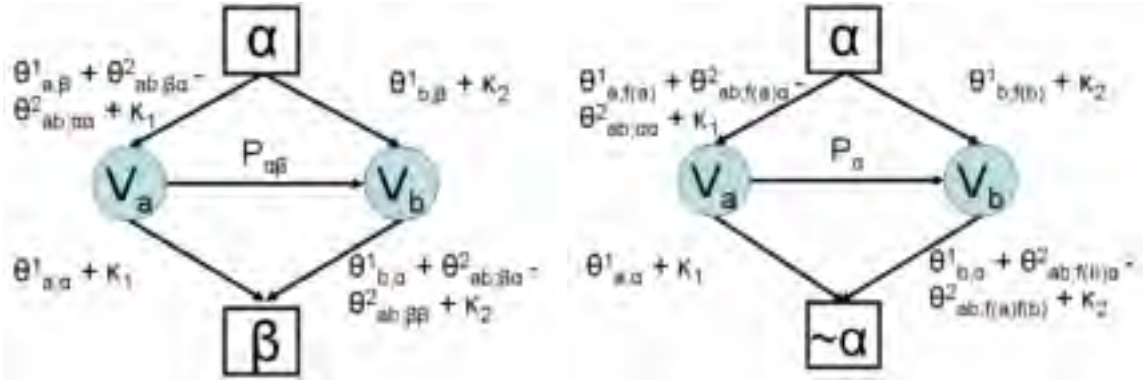
$$\hat{\theta}^2_{ab;00} = \hat{\theta}^2_{ab;10} = \hat{\theta}^2_{ab;11} = 0, \tag{2.4.25}$$

$$\hat{\theta}^2_{ab;01} = \theta^2_{ab;01} + \theta^2_{ab;10} - \theta^2_{ab;00} - \theta^2_{ab;11}. \tag{2.4.26}$$

Here, the constants κ_1 and κ_2 are chosen such that $\hat{\theta}^1_{a;i} \geq 0$ and $\hat{\theta}^1_{b;i} \geq 0$ for both $i = 0$ and $i = 1$. The above parameter $\hat{\boldsymbol{\theta}}$ can be easily represented using a graph as shown in Fig. 2.6(a). Specifically, the unary potentials correspond to weights of the edges between the terminals (i.e. the source s or the sink t) and the vertices. For example, $\hat{\theta}^1_{a;0}$ is the weight of the edge from V_a to t while $\hat{\theta}^1_{a;1}$ is the weight of the edge from s to V_a. When $f(a) = 0$ (i.e $V_a \in \mathbf{V}^0_G$), the edge with weight $\hat{\theta}^1_{a;0}$ contributes to the cost of the cut as desired. Similarly, when $f(a) = 1$ (i.e. $V_a \in \mathbf{V}^1_G$), the edge with weight $\hat{\theta}^1_{a;1}$ is used for computing the cut. The only non-zero pairwise potential in the above parameter $\hat{\boldsymbol{\theta}}$ is $\hat{\theta}^2_{ab;01}$, which is non-negative by the definition of submodularity (see equation (2.4.12)). This is represented by an edge between V_a and V_b which only contributes to the cost of the cut when $f(a) = 0$ and $f(b) = 1$ (as desired). Using the above observations, the graph in Fig. 2.6(a) can be shown to satisfy equation (2.4.13).

The equivalent graph construction for an arbitrary set $\mathbf{v}$ of random variables can be obtained by simply appending the graphs for all neighbouring pairs of random variables $(a, b) \in \mathcal{E}$ (using the additivity theorem of [48]). For example, the graph construction for the CRF in Fig. 2.5 (which has a binary submodular energy function) is shown in Fig. 2.6(b). Next, we describe two approximate algorithms for non-submodular energy functions defined over labels $\mathbf{l}$ where $|\mathbf{l}| > 2$.

The $\alpha\beta$-Swap Algorithm: The $\alpha\beta$-swap algorithm is an iterative procedure for obtaining an approximate MAP estimate for a random field by solving a series of st-MINCUT problems. It starts with an initial labelling f^0. At each iteration, it considers a pair of labels l_α and l_β together with the subset of variables $\mathbf{v}_{\alpha\beta} \subset \mathbf{v}$ which are currently assigned one of these labels. It then solves an st-MINCUT problem which potentially swaps the labels of these variables. For example, a



(a) (b)

Figure 2.7: **(a)** *The graph construction for an $\alpha\beta$-swap on two variables $\mathbf{v} = \{v_a, v_b\}$. Note that the α and β vertices represent the source and sink respectively. After the cut, the variables corresponding to the vertices connected to α (i.e. in the set $\mathbf{V}^0_G$) will be labelled l_α while the other variables will take a label l_β. Here $P_{\alpha\beta} = \theta^2_{ab;\alpha\beta} + \theta^2_{ab;\beta\alpha} - \theta^2_{ab;\alpha\alpha} - \theta^2_{ab;\beta\beta}$. When the pairwise potentials form a semi-metric, then it can be shown that $P_{\alpha\beta} \geq 0$ [15]. The terms $\kappa_1 = \min\{0, |\theta^1_{a;\alpha}|, |\theta^1_{a;\beta} + \theta^2_{ab;\beta\alpha} - \theta^2_{ab;\alpha\beta}|\}$ and $\kappa_2 = \min\{0, |\theta^1_{b;\beta}|, |\theta^1_{b;\alpha} + \theta^2_{ab;\beta\alpha} - \theta^2_{ab;\beta\beta}|\}$ ensure that the remaining edges also have non-negative weights.* **(b)** *The graph construction for an α-expansion on two variables $\mathbf{v} = \{v_a, v_b\}$. We assume that $f(a)$ and $f(b)$ denote the current labelling of variables v_a and v_b respectively. The vertices α and $\sim\alpha$ represent the source and the sink respectively. After the cut, the variables corresponding to the vertices connected to α will take the label l_α while the remaining variables will retain their previous labelling. The term $P_\alpha = \theta^2_{ab;\alpha f(b)} + \theta^2_{ab;f(a)\alpha} - \theta^2_{ab;\alpha\alpha} - \theta^2_{ab;f(a)f(b)}$. When the pairwise potentials form a metric, then it can be shown that $P_\alpha \geq 0$ [15]. Again, the terms κ_1 and κ_2 are chosen to ensure that the remaining edges also have non-negative weights.*

variable v_a which was assigned a label l_α in some previous iteration may now take a label l_β if it results in a lower energy. The algorithm terminates when the energy cannot be reduced further for any pairs of labels l_α and l_β.

Fig. 2.7(a) shows the st-MINCUT graph construction for performing $\alpha\beta$-swap for two variables v_a and v_b. This graph is constructed using the method of [48] (similar to Fig. 2.6(a)) using the appropriate values of the unary and pairwise potentials. Note that in general the edges of the graph (specifically the edge weight $P_{\alpha\beta}$) are not guaranteed to be positive. Hence, the st-MINCUT cannot be obtained in polynomial time[2]. However, Boykov *et al.* [15] showed that $\alpha\beta$-swap is applicable for a large and useful class of energy functions, i.e. when the pairwise potentials form a semi-metric[3], e.g. the Potts model or the truncated linear/quadratic model. This follows from the fact that $P_{\alpha\beta} = \theta^2_{ab;\alpha\beta} + \theta^2_{ab;\beta\alpha} - \theta^2_{ab;\alpha\alpha} - \theta^2_{ab;\beta\beta}$ (in Fig. 2.7(a)) is non-negative for semi-metrics.

The α-Expansion Algorithm: The α-expansion algorithm is also an iterative procedure which starts with an initial labelling f^0 and obtains an approximate MAP estimate by solving a series of st-MINCUT problems. Unlike $\alpha\beta$-swap, it considers one label l_α at each iteration together with all the variables $\mathbf{v}$. It then solves an st-MINCUT problem which allows the variables to either take the label l_α or retain their label from the previous iteration. The algorithm terminates when the energy cannot be reduced further for any label l_α.

Fig. 2.7(b) shows the st-MINCUT graph construction for performing α-expansion for two variables v_a and v_b (obtained using the method of [48]). Again, the graph is not guaranteed to have only positive edges (specifically the edges with weights P_α) for an arbitrary energy function. However, α-expansion is applicable when the pairwise potentials form a metric [15][4], e.g. the Potts model or the truncated linear model. This follows from the fact that $P_\alpha = \theta^2_{ab;\alpha f(b)} + \theta^2_{ab;f(a)\alpha} - \theta^2_{ab;\alpha\alpha} - \theta^2_{ab;f(a)f(b)}$ (in Fig. 2.7(b)) is non-negative for metrics. Note that unlike $\alpha\beta$-swap, the triangular inequality must be satisfied for α-expansion (to ensure $P_\alpha \geq 0$).

2.5. Algorithms for Computing Marginals

Marginal probabilities of random variables prove useful in many areas of Computer Vision. Specifically, (i) they provide an estimate of the partition function

[2]The st-MINCUT problem has a polynomial time solution when the weights of the graph are all non-negative. For a general graph, the st-MINCUT problem is called the MAXCUT problem which is well-known to be NP-hard.

[3]The pairwise potentials are said to form a semi-metric if they satisfy the following properties for all $(a, b) \in \mathcal{E}$: (i) $\theta^2_{ab;ij} = \theta^2_{ab;ji}$ for all $l_i, l_j \in \mathbf{l}$; (ii) $\theta^2_{ab;ij} \geq 0$ for all $l_i, l_j \in \mathbf{l}$; and (iii) $\theta^2_{ab;ij} = 0$ if, and only if, $i = j$.

[4]The pairwise potentials are said to form a metric if they form a semi-metric and obey the additional property of triangular inequality, i.e. $\theta^2_{ab;ij} + \theta^2_{ab;jk} \geq \theta^2_{ab;ik}$ for all $(a, b) \in \mathcal{E}$ and $l_i, l_j, l_k \in \mathbf{l}$.

for parameter estimation; (ii) they allow us to obtain samples from the probability distribution being modelled by the random field [23]. These samples can be used for determining the expectation of some function of the random variables (e.g. when using the EM algorithm [30]); and (iii) they provide the MMSE labelling of the random field as shown in equation (2.3.15). The MMSE labelling may sometimes be smoother and hence more desirable than the MAP estimate (see [91] for examples on stereo reconstruction).

Like MAP estimation, the problem of computing marginals for a general random field model is also NP-hard. However, it is well-known that the marginals for tree structured random fields can be computed using a message passing algorithm [70] (similar to the min-sum BP algorithm for MAP estimation described in § 2.4.1). Below, we described this algorithm and show how it can be generalized to obtain approximate marginals for arbitrary random fields.

2.5.1 Sum-Product Belief Propagation

Sum-product belief propagation (sum-product BP) is an iterative message passing algorithm proposed by Pearl [70] for approximately computing the marginals of random variables $\mathbf{v}$ which take labels from the set $\mathbf{l}$. Similar to min-sum BP, at each iteration every random variable v_a passes messages to its neighbouring random variable v_b (denoted by $\mathbf{m}_{ab}$). The j^{th} element of the message $\mathbf{m}_{ab}$ is given by

$$m_{ab;j} \longleftarrow \eta_1 \sum_{l_i \in \mathbf{l}} \left(\exp(-\theta^1_{a;i}) \exp(-\theta^2_{ab;ij}) \prod_{(a,c)\in\mathcal{E}, c\neq b} m_{ca;i} \right), \tag{2.5.1}$$

where η_1 is a constant which prevents numerical overflow and underflow. As can be seen in the above equations, the messages are computed as the sum of the product of terms. Hence, the name *sum-product* BP. Typically, all messages are initialized to 1. The algorithm is said to have converged when the rate of change of messages from one iteration to the next falls below a certain threshold.

Upon convergence, sum-product BP provides us with approximate marginals (or beliefs) for all random variables as

$$P'_{a;i} \longleftarrow \eta_2 \left(\exp(-\theta^1_{a;i}) \prod_{(a,c)\in\mathcal{E}} m_{ca;i} \right). \tag{2.5.2}$$

It also provides the approximate marginals for two neighbouring random variables v_a and v_b as

$$P'_{ab;ij} \longleftarrow \eta_2 \left(\exp(-\theta^1_{a;i} - \theta^1_{b;j} - \theta^2_{ab;ij}) \prod_{(a,c)\in\mathcal{E}, c\neq b} m_{ca;i} \prod_{(b,c)\in\mathcal{E}, c\neq a} m_{cb;j} \right). \tag{2.5.3}$$

Forward Messages	Backward Messages
$m_{ab;0} \leftarrow 5.652502e-02$	$m_{dc;0} \leftarrow 1.922752e-02$
$m_{ab;1} \leftarrow 1.378140e-01$	$m_{dc;1} \leftarrow 1.833234e-02$
$m_{bc;0} \leftarrow 3.155822e-03$	$m_{cb;0} \leftarrow 3.544267e-04$
$m_{bc;1} \leftarrow 2.905015e-03$	$m_{cb;1} \leftarrow 1.749953e-04$
$m_{cd;0} \leftarrow 1.572510e-04$	$m_{ba;0} \leftarrow 4.914555e-05$
$m_{cd;1} \leftarrow 6.044992e-05$	$m_{ba;1} \leftarrow 2.085102e-05$

Label	Beliefs
$f(a) = 0$	$P'_{a;0} \leftarrow 0.10502$
$f(a) = 1$	$P'_{a;1} \leftarrow 0.89498$
$f(b) = 0$	$P'_{b;0} \leftarrow 0.85991$
$f(b) = 1$	$P'_{b;1} \leftarrow 0.14009$
$f(c) = 0$	$P'_{c;0} \leftarrow 0.95814$
$f(c) = 1$	$P'_{c;1} \leftarrow 0.04186$
$f(d) = 0$	$P'_{d;0} \leftarrow 0.04548$
$f(d) = 1$	$P'_{d;1} \leftarrow 0.95452$

Table 2.5: *Forward and backward pass messages computed using sum-product* BP *for the* CRF *shown in Fig. 2.5. The messages are computed using equation (2.5.1) with* $\eta_1 = 1$ *for both iterations. The beliefs computed using equation (2.5.2) are also shown. The value of* η_2 *is set such that the marginals are normalized (i.e. sum to 1) for all variables.*

The notation P' is used to indicate that these are not necessarily the exact marginals. Note that these beliefs are different from the beliefs of min-sum BP which approximate the min-marginals. The beliefs are scaled by a constant η_2 such that

$$\sum_{l_i \in \mathbf{l}} P'_{a;i} = 1, \forall v_a \in \mathbf{v}, \tag{2.5.4}$$

$$\sum_{l_i, l_j \in \mathbf{l}} P'_{ab;ij} = 1, \forall (a,b) \in \mathcal{E}. \tag{2.5.5}$$

Sum-product BP allows us to obtain an approximate MMSE labelling f as

$$f(a) = \arg\max_{l_i \in \mathbf{l}} P'_{a;i}. \tag{2.5.6}$$

As mentioned earlier, the sum-product BP algorithm obtains the exact marginals for a tree structured random field [70]. Similar to min-sum BP (in § 2.4.1), this is achieved in two iterations: (i) forward pass where messages are sent from leaf nodes towards the root; and (ii) backward pass where the messages are sent from the root towards the leaf nodes. It follows that sum-product BP provides the exact MMSE estimate. Table 2.5 lists the messages and the beliefs computed by sum-product BP for the tree structured CRF shown in Fig. 2.5 (assuming v_a to be the leaf and v_d to be the root). Note that the marginals correspond exactly with those listed in table 2.3.

For a general random field (i.e. not necessarily tree structured), sum-product BP is not guaranteed to converge. However, in practice it often provides a good estimate of the marginals [66]. Further, it can be shown that when it converges, the solution provided by sum-product BP is a local minimum of the *Bethe approximation* of the free energy defined by the random field (described in more detail below) [114].

2.5.2 Other Algorithms

In this work, we only use the sum-product BP algorithm for computing the marginals. However, several other approximate algorithms have been proposed in the literature. We now briefly describe a few other approaches. A more comprehensive review of algorithms for computing marginals can be found in [7, 60].

Generalized Belief Propagation: The Bethe approximation of the free energy of a random field is given by

$$F_\beta = \sum_{(a,b)\in\mathcal{E}} \sum_{l_i,l_j\in\mathbf{l}} P_{ab;ij}\left(\log P_{ab;ij} + \theta^2_{ab;ij}\right) - \sum_{v_a\in\mathbf{v}} (q_a - 1) \sum_{l_i\in\mathbf{l}} P_{a;i}\left(\log P_{a;i} + \theta^1_{a;i}\right), \tag{2.5.7}$$

where q_a is the number of neighbouring random variables of v_a. Recall that $P_{a;i}$ and $P_{ab;ij}$ are the marginals corresponding to the given CRF. Yedidia *et al.* [114] showed that sum-product BP converges to a local minimum of the Bethe approximation of the free energy. In other words, the approximate marginals $P'_{a;i}$ and $P'_{ab;ij}$ in equations (2.5.2) and (2.5.3) define a stationary point of F_β. This can be verified by equating the derivatives of F_β with respect to $P_{a;i}$ and $P_{ab;ij}$ to zero. This connection between sum-product BP and Bethe approximation led them to propose a message passing algorithm which attempts to minimize more accurate approximations. These approximations, called Kikuchi approximations, are defined using a set of clusters of random variables $\mathbf{v}$. Let $\mathcal{C}$ be one such set of clusters, i.e. for every $c \in \mathcal{C}$ there exists a cluster of variables $\mathbf{v}_c = \{v_a | a \in c\}$. For each cluster c we can define a set $super(c)$ such that $d \in super(c)$ if, and only if, $c \subset d$. Using the sets $super(c)$ we assign an *over-counting number* o_c to each set c which is defined as

$$o_c = \begin{cases} 1 & \text{if} \quad c \text{ is the biggest cluster,} \\ 1 - \sum_{d\in super(c)} o_d & \text{otherwise.} \end{cases} \tag{2.5.8}$$

Given such a set $\mathcal{C}$ the Kikuchi approximation F_κ is specified as

$$F_\kappa = \sum_{c\in\mathcal{C}} o_c \left(\sum_{\mathbf{l}_i\in\mathbf{l}^{|c|}} P_{c;i}\left(\log P_{c;i} + \theta_{c;i}\right) \right), \tag{2.5.9}$$

where $\mathbf{l}_i = \{l_{i_1}, \cdots, l_{i_{|c|}}\} \in \mathbf{l}^{|c|}$ and $P_{c;i}$ is the marginal probability of cluster c taking labels $\mathbf{l}_i$. The term $\theta_{c;i}$ is given by

$$\theta_{c;i} = \sum_{a\in c} \theta^1_{a;i_a} + \sum_{(a,b)\in c} \theta^2_{ab;i_a i_b}. \tag{2.5.10}$$

Yedidia *et al.* [114] provided the message update rules that compute the approximate marginals which are stationary point of the Kikuchi approximation F_κ. This new message passing approach is called the Generalized Belief Propagation

(GBP) algorithm. We refer the interested reader to [114] for details. Similar to sum-product BP, GBP is not guaranteed to converge for general random fields. In practice, however, it provides better estimates of the marginal probabilities than sum-product BP at the cost of more computational time. Yuille [115] proposed convergent algorithms for minimizing the Bethe and Kikuchi approximation. However, these algorithms are slower than their message passing counterparts (i.e. sum-product BP and GBP).

Tree-reweighted sum-product Belief Propagation: The Bethe and Kikuchi approximations defined above suffer from the problem that they have multiple minima. Hence, any algorithm which attempts to minimize them, e.g. those described in [114, 115], is prone to converge to a local minimum. Wainwright *et al.* [100] addressed this issue by minimizing a new class of convex upper bounds on the log partition function of a given random field $\boldsymbol{\theta}$ (i.e. $\log Z(\boldsymbol{\theta})$). In order to describe this upper bound, we consider a set of tree-structured random fields $\mathcal{R} = \{R_1, \cdots, R_m\}$ defined over subsets of random variables $\mathbf{v}$ of the original random field. Let $\boldsymbol{\mu} = \{\mu_1, \mu_2, \cdots, \mu_m\}$ be a set of non-negative real numbers such that they sum to 1. Using the set $\boldsymbol{\mu}$, we define an *edge appearance term* for each pair of neighbouring random variable in the original random field, i.e. $(a, b) \in \mathcal{E}$, as

$$\mu_{ab} = \sum_{R_i \in I(a,b)} \mu(R_i), \tag{2.5.11}$$

where $R_i \in I(a, b)$ if, and only if, v_a and v_b are neighbouring random variables in R_i. By choosing $\boldsymbol{\mu}$ such that $\mu_{ab} > 0$ for all $(a, b) \in \mathcal{E}$, $\log Z(\boldsymbol{\theta})$ is bounded above by

$$\begin{aligned} \log Z(\boldsymbol{\theta}) \leq \quad & \min_{\mathbf{T}} F(\mathbf{T}; \boldsymbol{\mu}, \boldsymbol{\theta}), \\ \text{s.t.} \quad & \textstyle\sum_{l_j \in \mathbf{l}} T_{ab;ij} = T_{a;i}, \forall (a, b) \in \mathcal{E}, l_i \in \mathbf{l}, \\ & \textstyle\sum_{l_i \in \mathbf{l}} T_{a;i} = 1, \forall v_a \in \mathbf{v}, \end{aligned} \tag{2.5.12}$$

where the function $F(\mathbf{T}; \boldsymbol{\mu}, \boldsymbol{\theta})$ is given by

$$F(\mathbf{T}; \boldsymbol{\mu}, \boldsymbol{\theta}) = \sum_{v_a, l_i} T_{a;i} \log T_{a;i} - \sum_{(a,b) \in \mathcal{E}, l_i, l_j} \mu_{ab} T_{ab;ij} \log \frac{T_{ab;ij}}{(\sum_{l_k} T_{ab;ik})(\sum_{l_k} T_{ab;kj})} + \mathbf{T} \cdot \boldsymbol{\theta}. \tag{2.5.13}$$

It can be shown that the optimization problem (2.5.12) has a unique minima. Wainwright *et al.* [100] also defined an efficient algorithm, which we call tree-reweighted sum-product belief propagation, that obtains the upper bound of the log partition function (i.e. $\min F(\mathbf{T}; \boldsymbol{\mu}, \boldsymbol{\theta})$) together with the associated approximate marginals (i.e. $\mathbf{T}$). We refer the interested reader to [100] for details.

Min-marginals using st-MINCUT: For binary submodular random fields, Kohli and Torr [45] showed that the min-marginal energies of all the random variables

can be obtained by solving $2n$ st-MINCUT problems (where $n = |\mathbf{v}|$). In each of these st-MINCUT problems, the label of a particular variable is fixed to either l_0 or l_1 (by using $\theta^1_{a;1} = \infty$ or $\theta^2_{a;0} = \infty$ respectively). Clearly, the value of the st-MINCUT problems provide the min-marginals $p_{a;i}$. Solving each of these st-MINCUT problems individually would be computationally infeasible. However, since the structure and weights of the graphs corresponding to these st-MINCUT problems are very similar to each other, Kohli and Torr [45] observed that the problems lend themselves to the dynamic graph cuts algorithm [44]. The min-marginals obtained in this manner can be used as an approximation of the desired marginal probabilities.

Chapter 3

Convex Relaxations for MAP Estimation

3.1. Introduction

In the previous chapter, we outlined some methods which provide the exact MAP estimate for certain random fields, i.e. tree-structured random fields (using min-sum BP) and random fields with binary submodular energy functions (using st-MINCUT). We also presented some generalizations of these approaches that have been previously proposed in the literature to obtain an approximate MAP estimate for less restricted classes of random fields. In this chapter, we present a new formulation of the problem of MAP estimation. As will be seen shortly, this formulation allows us to utilize the well-studied *Convex Optimization* literature to obtain approximate MAP labellings.

Once again, we will assume throughout this chapter that we are given a CRF while noting that the methods outlined here are equally applicable to the MRF formulation. The CRF is defined using random variables $\mathbf{v} = \{v_0, \cdots, v_{n-1}\}$ each of which can take a label from the set $\mathbf{l} = \{l_0, \cdots, l_{h-1}\}$. Recall that for a given CRF with parameter $\boldsymbol{\theta}$, the energy of a labelling $f : \{0, \cdots, n-1\} \longrightarrow \{0, \cdots, h-1\}$ is given by

$$Q(f; \mathbf{D}, \boldsymbol{\theta}) = \sum_{v_a \in \mathbf{v}} \theta^1_{a;f(a)} + \sum_{(a,b) \in \mathcal{E}} \theta^2_{ab;f(a)f(b)}, \tag{3.1.1}$$

where $\theta^1_{a;f(a)}$ is the unary potential for assigning label $l_{f(a)}$ to v_a. Similarly, $\theta^2_{ab;f(a)f(b)}$ is the pairwise potential for assigning labels $l_{f(a)}$ and $l_{f(b)}$ to neighbouring random variables v_a and v_b respectively. Given the CRF, the goal of the MAP estimation problem is to obtain a labelling f^* which has the minimum energy (and equivalently the maximum posterior probability).

Before we describe the new formulation of MAP estimation, we will briefly outline some main concepts from the *Mathematical Optimization* literature [12, 90]. A reader who is familiar with Mathematical Optimization may skip the next section.

3.2. Mathematical Optimization - Background

In this section, we briefly set up the terminology of Mathematical Optimization which is employed throughout the remaining thesis. For more details on Mathematical Optimization, we refer the reader to [12, 90].

Optimization Problem: A Mathematical Optimization problem (or simply the optimization problem) has the following form:

$$\text{OP:} \quad \mathbf{x}^* = \arg\min_{\mathbf{x}} g(\mathbf{x}), \tag{3.2.1}$$

$$\text{subject to} \quad h_i(\mathbf{x}) \leq 0, \tag{3.2.2}$$

$$i = 1, \cdots, n_C. \tag{3.2.3}$$

Here $\mathbf{x}$ is called the *optimization variable* or simply the *variable*. The function $g(\cdot)$ which is being minimized (in equation (3.2.1)) is called the *objective function*. The functions $h_i(\cdot)$, $i = 1, \cdots, n_C$ (in equation 3.2.2) which restrict the values that $\mathbf{x}$ can take are called the *constraints*. The set of all $\mathbf{x}$ which satisfy all the constraints of the optimization problem OP is called the *feasibility region* (denoted by $\mathcal{F}(\text{OP})$). In other words,

$$\mathcal{F}(\text{OP}) = \{\mathbf{x} | h_i(\mathbf{x}) \leq 0, \forall i = 1, \cdots, n_C\}. \tag{3.2.4}$$

An *optimal solution* of OP is a point $\mathbf{x}^* \in \mathcal{F}(\text{OP})$ that minimizes the objective function $g(\cdot)$. Note that a given optimization problem may have more than one optimal solution. The value of the objective function at an optimal solution $\mathbf{x}^*$, i.e. $g(\mathbf{x}^*)$, is called the *optimal value* of the optimization problem OP.

Lagrangian Dual: The Lagrangian $\mathcal{L}(\cdot, \cdot)$ of the optimization problem OP (specified in equations (3.2.1)-(3.2.2)) is defined as

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = g(\mathbf{x}) + \sum_{i=1}^{n_C} \lambda_i h_i(\mathbf{x}), \mathbf{x} \in \mathcal{F}(\text{OP}), \lambda_i \geq 0. \tag{3.2.5}$$

The non-negative variables $\boldsymbol{\lambda} = \{\lambda_1, \cdots, \lambda_{n_C}\}$ are called Lagrangian multipliers. Note that for any feasible solution $\mathbf{x}$ of the problem OP, i.e. $\mathbf{x} \in \mathcal{F}(\text{OP})$, the following inequality holds true:

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) \leq g(\mathbf{x}), \forall \boldsymbol{\lambda}. \tag{3.2.6}$$

This can be easily verified using the definition of the Lagrangian (3.2.5) since $\lambda_i \geq 0$ and $h_i(\mathbf{x}) \leq 0$ for all $i = 1, \cdots, n_C$.

By employing the Lagrangian $\mathcal{L}(\cdot, \cdot)$, we can define the Lagrangian dual problem of OP as

$$\text{DP:} \quad \max_{\boldsymbol{\lambda}} \min_{\mathbf{x}} \quad \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}), \tag{3.2.7}$$

$$\mathbf{x} \in \mathcal{F}(\text{OP}), \tag{3.2.8}$$

$$\lambda_i \geq 0, \forall i = 1, \cdots, n_C. \tag{3.2.9}$$

It is common practice to refer to the problem DP as the *dual* and the original problem OP as the *primal*. Using inequality (3.2.6) it follows that the optimal value of the dual provides the lower bound of the optimal value of the primal.

This property is known as the *weak duality condition*. An important special case of the weak duality condition is when the optimal values of the primal and the dual problems are the same. Such primal and dual problems (which share the same optimal values) are said to satisfy the *strong duality condition*. Duals play a very important role when we are solving optimization problems [12]. In this work, we will use the concept of duality extensively in chapter 7.

Integer Program: Integer Program (IP) refers to the class of optimization problems where the elements of the variable $\mathbf{x}$ are constrained to take integer values. In other words, the feasibility region of an IP consists of points $\mathbf{x} \in \mathbb{Z}^m$ (where $\mathbb{Z}$ is the set of integers and m is the dimensionality of $\mathbf{x}$). Note that one can also define Mixed Integer Programs as optimization problems where some elements of the variable $\mathbf{x}$ are restricted to be integers. In this work, we will focus on pure IP only. The class of IP plays an important role in many areas of Computer Science due to its ability to model a large number of NP-hard problems. Of particular interest to us is the formulation of the MAP estimation problem as an IP which will be described in the next section.

Convex Optimization Problem: Typically, the subclass IP of optimization problems is very hard to solve. In contrast, an optimal solution of a convex optimization problem is usually easier to obtain. A convex optimization problem is one where the objective function $g(\cdot)$ and the constraints $h_i(\cdot)$ are convex functions[1]. Note that if all the constraints $h_i(\cdot)$ are convex functions, then the feasibility region forms a convex set[2]. Several convex optimization problems can be solved in a time which is polynomial in the number of variables (i.e. the dimensionality m of $\mathbf{x}$) and constraints (i.e. n_C). Below, we describe four such classes of problems which provide a natural platform for obtaining approximate solutions for the MAP estimation problem. It is also interesting to note that the strong duality condition holds for all these four classes. For other problems with polynomial time algorithms (and strong duality condition), we refer the reader to [12].

Linear Program: A linear program (LP) is an optimization problem with a linear objective function and linear constraints. Formally, an LP is specified as

[1] A function $g(\cdot)$ is convex if, and only if, the following holds true for all points $\mathbf{x}$ and $\mathbf{y}$ which belong to the domain of $g(\cdot)$: $g\left(\frac{\mathbf{x}+\mathbf{y}}{2}\right) \leq \frac{1}{2}\left(g(\mathbf{x}) + g(\mathbf{y})\right)$.

[2] A set $\mathbf{S}$ is said to be convex if, and only if, for all $\mathbf{x}, \mathbf{y} \in \mathbf{S}$ the following holds true: $\frac{\mathbf{x}+\mathbf{y}}{2} \in \mathbf{S}$. It is worth noting that if $h_i(\cdot)$ is a convex function then the set $\mathbf{S} = \{\mathbf{x} | h_i(\mathbf{x}) \leq 0\}$ is a convex set. Further, the intersection operation preserves convexity. In other words, if $\mathbf{S}_1$ and $\mathbf{S}_2$ are convex, then $\mathbf{S}_1 \bigcap \mathbf{S}_2$ is also convex.

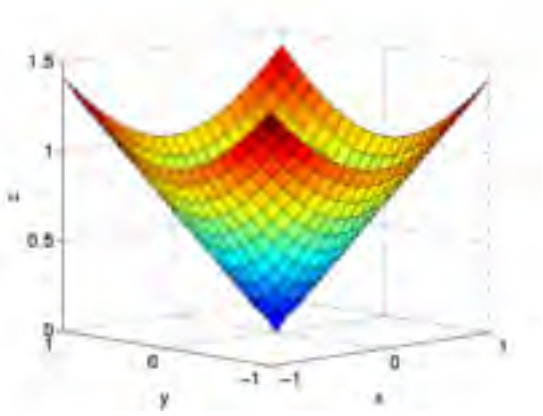


Figure 3.1: *An example of a 3 dimensional second order cone. The points satisfying an* SOC *constraint lie in the interior of the cone and form a convex set.*

follows:

$$\begin{aligned} \mathbf{x}^* = \quad & \arg\min_{\mathbf{x}} \mathbf{c}^\top \mathbf{x} + \kappa \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b}, \end{aligned} \tag{3.2.10}$$

where κ is some constant independent of $\mathbf{x}$. The vector $\mathbf{c}$ defines the objective function, while the matrix $\mathbf{A}$ and vector $\mathbf{b}$ specify the constraints.

Convex Quadratic Program: A convex quadratic program (QP) is an optimization problem with a convex quadratic objective function and linear constraints, i.e.

$$\begin{aligned} \mathbf{x}^* = \quad & \arg\min_{\mathbf{x}} ||\mathbf{A}\mathbf{x} + \mathbf{b}||^2 + \mathbf{c}^\top \mathbf{x} + \kappa \\ \text{s.t.} \quad & \mathbf{A}'\mathbf{x} \leq \mathbf{b}'. \end{aligned} \tag{3.2.11}$$

Clearly, LP is a special case of QP. This can be seen by setting $\mathbf{A} = 0$ and $\mathbf{b} = 0$ to obtain the general form of the LP (3.2.10).

Second Order Cone Program: A second order cone program (SOCP) is an optimization problem with a linear objective function and convex quadratic constraints, i.e.

$$\begin{aligned} \mathbf{x}^* = \quad & \arg\min_{\mathbf{x}} \mathbf{c}^\top \mathbf{x} + \kappa, \\ \text{s.t.} \quad & ||\mathbf{A}_i\mathbf{x} + \mathbf{b}_i||^2 \leq \mathbf{c}_i^\top \mathbf{x} + d_i, \\ & i = 1, \cdots, n_C. \end{aligned} \tag{3.2.12}$$

Convex quadratic constraints are also known as second order cone (SOC) constraints, Lorentz cone constraints or ice-cream cone constraints [12]. Fig. 3.1 shows the shape of a 3-D SOC. Clearly, the interior of the cone, i.e. points which

satisfy the SOC constraint, form a convex set. The class of QP is a special case of SOCP. For example, the QP (3.2.11) can be converted to an SOCP as follows:

$$\begin{aligned} \mathbf{x}^* = \quad & \arg\min_{(\mathbf{x},t)} t + \kappa, \\ \text{s.t.} \quad & ||\mathbf{A}\mathbf{x} + \mathbf{b}||^2 + \mathbf{c}^\top \mathbf{x} \leq t, \\ & \mathbf{A}'\mathbf{x} \leq \mathbf{b}', \end{aligned} \tag{3.2.13}$$

where t is some slack variable. If $(\mathbf{x}^*, t^*)$ is an optimal solution of the above SOCP, then it follows that $\mathbf{x}^*$ is an optimal solution of the QP (3.2.11) and

$$t^* = ||\mathbf{A}\mathbf{x}^* + \mathbf{b}||^2 + \mathbf{c}^\top \mathbf{x}, \tag{3.2.14}$$

since t is being minimized in the objective function.

Semidefinite Program: A semidefinite program (SDP) is a optimization problem with linear objective function and linear constraints defined over a variable matrix $\mathbf{X}$ which is restricted to be positive semidefinite[3] (denoted by $\mathbf{X} \succeq 0$). Formally, an SDP is written as:

$$\begin{aligned} \mathbf{X}^* = \quad & \arg\min_{\mathbf{X}} \mathbf{C} \bullet \mathbf{X}, \\ \text{s.t.} \quad & \mathbf{X} \succeq 0, \\ & \mathbf{A}_i \bullet \mathbf{X} \leq b_i, \\ & i = 1, \cdots, n_C. \end{aligned} \tag{3.2.15}$$

Here the operator ($\bullet$) denotes the Frobenius inner product, i.e.

$$\mathbf{A} \bullet \mathbf{B} = \sum_i \sum_j A_{ij} B_{ij}. \tag{3.2.16}$$

It can be shown that SOCP is a special case of SDP. Specifically, it can be shown that any SOC constraint $||\mathbf{A}_i\mathbf{x} + \mathbf{b}_i||^2 \leq \mathbf{c}_i^\top \mathbf{x} + d_i$ can be written as $||\mathbf{u}|| \leq t$, where

$$\mathbf{u} = \begin{pmatrix} 2(\mathbf{A}_i\mathbf{x} + \mathbf{b}_i) \\ \mathbf{c}_i^\top \mathbf{x} + d_i - 1 \end{pmatrix}, t = \mathbf{c}_i^\top \mathbf{x} + d_i + 1. \tag{3.2.17}$$

This SOC is equivalent to the following SDP constraint:

$$\begin{pmatrix} tI & \mathbf{u} \\ \mathbf{u}^\top & t \end{pmatrix} \succeq 0, \tag{3.2.18}$$

where I is the identity matrix of appropriate dimensions.

[3] An $m \times m$ matrix $\mathbf{X}$ is said to be positive semidefinite if all its m eigenvalues are non-negative. Equivalently, $\mathbf{X}$ is positive semidefinite if $\mathbf{c}^\top \mathbf{X}\mathbf{c} \geq 0$, for all $\mathbf{c} \in \mathbb{R}^m$. It is worth noting that any positive semidefinite matrix $\mathbf{X}$ can be written as $\mathbf{X} = \mathbf{U}^\top \mathbf{U}$ for an appropriate matrix $\mathbf{U}$.

Relaxation: We now describe the concept of relaxation [67] which is used extensively in the rest of the chapter (and also chapters 6 and 7). A relaxation of an optimization problem A is another optimization problem B such that

- The feasibility region of B is a strict superset of the feasibility region of A, i.e. $\mathcal{F}(\text{A}) \subset \mathcal{F}(\text{B})$.
- If $\mathbf{x} \in \mathcal{F}(\text{A})$ (which also implies that $\mathbf{x} \in \mathcal{F}(\text{B})$), then the value of the objective function of B is less than or equal to the value of the objective function of A at $\mathbf{x}$.

If the optimization problem B is convex, then it is called a convex relaxation of A. In the Optimization literature, typically problem A is hard to solve (e.g. an IP). Hence, its approximate solution is obtained using its easier-to-solve relaxation B (e.g. an LP, QP, SOCP or SDP relaxation of A). The quality of the solution obtained using B (by rounding the optimal solution of B to obtain a feasible solution of A) is usually specified by multiplicative or additive bounds which are described below.

Multiplicative Bound: Consider a set of optimization problems $\mathcal{A}$ and a relaxation scheme defined over this set $\mathcal{A}$. In other words, for every optimization problem $\text{A} \in \mathcal{A}$, the relaxation scheme provides a relaxation $\text{B} \in \mathcal{B}$ of A. Let e^A denote the optimal value of the optimization problem A. Further, let $\hat{e}^A$ denote the value of the objective function of A at the point obtained by rounding the optimal solution of its relaxation B. The relaxation scheme is said to provide a multiplicative bound of ρ for the set $\mathcal{A}$ if, and only if, the following condition is satisfied:

$$E(\hat{e}^A) \leq \rho e^A, \forall \text{A} \in \mathcal{A}, \tag{3.2.19}$$

where $E(\cdot)$ denotes the expectation of its argument under the rounding scheme employed.

Additive Bound: A relaxation scheme defined over the set of optimization problems $\mathcal{A}$ is said to provide an additive bound of σ for $\mathcal{A}$ if, and only if, the following holds true:

$$E(\hat{e}^A) \leq e^A + \sigma, \forall \text{A} \in \mathcal{A}. \tag{3.2.20}$$

Here the terms $\hat{e}^A$ and e^A are defined as above.

Tightness of the Bound: The multiplicative bound specified by a relaxation scheme defined over the set of optimization problems $\mathcal{A}$ is said to be *tight* if, and only if, there exists an $\text{A} \in \mathcal{A}$ such that

$$E(\hat{e}^A) = \rho e^A. \tag{3.2.21}$$

Similarly, the additive bound specified by a relaxation scheme defined over $\mathcal{A}$ is said to be tight if, and only if, there exists an $\text{A} \in \mathcal{A}$ such that

$$E(\hat{e}^A) = e^A + \sigma. \tag{3.2.22}$$

3.3. Convex Relaxations for MAP Estimation

We now formulate the problem of MAP estimation, defined using a discrete and finite set of labels, as an IP. We also briefly describe four approximate solutions for this IP based on convex relaxations.

3.3.1 Integer Programming Formulation

We consider a pairwise CRF defined over variables $\mathbf{v} = \{v_0, v_1, \cdots, v_{n-1}\}$ each of which can take a label from the finite and discrete set $\mathbf{l} = \{l_0, l_1, \cdots, l_{h-1}\}$. Recall that the set $\mathcal{E}$ associated with the CRF specifies a neighbourhood relationship between the random variables such that $(a, b) \in \mathcal{E}$ if, and only if, v_b is a neighbour of v_a. In order to formulate the MAP estimation problem of this CRF as an IP, we define a binary variable vector $\mathbf{x}$ of length nh. We denote the element of $\mathbf{x}$ at index $a \cdot h + i$ as $x_{a;i}$ where $v_a \in \mathbf{v}$ and $l_i \in \mathbf{l}$. These elements $x_{a;i}$ specify a labelling f such that

$$x_{a;i} = \begin{cases} 1 & \text{if} \quad f(a) = i, \\ -1 & \text{otherwise.} \end{cases}$$

We say that the variable $x_{a;i}$ *belongs to* variable v_a since it defines which label v_a does (or does not) take. Let $\mathbf{X} = \mathbf{x}\mathbf{x}^\top$. We refer to the $(a \cdot h + i, b \cdot h + j)^{th}$ element of the matrix $\mathbf{X}$ as $X_{ab;ij}$ where $v_a, v_b \in \mathbf{v}$ and $l_i, l_j \in \mathbf{l}$. Clearly the sum of the unary potentials for a labelling specified by $(\mathbf{x}, \mathbf{X})$ is given by

$$\sum_{v_a, l_i} \theta^1_{a;i} \frac{(1 + x_{a;i})}{2}. \tag{3.3.1}$$

Similarly the sum of the pairwise potentials for a labelling $(\mathbf{x}, \mathbf{X})$ is given by

$$\sum_{(a,b)\in\mathcal{E}, l_i, l_j} \theta^2_{ab;ij} \frac{(1 + x_{a;i})}{2} \frac{(1 + x_{b;j})}{2} = \sum_{(a,b)\in\mathcal{E}, l_i, l_j} \theta^2_{ab;ij} \frac{(1 + x_{a;i} + x_{b;j} + X_{ab;ij})}{4}. \tag{3.3.2}$$

Hence, the following IP finds the labelling with the minimum energy, i.e. it is equivalent to the MAP estimation problem:

$$\begin{aligned} \text{IP:} \quad & \mathbf{x}^* = \arg\min_{\mathbf{x}} \textstyle\sum_{v_a, l_i} \theta^1_{a;i} \frac{(1+x_{a;i})}{2} + \sum_{(a,b)\in\mathcal{E}, l_i, l_j} \theta^2_{ab;ij} \frac{(1+x_{a;i}+x_{b;j}+X_{ab;ij})}{4} \\ \text{s.t.} \quad & \mathbf{x} \in \{-1, 1\}^{nh}, && (3.3.3) \\ & \textstyle\sum_{l_i \in \mathbf{l}} x_{a;i} = 2 - h, && (3.3.4) \\ & \mathbf{X} = \mathbf{x}\mathbf{x}^\top. && (3.3.5) \end{aligned}$$

Constraints (3.3.3) and (3.3.5) specify that the variables $\mathbf{x}$ and $\mathbf{X}$ are binary such that $X_{ab;ij} = x_{a;i} x_{b;j}$. We will refer to them as the *integer constraints*.

Constraint (3.3.4), which specifies that each variable should be assigned only one label, is known as the *uniqueness constraint*. Note that one uniqueness constraint is specified for each variable v_a. Solving the above IP is in general NP-hard. We now describe four different convex relaxations of the IP.

3.3.2 Linear Programming Relaxation

The LP relaxation (proposed by Schlesinger [79] for a special case[1] and independently in [17, 51, 101] for the general case), which we call LP-S, is given as follows[2]:

$$\text{LP-S:} \quad \mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{v_a, l_i} \theta^1_{a;i} \frac{(1+x_{a;i})}{2} + \sum_{(a,b)\in\mathcal{E}, l_i, l_j} \theta^2_{ab;ij} \frac{(1+x_{a;i}+x_{b;j}+X_{ab;ij})}{4}$$

$$\text{s.t.} \quad \mathbf{x} \in [-1,1]^{nh}, \mathbf{X} \in [-1,1]^{nh\times nh}, \tag{3.3.6}$$

$$\textstyle\sum_{l_i \in \mathbf{l}} x_{a;i} = 2 - h, \tag{3.3.7}$$

$$\textstyle\sum_{l_j \in \mathbf{l}} X_{ab;ij} = (2-h) x_{a;i}, \tag{3.3.8}$$

$$X_{ab;ij} = X_{ba;ji}, \tag{3.3.9}$$

$$1 + x_{a;i} + x_{b;j} + X_{ab;ij} \geq 0. \tag{3.3.10}$$

In the LP-S relaxation only those elements $X_{ab;ij}$ of $\mathbf{X}$ are used for which $(a,b) \in \mathcal{E}$ and $l_i, l_j \in \mathbf{l}$. Unlike the IP, the feasibility region of the above problem is relaxed such that the variables $x_{a;i}$ and $X_{ab;ij}$ lie in the interval $[-1,1]$. Further, the constraint (3.3.5) is replaced by equation (3.3.8) which is called the *marginalization constraint* [101]. One marginalization constraint is specified for each $(a,b) \in \mathcal{E}$ and $l_i \in \mathbf{l}$. Constraint (3.3.9) specifies that $\mathbf{X}$ is symmetric. Constraint (3.3.10) ensures that $\theta^2_{ab;ij}$ is multiplied by a number between 0 and 1 in the objective function. These constraints (3.3.9) and (3.3.10) are defined for all $(a,b) \in \mathcal{E}$ and $l_i, l_j \in \mathbf{l}$. Note that the above constraints are not exhaustive, i.e. it is possible to specify other constraints for the problem of MAP estimation (as will be seen in the different relaxations described in the subsequent sections).

Properties of the LP-S Relaxation:

- Since the LP-S relaxation specifies a linear program it can be solved in polynomial time. A labelling f can then be obtained by rounding the (possibly fractional) solution of the LP-S.
- Using the rounding scheme of [42], the LP-S provides a multiplicative bound of 2 when the pairwise potentials form a Potts model [17].

[1] The special case considered in [79] consists of pairwise potentials which specify a hard constraint, i.e. they are either 0 or ∞.

[2] The formulation of the LP-S relaxation presented here uses a slightly different notation to the ones described in [46, 101]. However, it can easily be shown that the two formulations are equivalent by using the variables $\mathbf{y}$ and $\mathbf{Y}$ instead of $\mathbf{x}$ and $\mathbf{X}$ such that $y_{a;i} = \frac{1+x_{a;i}}{2}, Y_{ab;ij} = \frac{1+x_{a;i}+x_{b;j}+X_{ab;ij}}{4}$.

- Using the rounding scheme of [17], LP-S obtains a multiplicative bound of $2 + \sqrt{2}$ for truncated linear pairwise potentials.
- LP-S provides a multiplicative bound of 1 when the energy function $Q(\cdot; \mathbf{D}, \boldsymbol{\theta})$ of the CRF is submodular [82] (also see [37, 78] for the st-MINCUT graph construction for minimizing submodular energy functions).
- The LP-S relaxation provides the same optimal solution for all reparameterizations $\overline{\boldsymbol{\theta}}$ of $\boldsymbol{\theta}$ (i.e. for all $\overline{\boldsymbol{\theta}} \equiv \boldsymbol{\theta}$) [46].

Although the LP-S relaxation can be solved in polynomial time, the state of the art Interior Point algorithms can only handle upto a few thousand variables and constraints. In order to overcome this deficiency several closely related algorithms have been proposed in the literature for approximately solving the Lagrangian dual of LP-S [46, 101, 107][3]. We will return to these algorithms and build upon them in chapter 7.

3.3.3 Quadratic Programming Relaxation

We now describe the QP relaxation for the MAP estimation IP which was proposed by Ravikumar and Lafferty [75]. To this end, it would be convenient to reformulate the objective function of the IP using a vector of unary potentials of length nh (denoted by $\hat{\boldsymbol{\theta}}^1$) and a matrix of pairwise potentials of size $nh \times nh$ (denoted by $\hat{\boldsymbol{\theta}}^2$). The element of the unary potential vector at index $(a \cdot h + i)$ is defined as

$$\hat{\theta}^1_{a;i} = \theta^1_{a;i} - \sum_{v_c \in \mathbf{v}} \sum_{l_k \in \mathbf{l}} |\theta^2_{ac;ik}|, \tag{3.3.11}$$

where $v_a \in \mathbf{v}$ and $l_i \in \mathbf{l}$. The $(a \cdot h + i, b \cdot h + j)^{th}$ element of the pairwise potential matrix $\hat{\boldsymbol{\theta}}^2$ is defined such that

$$\hat{\theta}^2_{ab;ij} = \begin{cases} \sum_{v_c \in \mathbf{v}} \sum_{l_k \in \mathbf{l}} |\theta^2_{ac;ik}|, & \text{if} \quad a = b, i = j, \\ \theta^2_{ab;ij} & \text{otherwise,} \end{cases}$$

where $v_a, v_b \in \mathbf{v}$ and $l_i, l_j \in \mathbf{l}$. In other words, the potentials are modified by defining a pairwise potential $\hat{\theta}^2_{aa;ii}$ and subtracting the value of that potential from the corresponding unary potential $\theta^1_{a;i}$. The advantage of this reformulation is that the matrix $\hat{\boldsymbol{\theta}}^2$ is guaranteed to be positive semidefinite, i.e. $\hat{\boldsymbol{\theta}}^2 \succeq 0$. Using the fact that for $x_{a;i} \in \{-1, 1\}$,

$$\left(\frac{1 + x_{a;i}}{2}\right)^2 = \frac{1 + x_{a;i}}{2}, \tag{3.3.12}$$

[3] It is worth noting that the α-expansion algorithm described in the previous chapter has also been shown to be related to LP-S [50].

it can be shown that the following is equivalent to the MAP estimation problem [75]:

$$\text{QP-RL:} \quad \mathbf{x}^* = \arg\min_{\mathbf{x}} \left(\tfrac{\mathbf{1}+\mathbf{x}}{2}\right)^\top \hat{\boldsymbol{\theta}}^1 + \left(\tfrac{\mathbf{1}+\mathbf{x}}{2}\right)^\top \hat{\boldsymbol{\theta}}^2 \left(\tfrac{\mathbf{1}+\mathbf{x}}{2}\right), \tag{3.3.13}$$

$$\text{s.t.} \quad \textstyle\sum_{l_i \in \mathbf{l}} x_{a;i} = 2 - h, \forall v_a \in \mathbf{v}, \tag{3.3.14}$$

$$\mathbf{x} \in \{-1, 1\}^{nh}, \tag{3.3.15}$$

where $\mathbf{1}$ is a vector of appropriate dimensions whose elements are all equal to 1. By relaxing the feasibility region of the above problem to $\mathbf{x} \in [-1, 1]^{nh}$, the resulting QP can be solved in polynomial time since $\hat{\boldsymbol{\theta}}^2 \succeq 0$ (i.e. the relaxation of the QP (3.3.13)-(3.3.15) is convex). We call the above relaxation QP-RL. Note that in [75], the QP-RL relaxation was described using the variable $\mathbf{y} = \frac{\mathbf{1}+\mathbf{x}}{2}$. However, the above formulation can easily be shown to be equivalent to the one presented in [75].

Ravikumar and Lafferty [75] proposed a rounding scheme for QP-RL (different from the ones used in [17, 42]) that provides an additive bound of $\frac{S}{4}$ for the MAP estimation problem, where $S = \sum_{(a,b)\in\mathcal{E}} \sum_{l_i, l_j \in \mathbf{l}} |\theta^2_{ab;ij}|$ (i.e. S is the sum of the absolute values of all pairwise potentials) [75]. Under their rounding scheme, this bound can be shown to be tight using a random field defined over two random variables which specifies uniform unary potentials and Ising model pairwise potentials. Further, they also proposed an efficient iterative procedure for solving the QP-RL relaxation approximately. However, unlike LP-S, no multiplicative bounds have been established for the QP-RL formulation for special cases of the MAP estimation problem.

3.3.4 Semidefinite Programming Relaxation

The SDP relaxation of the MAP estimation problem replaces the non-convex constraint $\mathbf{X} = \mathbf{x}\mathbf{x}^\top$ by the convex semidefinite constraint $\mathbf{X} - \mathbf{x}\mathbf{x}^\top \succeq 0$ [22, 31, 57] which can be expressed as

$$\begin{pmatrix} 1 & \mathbf{x}^\top \\ \mathbf{x} & \mathbf{X} \end{pmatrix} \succeq 0, \tag{3.3.16}$$

using Schur's complement [12]. Further, like LP-S, it relaxes the integer constraints by allowing the variables $x_{a;i}$ and $X_{ab;ij}$ to lie in the interval $[-1, 1]$ with $X_{aa;ii} = 1$ for all $v_a \in \mathbf{v}, l_i \in \mathbf{l}$. Note that the value of $X_{aa;ii}$ is derived using the fact that $X_{aa;ii} = x^2_{a;i}$. Since $x_{a;i}$ can only take the values -1 or 1 in the MAP estimation IP, it follows that $X_{aa;ii} = 1$. The SDP relaxation is a well-studied approach which provides accurate solutions for the MAP estimation problem (e.g. see [103]). However, due to its computational inefficiency, it is not practically useful for large scale problems with $nh > 1000$. See however [68, 77, 93].

3.3.5 Second Order Cone Programming Relaxation

We now describe the SOCP relaxation that was proposed by Muramatsu and Suzuki [64] for the MAXCUT problem (i.e. MAP estimation with $h = 2$) and later extended for a general label set [55]. This relaxation, which we call SOCP-MS, is based on the technique of Kim and Kojima [40]. For completeness we first describe the general technique of [40] and later show how SOCP-MS can be derived using it.

SOCP Relaxations: Kim and Kojima [40] observed that the SDP constraint $\mathbf{X} - \mathbf{x}\mathbf{x}^\top \succeq 0$ can be further relaxed to SOC constraints. Their technique uses the fact that the Frobenius inner product of two semidefinite matrices is non-negative. For example, consider the inner product of a fixed matrix $\mathbf{C} = \mathbf{U}\mathbf{U}^\top \succeq 0$ with $\mathbf{X} - \mathbf{x}\mathbf{x}^\top$ (which, by the SDP constraint, is also positive semidefinite). This inner product can be expressed as an SOC constraint as follows:

$$\mathbf{C} \bullet (\mathbf{X} - \mathbf{x}\mathbf{x}^\top) \geq 0, \tag{3.3.17}$$

$$\Rightarrow \|(\mathbf{U})^\top \mathbf{x}\|^2 \leq \mathbf{C} \bullet \mathbf{X}. \tag{3.3.18}$$

Hence, by using a set of matrices $\mathcal{S} = \{\mathbf{C}^k | \mathbf{C}^k = \mathbf{U}^k (\mathbf{U}^k)^\top \succeq 0, k = 1, 2, \ldots, n_C\}$, the SDP constraint can be further relaxed to n_C SOC constraints, i.e.

$$\Rightarrow \|(\mathbf{U}^k)^\top \mathbf{x}\|^2 \leq \mathbf{C}^k \bullet \mathbf{X}, k = 1, \cdots, n_C. \tag{3.3.19}$$

It can be shown that, for the above set of SOC constraints to be equivalent to the SDP constraint, $n_C = \infty$. However, in practice, we can only specify a finite set of SOC constraints. Each of these constraints may involve some or all variables $x_{a;i}$ and $X_{ab;ij}$. For example, if $C^k_{ab;ij} = 0$, then the k^{th} SOC constraint will not involve $X_{ab;ij}$ (since its coefficient will be 0).

The SOCP-MS Relaxation: Consider a pair of neighbouring variables v_a and v_b, i.e. $(a, b) \in \mathcal{E}$, and a pair of labels l_i and l_j. These two pairs define the following variables: $x_{a;i}$, $x_{b;j}$, $X_{aa;ii} = X_{bb;jj} = 1$ and $X_{ab;ij} = X_{ba;ji}$ (since $\mathbf{X}$ is symmetric). For each such pair of variables and labels, the SOCP-MS relaxation specifies two SOC constraints which involve only the above variables [55, 64]. In order to specify the exact form of these SOC constraints, we need the following definitions.

Using the variables v_a and v_b (where $(a, b) \in \mathcal{E}$) and labels l_i and l_j, we define the submatrices $\mathbf{x}^{(a,b,i,j)}$ and $\mathbf{X}^{(a,b,i,j)}$ of $\mathbf{x}$ and $\mathbf{X}$ respectively as:

$$\mathbf{x}^{(a,b,i,j)} = \begin{pmatrix} x_{a;i} \\ x_{b;j} \end{pmatrix}, \mathbf{X}^{(a,b,i,j)} = \begin{pmatrix} X_{aa;ii} & X_{ab;ij} \\ X_{ba;ji} & X_{bb;jj} \end{pmatrix}. \tag{3.3.20}$$

The SOCP-MS relaxation specifies SOC constraints of the form:

$$\|(\mathbf{U}^k_{MS})^\top \mathbf{x}^{(a,b,i,j)}\|^2 \leq \mathbf{C}^k_{MS} \bullet \mathbf{X}^{(a,b,i,j)}, \tag{3.3.21}$$

for all pairs of neighbouring variables $(a,b) \in \mathcal{E}$ and labels $l_i, l_j \in \mathbf{l}$. To this end, it uses the following two matrices:

$$\mathbf{C}^1_{MS} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, \mathbf{C}^2_{MS} = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}. \tag{3.3.22}$$

In other words SOCP-MS specifies a total of $2|\mathcal{E}|h^2$ SOC constraints. Note that both the matrices $\mathbf{C}^1_{MS}$ and $\mathbf{C}^2_{MS}$ defined above are positive semidefinite, and hence can be written as $\mathbf{C}^1_{MS} = \mathbf{U}^1_{MS}(\mathbf{U}^1_{MS})^\top$ and $\mathbf{C}^1_{MS} = \mathbf{U}^1_{MS}(\mathbf{U}^1_{MS})^\top$ where

$$\mathbf{U}^1_{MS} = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \text{ and } \mathbf{U}^2_{MS} = \begin{pmatrix} 0 & -1 \\ 0 & 1 \end{pmatrix}, \tag{3.3.23}$$

Substituting these matrices in inequality (3.3.21) we see that the constraints defined by the SOCP-MS relaxation are given by

$$\begin{aligned}
&\|(\mathbf{U}^1_{MS})^\top \mathbf{x}^{(a,b,i,j)}\|^2 \leq \mathbf{C}^1_{MS} \bullet \mathbf{X}^{(a,b,i,j)}, \\
&\|(\mathbf{U}^2_{MS})^\top \mathbf{x}^{(a,b,i,j)}\|^2 \leq \mathbf{C}^2_{MS} \bullet \mathbf{X}^{(a,b,i,j)}, && (3.3.24)\\
\Rightarrow \quad &\left\| \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x_{a;i} \\ x_{b;j} \end{pmatrix} \right\|^2 = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \bullet \begin{pmatrix} X_{aa;ii} & X_{ab;ij} \\ X_{ba;ji} & X_{bb;jj} \end{pmatrix}, \\
&\left\| \begin{pmatrix} 0 & 0 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} x_{a;i} \\ x_{b;j} \end{pmatrix} \right\|^2 = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \bullet \begin{pmatrix} X_{aa;ii} & X_{ab;ij} \\ X_{ba;ji} & X_{bb;jj} \end{pmatrix}, && (3.3.25)\\
\Rightarrow \quad &(x_{a;i} + x_{b;j})^2 \leq X_{aa;ii} + X_{bb;jj} + X_{ab;ij} + X_{ba;ji}, \\
&(x_{a;i} - x_{b;j})^2 \leq X_{aa;ii} + X_{bb;jj} - X_{ab;ij} - X_{ba;ji}, && (3.3.26)\\
\Rightarrow \quad &(x_{a;i} + x_{b;j})^2 \leq 2 + 2X_{ab;ij}, \\
&(x_{a;i} - x_{b;j})^2 \leq 2 - 2X_{ab;ij}. && (3.3.27)
\end{aligned}$$

The last expression is obtained using the fact that $\mathbf{X}$ is symmetric and $X_{aa;ii} = 1$, for all $v_a \in \mathbf{v}$ and $l_i \in \mathbf{l}$. Hence, in the SOCP-MS formulation, the MAP estimation IP is relaxed to

$$\begin{aligned}
\text{SOCP-MS:} \quad \mathbf{x}^* = \arg\min_{\mathbf{x}} &\textstyle\sum_{v_a,l_i} \theta^1_{a;i} \frac{(1+x_{a;i})}{2} + \sum_{(a,b)\in\mathcal{E},l_i,l_j} \theta^2_{ab;ij} \frac{(1+x_{a;i}+x_{b;j}+X_{ab;ij})}{4} \\
\text{s.t.} \quad &\mathbf{x} \in [-1,1]^{nh}, \mathbf{X} \in [-1,1]^{nh \times nh}, && (3.3.28)\\
&\textstyle\sum_{l_i \in \mathbf{l}} x_{a;i} = 2 - h, && (3.3.29)\\
&(x_{a;i} - x_{b;j})^2 \leq 2 - 2X_{ab;ij}, && (3.3.30)\\
&(x_{a;i} + x_{b;j})^2 \leq 2 + 2X_{ab;ij}, && (3.3.31)\\
&X_{ab;ij} = X_{ba;ji}, && (3.3.32)\\
&1 + x_{a;i} + x_{b;j} + X_{ab;ij} \geq 0. && (3.3.33)
\end{aligned}$$

We refer the reader to [55, 64] for details. The SOCP-MS relaxation yields the supremum and infimum for the elements of the matrix $\mathbf{X}$ using constraints (3.3.30) and (3.3.31) respectively, i.e.

$$\frac{(x_{a;i} + x_{b;j})^2}{2} - 1 \leq X_{ab;ij} \leq 1 - \frac{(x_{a;i} - x_{b;j})^2}{2}. \tag{3.3.34}$$

These constraints are specified for all $(a, b) \in \mathcal{E}$ and $l_i, l_j \in \mathbf{l}$. When the objective function of SOCP-MS is minimized, one of the two inequalities would be satisfied as an equality. This can be proved by assuming that the value for the vector $\mathbf{x}$ has been fixed. Hence, the elements of the matrix $\mathbf{X}$ should take values such that it minimizes the objective function subject to the constraints (3.3.30) and (3.3.31). Clearly, the objective function would be minimized when $X_{ab;ij}$ equals either its supremum or infimum value, depending on the sign of the corresponding pairwise potential $\theta^2_{ab;ij}$, i.e.

$$X_{ab;ij} = \begin{cases} \frac{(x_{a;i}+x_{b;j})^2}{2} - 1 & \text{if} \quad \theta^2_{ab;ij} > 0, \\ 1 - \frac{(x_{a;i}-x_{b;j})^2}{2} & \text{otherwise.} \end{cases}$$

Similar to the LP-S and QP-RL relaxations defined above, the SOCP-MS relaxation can also be solved in polynomial time. To the best our knowledge, no bounds have been established for the SOCP-MS relaxation in earlier work. Furthermore, no previous specialized algorithms exist for solving SOCP-MS (or indeed any other SOCP relaxation) efficiently.

We will return to the above convex relaxations (specifically LP-S, QP-RL and SOCP-MS) and identify the best one in the second part of the thesis (chapters 6 and 7). Before that, we turn our attention to the two Computer Vision problems described in chapter 1.

Chapter 4

Learning Layered Motion Segmentations of Video

In this chapter, we present the first of two applications of optimization for Computer Vision, namely unsupervised motion segmentation. To this end, we develop an approach for learning a layered representation of a scene from a video. Our method is applicable to any video containing piecewise parametric motion. The learnt model is a composition of layers, which consist of one or more *segments*. The shape of each segment is represented using a binary matte and its appearance is given by the RGB value for each point belonging to the matte.

In terms of the probabilistic model, the main contributions of our method are: (i) The effects of image projection, lighting, and motion blur are taken into account; (ii) Spatial continuity is explicitly modelled resulting in contiguous segments; and (iii) Unlike previous approaches [94, 105], our method does not use reference frame(s) for initialization.

In terms of optimization, the main contributions are: (i) A novel algorithm for obtaining the initial estimate of the model by dividing the scene into rigidly moving components using efficient sum-product belief propagation; and (ii) Refining the initial estimate using $\alpha\beta$-swap and α-expansion algorithms, which guarantee a strong local minima.

Results are presented on several classes of objects with different types of camera motion, e.g. videos of humans and animals walking which are shot with static or moving cameras. We compare our method with the state of the art and demonstrate significant improvements.

4.1. Introduction

We present an approach for learning a layered representation from a video for motion segmentation. The layered representation is composed of a set of layers, each of which contains a number of segments. Each segment is defined by its shape (represented as a binary matte) and appearance (represented as RGB values for every point belonging to the segment). The model accounts for the effects of occlusion, lighting and motion blur. Our method for learning this model is applicable to any video containing piecewise parametric motion, e.g. piecewise homography, without any restrictions on camera motion. For example, Fig. 4.1 shows one such sequence where a layered representation can be learnt and used to segment the walking person from the static background.

Many different approaches for motion segmentation have been reported in the literature. Important issues are: (i) whether the methods are feature-based or dense; (ii) whether they model occlusion; (iii) whether they model spatial continuity; (iv) whether they apply to multiple frames (i.e. a video sequence); and (v) whether they are independent of which frames are used for initialization.

A comprehensive survey of feature-based methods can be found in [95]. Most of these method rely on computing a homography corresponding to the motion of a planar object. This limits their application to a restricted set of scenes and motions. Dense methods [8, 19, 94, 105] overcome this deficiency by computing pixel-wise motion. However, many dense approaches do not model occlusion which can lead to overcounting of data when obtaining the segmentation, e.g. see [8, 19].

Chief amongst the methods which do model occlusion are those that use a layered representation [104]. One such approach, described in [105], divides a scene into (almost) planar regions for occlusion reasoning. Torr *et al.* [94] extend this representation by allowing for parallax disparity. However, these methods rely on a keyframe for the initial estimation. Other approaches [38, 108] overcome this problem by using layered flexible sprites. A flexible sprite is a 2D appearance map and matte of an object which is allowed to deform from frame to frame according to pure translation. Winn *et al.* [110] extend the model to handle affine deformations. However, these methods do not enforce spatial continuity, i.e. they assume each pixel is labelled independent of its neighbours. This leads to non-contiguous segmentation when the foreground and background are similar in appearance (see Fig. 4.23(b)). Most of these approaches, namely those described in [19, 38, 94, 104, 105], use either EM or variational methods for learning the parameters of the model which makes them prone to local minima.

Wills *et al.* [109] noted the importance of spatial continuity when learning the regions in a layered representation. Given an initial estimate, they learn the

Figure 4.1: *Four intermediate frames of a 31 frame video sequence of a person walking sideways where the camera is static. Given the sequence, the model which best describes the person and the background is learnt in an unsupervised manner. Note that the arm always partially occludes the torso which makes it important to handle occlusion. As the person is moving, her change in appearance due to motion blur should also be taken into account.*

shape of the regions using the powerful α-expansion algorithm [15] which guarantees a strong local minima. However, their method does not deal with more than 2 views. In our earlier work [54], we described a similar approach to [109] for a video sequence which automatically learns a model of an object. Unlike previous approaches for learning object models (e.g. see [74], which approximates the object using parallel lines of contrast) our approach obtains a more accurate shape representation using motion segmentation. However, it depends on a keyframe to obtain an initial estimate of the model. This has the disadvantage that points not visible in the keyframe are not included in the model, which leads to incomplete segmentation.

In this chapter, we present a model which does not suffer from the problems mentioned above, i.e. (i) it models occlusion; (ii) it models spatial continuity; (iii) it handles multiple frames; and (iv) it is learnt independent of keyframes. An initial estimate of the model is obtained based on a method to estimate image motion with discontinuities (similar to dense methods) using a new efficient sum-product belief propagation (sum-product BP) algorithm. Although we use the new sum-product BP algorithm only for estimating image motion, it is worth noting that it is applicable to any random field model. The image motion computed in this manner is approximated as piecewise parametric motion. However, unlike the feature-based methods described in [95], our two step process for obtaining the parametric motion allows us to learn the model for a wide variety of scenes. Given the initial estimate, the shape of the segments along with the layering are learnt by minimizing an objective function using $\alpha\beta$-swap and α-expansion algorithms [15]. Results are demonstrated on several classes of objects with different types of camera motion.

In the next section, we describe the layered representation. Such a model is particularly suited for applications like motion segmentation. In section 4.3, we present a five stage approach to learn the model parameters from a video. Results of our method are presented in section 4.4 for several classes of objects and camera motions.

4.2. Layered Representation

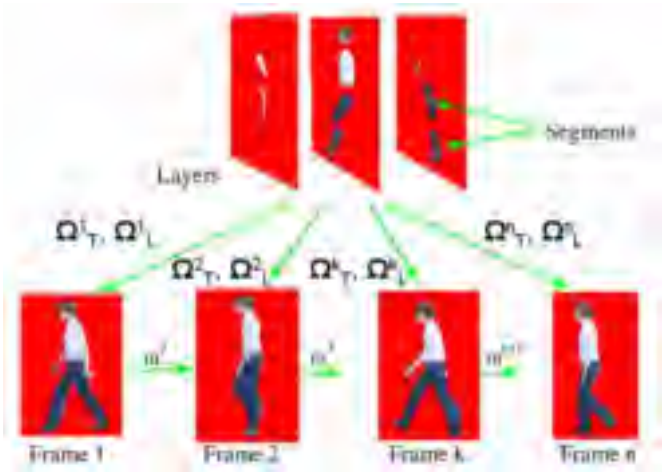


Figure 4.2: *The top row shows the various layers of a human model (the* latent image *in this case). Each layer consists of one of more segments whose appearance is shown. The shape of each segment is represented by a binary matte (not shown in the image). Any frame j can be generated using this representation by assigning appropriate values to its parameters. Note that the background is not shown for the sake of clarity of the figure.*

We introduce the model for a layered representation which describes the scene as a composition of layers. Any frame of a video can be generated from our model by assigning appropriate values to its parameters as illustrated in Fig. 4.2. The parameters of the model define the *latent image* and describe how to generate the frames using the latent image (see table 4.1). The model also assigns a (conditional) probability of the frame being generated for a given value of the parameters.

Latent Image: The latent image is defined as follows. It consists of a set of n_S *segments*, which are 2D patterns (specified by their shape and appearance) along with their layering. The layering determines the occlusion ordering. Each layer contains a number of non-overlapping segments. We denote the i^{th} segment of the latent image as s_i. The shape of a segment s_i is modelled as a binary matte $\mathbf{\Omega}_{Mi}$ of size equal to the frame of the video. Let a_i denote the a^{th} point in the matte $\mathbf{\Omega}_{Mi}$ (where the subscript i indicates the segment s_i). For such a point a_i, $\mathbf{\Omega}_{Mi}(a_i) = 1$ if a_i belongs to segment s_i (denoted by $a_i \in s_i$) and $\mathbf{\Omega}_{Mi}(a_i) = 0$ otherwise.

The appearance $\mathbf{\Omega}_{Ai}(a_i)$ of the segment s_i is the RGB value of points $a_i \in s_i$. We denote the set of mattes and appearance parameters for all segments as $\mathbf{\Omega}_M$ and $\mathbf{\Omega}_A$ respectively. The distribution of the RGB values $\mathbf{\Omega}_{Ai}(a_i)$ for all points

Input	
$\mathbf{D}$	Data (RGB values of all pixels in every frame of a video).
$\mathbf{D}^j$	The j^{th} frame of the video $\mathbf{D}$.
n_F	Number of frames.
Latent Image	
n_S	Number of segments s_i including the background.
$\mathbf{\Omega}_{Mi}$	Matte for segment s_i.
$\mathbf{\Omega}_M$	Set of all mattes, i.e. $\{\mathbf{\Omega}_{Mi}, i = 1, \cdots, n_S\}$.
$\mathbf{\Omega}_{Ai}$	Appearance parameter for segment s_i.
$\mathbf{\Omega}_A$	Set of all appearance parameters, i.e. $\{\mathbf{\Omega}_{Ai}, i = 1, \cdots, n_S\}$.
$\mathcal{H}_i$	Histogram specifying the distribution of the RGB values for s_i.
$\mathcal{H}$	Set of all histograms, i.e. $\{\mathcal{H}_i, i = 1, \cdots, n_S\}$.
o_i	Occlusion number of segment s_i.
$\mathbf{o}$	Set of all occlusion numbers, i.e. $\{o_i, i = 1, \cdots, n_S\}$.
Other Parameters	
$\mathbf{\Omega}_{Ti}^j$	Transformation $\{t_x, t_y, s_x, s_y, \phi\}$ of segment s_i to frame j.
$\mathbf{\Omega}_T$	Set of transformations for all segments s_i and frames j.
$\mathbf{\Omega}_{Li}^j$	Lighting variables $\{\mathbf{W}_i^j, \mathbf{w}_i^j\}$ of segment s_i to frame j.
$\mathbf{\Omega}_L$	Set of lighting variables for all segments s_i and frames j.
$\mathbf{\Omega}$	$\{n_S, \mathbf{\Omega}_M, \mathbf{\Omega}_A, \mathcal{H}, \mathbf{o}; \mathbf{\Omega}_T, \mathbf{\Omega}_L\}$.

Table 4.1: *Parameters of the layered representation.*

$a_i \in s_i$ is specified using a histogram $\mathcal{H}_i$ for each segment s_i. The set of all such histograms (i.e. for all segments s_i) is represented as $\mathcal{H}$. In order to model the layers, we assign a occlusion number o_i to each segment s_i such that segments belonging to the same layer share a common occlusion number. Each segment s_i can partially or completely occlude segment s_k if, and only if, $o_i > o_k$. In summary, the latent image is defined by the mattes $\mathbf{\Omega}_M$, the appearance $\mathbf{\Omega}_A$, the histograms $\mathcal{H}$ and the occlusion numbers $\mathbf{o} = \{o_i, i = 1, \cdots, n_S\}$ of the n_S segments.

Transformations and Lighting Parameters: When generating frame j, we start from a latent image and map each point $a_i \in s_i$ to a point a' in the frame using the transformation $\mathbf{\Omega}_{Ti}^j$. Note that we have assumed that points belonging to the same segment move according to a common transformation. The generated frame is then obtained by compositing the transformed segments in descending order of their occlusion numbers. For this work, each transformation has five degrees of freedom: one rotation, two translations (in x and y directions) and two anisotropic scale factors.

The model accounts for the effects of lighting conditions on the appearance of a

segment s_i using parameter $\mathbf{\Omega}_{Li}^j = \{\mathbf{W}_i^j, \mathbf{w}_i^j\}$, where $\mathbf{W}_i^j$ is a 3×3 diagonal matrix and $\mathbf{w}_i^j$ is a 3-dimensional vector. The change in appearance of the segment s_i in frame j due to lighting conditions is modelled as a linear transformation of $\mathbf{\Omega}_{Ai}(a_i)$, i.e.

$$\mathbf{d}(a') = \mathbf{W}_i^j \cdot \mathbf{\Omega}_{Ai}(a_i) + \mathbf{w}_i^j. \tag{4.2.1}$$

The motion of segment s_i from frame $j-1$ to frame j, denoted by $\mathbf{m}_i^j$, can be determined using the transformations $\mathbf{\Omega}_{Ti}^{j-1}$ and $\mathbf{\Omega}_{Ti}^j$. This allows us to take into account the change in appearance due to motion blur as

$$\mathbf{g}_i^j(a') = \int_0^T \mathbf{d}(a' - \mathbf{m}_i^j(t))dt, \tag{4.2.2}$$

where T is the total exposure time when capturing the frame.

To summarize, the layered representation model is given by the parameters $\{n_S, \mathbf{\Omega}_M, \mathbf{\Omega}_A, \mathcal{H}, \mathbf{o}; \mathbf{\Omega}_T, \mathbf{\Omega}_L\}$, where $\{n_S, \mathbf{\Omega}_M, \mathbf{\Omega}_A, \mathcal{H}, \mathbf{o}\}$ define the latent image, $\mathbf{\Omega}_T$ is the set of transformation parameters and $\mathbf{\Omega}_L$ represents the lighting parameters (see table 4.1). We denote the set of all parameters of the model by $\mathbf{\Omega}$.

Posterior of the model: The layered representation described above is a discriminative model. In other words, it specifies the conditional probability of the parameters $\mathbf{\Omega}$ given the video $\mathbf{D}$ which can be modelled using the conditional random field (CRF) formulation (described in § 2.2.2). However, there are two disadvantages is using the CRF framework directly: (i) similar to the MRF model, the CRF describes the probability of a labelling (i.e. a given set of values for the parameter $\mathbf{\Omega}$) using only one type of pairwise potential. In other words, it does not separate the data dependent and data independent parts of the pairwise potential; and (ii) its graphical model representation is the same as that of an MRF. These similarities between the MRF and CRF models make it hard to judge whether a given probabilistic model is generative or discriminative. In order to make the distinction between these models more explicit, we will find it convenient to use an alternative representation of CRF. This alternative representation, which we call the Contrast dependent random field (CDRF), is explained below.

The CDRF formulation models the posterior probability of $\mathbf{\Omega}$ given data $\mathbf{D}$ as

$$\Pr(\mathbf{\Omega}|\mathbf{D}, \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \exp(-Q(\mathbf{\Omega}; \mathbf{D}, \boldsymbol{\theta})), \tag{4.2.3}$$

where $\boldsymbol{\theta}$ is the set of potentials which define the CDRF and $Z(\boldsymbol{\theta})$ is the partition function. The energy $Q(\mathbf{\Omega}; \mathbf{D}, \boldsymbol{\theta})$ has the form

$$Q(\mathbf{\Omega}; \mathbf{D}, \boldsymbol{\theta}) = \sum_{i=1}^{n_S} \left(\sum_{a_i \in \mathbf{\Omega}_{Mi}} \theta^1_{a_i; \mathbf{\Omega}_{Mi}(a_i)} + \sum_{(a_i,b_k)\in\mathcal{E}} \theta^p_{a_i,b_k; \mathbf{\Omega}_{Mi}(a_i), \mathbf{\Omega}_{Mk}(b_k)} + \theta^c_{a_i,b_k; \mathbf{\Omega}_{Mi}(a_i), \mathbf{\Omega}_{Mk}(b_k)} \right), \tag{4.2.4}$$

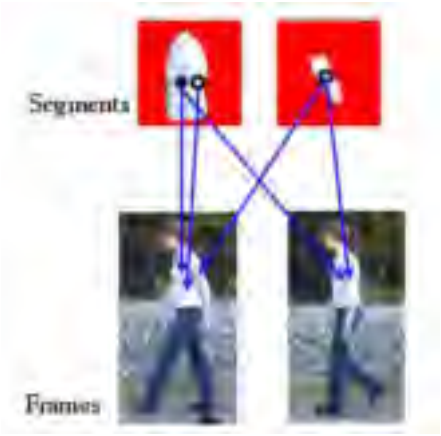


Figure 4.3: *The top row shows two segments of the human model. The unfilled circles represent two of the neighbouring points of the filled circle. The neighbourhood is defined across all mattes. We show one neighbour which lies on the same matte (i.e. the torso) and another neighbour which lies on a different matte (i.e. the upper arm). The bottom row shows two frames of the video along with the projections of the points on the segments. Note that the neighbouring point on the torso is occluded by the neighbouring point on the upper arm in the second frame.*

where $\mathcal{E}$ defines the neighbourhood of the CDRF. For this work, we define the neighbourhood of a_i as its 8-neighbourhood across all mattes $\mathbf{\Omega}_{Mi}$ of the layered representation (see Fig 4.3). As will be seen in § 4.3.3, this allows us to learn the model efficiently by minimizing the energy $Q(\mathbf{\Omega}; \mathbf{D}, \boldsymbol{\theta})$ using graph cuts. However, a larger neighbourhood can be used for each point at the cost of more computation time. Note that minimizing the energy $Q(\mathbf{\Omega}; \mathbf{D}, \boldsymbol{\theta})$ is equivalent to maximizing the posterior $\Pr(\mathbf{\Omega}|\mathbf{D}, \boldsymbol{\theta})$ since the partition function $Z(\boldsymbol{\theta})$ is independent of $\mathbf{\Omega}$.

Unlike the CRF formulation which uses two types of potentials (unary and pairwise), the energy of the CDRF model for the layered representation has three components: (i) the unary potential $\theta^1_{a_i;\mathbf{\Omega}_{Mi}(a_i)}$; (ii) the data independent prior term $\theta^p_{a_i,b_k;\mathbf{\Omega}_{Mi}(a_i),\mathbf{\Omega}_{Mk}(b_k)}$; and (iii) the data dependent contrast term $\theta^c_{a_i,b_k;\mathbf{\Omega}_{Mi}(a_i),\mathbf{\Omega}_{Mk}(b_k)}$. The unary potential measures the consistency of motion and colour distribution of the images of the point a_i. The prior and contrast terms encourage spatially continuous segments whose boundaries lie on edges in the frames. The graphical model representation of the CDRF is shown in Fig. 4.4. We note that unlike MRF it is not straightforward to generate the frames from CDRF since it is a discriminative model (due to the presence of contrast term) [7, 56]. Below, we specify the exact form of the potentials of the CDRF.

Unary Potential: We denote the observed RGB values at point $a' = \mathbf{\Omega}^j_{Ti}(a_i)$ (i.e. the image of the point a_i in frame j) by $\mathcal{I}^j(a_i)$. The generated RGB values of the point a' are described in equation (4.2.2). The unary potentials for a point

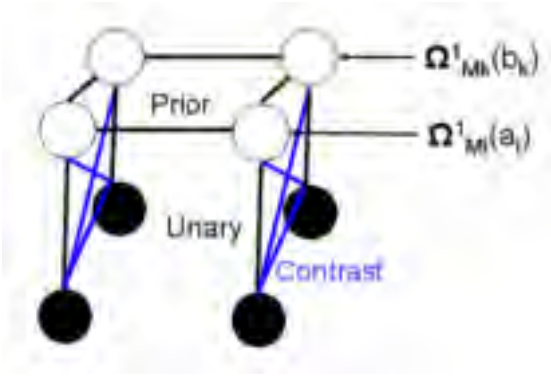


Figure 4.4: *Contrast-dependent random field (*CDRF*) formulation of the layered representation containing two segments s_i and s_k. The observed data is shown as filled circles while the points on the segment mattes are shown as unfilled circles. The unary potential $\theta^1_{a_i;\mathbf{\Omega}_{Mi}(a_i)}$ connects the data $\mathbf{D}$ (specifically the set of* RGB *values $\mathcal{I}^j(a_i)$) with the matte $\mathbf{\Omega}_{Mi}(a_i)$. The prior term $\theta^p_{a_i,b_k;\mathbf{\Omega}_{Mi}(a_i),\mathbf{\Omega}_{Mk}(b_k)}$ between two neighbouring points a_i and b_k, which encourages spatial continuity, connects $\mathbf{\Omega}_{Mi}(a_i)$ and $\mathbf{\Omega}_{Mk}(b_k)$. The data dependent contrast term $\theta^c_{a_i,b_k;\mathbf{\Omega}_{Mi}(a_i),\mathbf{\Omega}_{Mk}(b_k)}$ which cannot be included in the prior, is shown using the diagonal connections and the connections between observed nodes (in blue). Extending the formulation to more than two segments is trivial. Note that some of the connections for the contrast term are not shown for the sake of clarity of the figure.*

a_i are given by

$$\theta^1_{a_i;\mathbf{\Omega}_{Mi}(a_i)} = \begin{cases} \sum_{j=1}^{j=n_F} -\log(\Pr(\mathcal{I}^j(a_i)|\mathbf{\Omega}) & \text{if} \quad a_i \in s_i, \\ c_1 & \text{otherwise,} \end{cases} \tag{4.2.5}$$

where c_1 is a constant penalty assigned to points a_i which do not belong to the segment s_i. This penalty c_1 is necessary to avoid trivial solutions (i.e. $\mathbf{\Omega}_{Mi}(a_i) = 0$ for all s_i and a_i). For a point $a_i \in s_i$, i.e. $\mathbf{\Omega}_{Mi}(a_i) = 1$, the likelihood of $\mathcal{I}^j(a_i)$ is composed of two factors: (i) consistency with the colour distribution of the segment, which is the conditional probability of $\mathcal{I}^j(a_i)$ given $a_i \in s_i$, and is computed using histogram $\mathcal{H}_i$; and (ii) consistency of motion which measures how well the generated RGB values $\mathbf{g}^j_i(a')$ match the observed values $\mathcal{I}^j(a_i)$ (i.e. how well does the latent image project into the frames). These factors are chosen as

$$\Pr(\mathcal{I}^j(a_i)|\mathbf{\Omega}) \propto \Pr(\mathcal{I}^j(a_i)|\mathcal{H}_i) \exp(-\mu(\mathbf{c}^j_i(a') - \mathcal{I}^j(a_i))^2), \tag{4.2.6}$$

where μ is some scaling factor. We use $\mu = 1$ in our experiments. Note that in the above equation we assume that the point a_i is visible in frame j. When a_i is occluded by some other segment, we assign

$$\Pr(\mathcal{I}^j(a_i)|\mathbf{\Omega}) = c_2. \tag{4.2.7}$$

One might argue that motion consistency alone would provide sufficient discrimination between different segments. However, consider a video sequence with

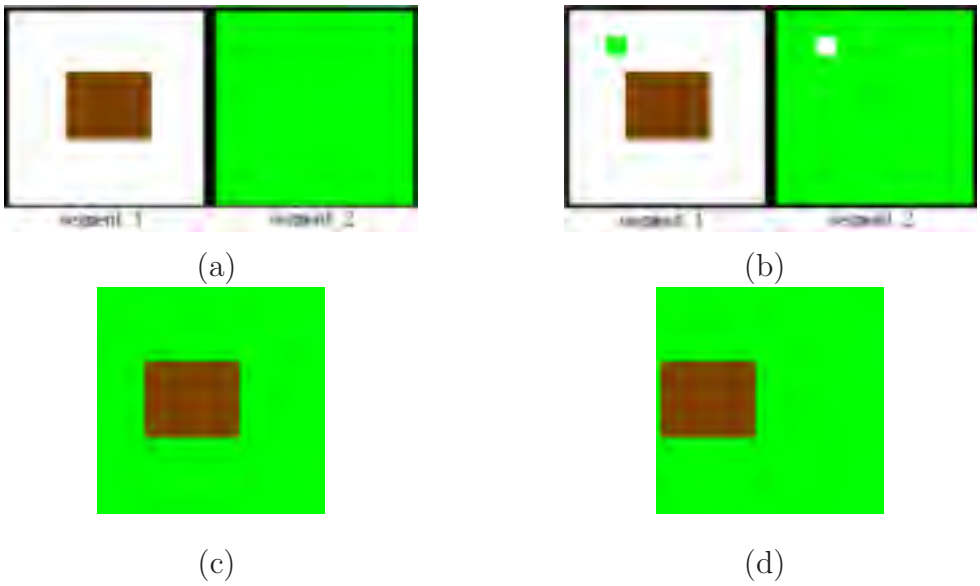


Figure 4.5: **(a)-(b)** *Two possible latent images of the model consisting of two segments. Unlike (a), the latent image shown in (b) assigns a block of green points to the first segment of the model.* **(c)-(d)** *Two frames of the video sequences. The brown region translates to the left while the green region remains stationary. Note that when only consistency of motion is used in equation (4.2.6), the unary potential* $\theta^1_{a_i;\mathbf{\Omega}_{Mi}(a_i)}$ *for* $a_i \in s_i$ *remains the same for both the latent images. This is because the green block provides the same match scores for both transformations (i.e. stationary and translation to left) since it maps onto green pixels in the frames. However, when consistency of appearance is also used this green block would be more likely to belong to the second segment (which is entirely green) instead of the first segment (which is mostly brown). Hence, the unary potential would favour the first latent image.*

homogeneous background (e.g. brown horse walking on green grass). In such cases, motion consistency would not be able to distinguish between assigning some blocks of the background (i.e. the green grass) to either the horse segment or the grass segment. Fig. 4.5 shows such a scenario where a block of the green background is assigned to different segments. However, the colour distribution of each segment would provide better discrimination. For example, the likelihood of a green point belonging to the first (mostly brown) segment would be low according to the colour distribution of that segment. Empirically, we found that using both motion and colour consistency provide much better results than using either of them alone.

Prior and Contrast: As noted above, the prior and contrast terms should be chosen to encourage spatially continuous segments whose boundaries lie on image edges. To this end, we closely follow the work of [13] on interactive binary image

Point a_i	Point b_k	Segments	Prior
$a_i \notin s_i$	$b_k \notin s_k$	$i \neq k$	κ_1
$a_i \notin s_i$	$b_k \notin s_k$	$i = k$	κ_1
$a_i \notin s_i$	$b_k \in s_k$	$i \neq k$	κ_1
$a_i \notin s_i$	$b_k \in s_k$	$i = k$	κ_2
$a_i \in s_i$	$b_k \notin s_k$	$i \neq k$	κ_1
$a_i \in s_i$	$b_k \notin s_k$	$i = k$	κ_2
$a_i \in s_i$	$b_k \in s_k$	$i \neq k$	κ_2
$a_i \in s_i$	$b_k \in s_k$	$i = k$	κ_1

Table 4.2: The value of the prior term $\theta^p_{a_i,b_k;\mathbf{\Omega}_{Mi}(a_i),\mathbf{\Omega}_{Mk}(b_k)}$ for various cases. The first column shows whether $a_i \in s_i$ or not. Similarly, the second column shows whether $b_k \in s_k$ or not. In the third column, $i = k$ implies that both points a_i and b_k belong to the same matte. The fourth column lists the value of the prior term corresponding to each case. Note that κ_1 and κ_2 are constants such that $\kappa_1 < \kappa_2$.

segmentation which was described in chapter 2. For clarity, we first describe the two terms separately while noting that their effect should be understood together. We subsequently describe their joint behaviour.

Prior Term: The prior term is specified by a Potts model as shown in table 4.2. Here κ_1 and κ_2 are constants such that $\kappa_1 < \kappa_2$. In other words, the prior term encourages spatial continuity by assigning a constant penalty to any pair of neighbouring pixels a_i and b_k which do not belong to the same segment.

Contrast Term: The contrast term encourages the projection of the boundary between segments to lie on image edges. Its value for various cases is shown in table 4.3. The term $\gamma_{ik}(a_i, b_k)$ is chosen such that it has a large value when a_i and b_k project onto image edges. Following [13], we use

$$\gamma_{ik}(a_i, b_k) = \lambda \left(1 - \exp\left(\frac{-\Delta^2_{ik}(a_i, b_k)}{2\sigma^2}\right) \cdot \frac{1}{\texttt{dist}(a_i, b_k)}\right), \tag{4.2.8}$$

where

$$\Delta_{ik}(a_i, b_k) = \frac{1}{n_F} \sum_{j=1}^{n_F} |\mathcal{I}^j(a_i) - \mathcal{I}^j(b_k)|. \tag{4.2.9}$$

measures the difference between the RGB values $\mathcal{I}^j(a_i)$ and $\mathcal{I}^j(b_k)$ throughout the video sequence. The term $\texttt{dist}(a_i, b_k)$, i.e. the Euclidean distance between a_i and b_k, gives more weight to the 4-neighbourhood of a_i than the rest of the 8-neighbourhood. Similar to the interactive binary image segmentation example (see § 2.2.2), the value of σ in equation (4.2.8) determines how the energy $Q(\mathbf{\Omega}; \mathbf{D}, \boldsymbol{\theta})$ is penalized since the penalty is high when $\Delta_{ik}(a_i, b_k) < \sigma$ and small

Point a_i	Point b_k	Segments	Contrast
$a_i \notin s_i$	$b_k \notin s_k$	$i \neq k$	0
$a_i \notin s_i$	$b_k \notin s_k$	$i = k$	0
$a_i \notin s_i$	$b_k \in s_k$	$i \neq k$	0
$a_i \notin s_i$	$b_k \in s_k$	$i = k$	$-\gamma_{ik}(a_i, b_k)$
$a_i \in s_i$	$b_k \notin s_k$	$i \neq k$	0
$a_i \in s_i$	$b_k \notin s_k$	$i = k$	$-\gamma_{ik}(a_i, b_k)$
$a_i \in s_i$	$b_k \in s_k$	$i \neq k$	$-\gamma_{ik}(a_i, b_k)$
$a_i \in s_i$	$b_k \in s_k$	$i = k$	0

Table 4.3: The value of the contrast term $\theta^c_{a_i,b_k;\mathbf{\Omega}_{Mi}(a_i),\mathbf{\Omega}_{Mk}(b_k)}$ for various cases. The first column shows whether $a_i \in s_i$ or not. Similarly, the second column shows whether $b_k \in s_k$ or not. In the third column, $i = k$ implies that both points a_i and b_k belong to the same matte. The fourth column lists the value of the contrast term corresponding to each case. The function $\gamma_{ik}(\cdot, \cdot)$ is as defined in equation (4.2.8).

when $\Delta_{ik}(a_i, b_k) > \sigma$. Thus σ should be sufficiently large to allow for the variation in RGB values within a segment.

Joint Behaviour of Prior and Contrast: In order to understand the joint behaviour of the prior and contrast terms, we model the energy of the layered representation in equation (4.2.4) using a CRF with parameter $\hat{\boldsymbol{\theta}}$ which is defined as follows:

$$\hat{\theta}^1_{a_i;\mathbf{\Omega}_{Mi}(a_i)} = \theta^1_{a_i;\mathbf{\Omega}_{Mi}(a_i)} \tag{4.2.10}$$

and

$$\hat{\theta}^2_{a_i,b_k;\mathbf{\Omega}_{Mi}(a_i),\mathbf{\Omega}_{Mk}(b_k)} = \theta^p_{a_i,b_k;\mathbf{\Omega}_{Mi}(a_i),\mathbf{\Omega}_{Mk}(b_k)} + \theta^c_{a_i,b_k;\mathbf{\Omega}_{Mi}(a_i),\mathbf{\Omega}_{Mk}(b_k)} \tag{4.2.11}$$

Note that, for every CDRF, there would exist an equivalent CRF whose pairwise potentials are the sum of the prior and contrast terms of the CDRF. Similar to the example CRF in § 2.2.2, we specify the pairwise potentials using the following weight values in all our experiments: $\kappa_1 = 1$, $\kappa_2 = 2.2$, $\lambda = 1$ and $\sigma = 5$. Table 4.4 shows the values of the prior and contrast term (as well as the pairwise potential of the equivalent CRF) for two neighbouring points a_i and b_k for these weight values. We consider two cases for the term $\Delta_{ik}(a_i, b_k)$: (i) $\Delta_{ik}(a_i, b_k) = \sigma/3$, i.e. points a_i and b_k have similar appearance; (ii) $\Delta_{ik}(a_i, b_k) = 3\sigma$, which implies points a_i and b_k have different appearance (as is the case when boundaries of segments lie on image edges). Note that the value of the pairwise potentials is small when the boundary of the segments lies on image edges (i.e. when $i \neq k$ and $\Delta_{ik}(a_i, b_k) = 3\sigma$). Hence, the weight values of our choice meet the desired criterion for the pairwise potentials of the energy of the layered representation.

Point a_i	Point b_k	Segments	$\Delta_{ik}(a_i, b_k)$	Prior	Contrast	Pairwise Pot.
$a_i \notin s_i$	$b_k \notin s_k$	$i \neq k$	$\sigma/3$	1	0	1
$a_i \notin s_i$	$b_k \notin s_k$	$i \neq k$	3σ	1	0	1
$a_i \notin s_i$	$b_k \notin s_k$	$i = k$	$\sigma/3$	1	0	1
$a_i \notin s_i$	$b_k \notin s_k$	$i = k$	3σ	1	0	1
$a_i \notin s_i$	$b_k \in s_k$	$i \neq k$	$\sigma/3$	1	0	1
$a_i \notin s_i$	$b_k \in s_k$	$i \neq k$	3σ	1	0	1
$a_i \notin s_i$	$b_k \in s_k$	$i = k$	$\sigma/3$	2.2	-0.0540	2.1460
$a_i \notin s_i$	$b_k \in s_k$	$i = k$	3σ	2.2	-0.9889	1.2111
$a_i \in s_i$	$b_k \notin s_k$	$i \neq k$	$\sigma/3$	1	0	1
$a_i \in s_i$	$b_k \notin s_k$	$i \neq k$	3σ	1	0	1
$a_i \in s_i$	$b_k \notin s_k$	$i = k$	$\sigma/3$	2.2	-0.0540	2.1460
$a_i \in s_i$	$b_k \notin s_k$	$i = k$	3σ	2.2	-0.9889	1.2111
$a_i \in s_i$	$b_k \in s_k$	$i \neq k$	$\sigma/3$	2.2	-0.0540	2.1460
$a_i \in s_i$	$b_k \in s_k$	$i \neq k$	3σ	2.2	-0.9889	1.2111
$a_i \in s_i$	$b_k \in s_k$	$i = k$	$\sigma/3$	1	0	1
$a_i \in s_i$	$b_k \in s_k$	$i = k$	σ	1	0	1

Table 4.4: *The prior and contrast terms for points a_i and b_k. The first two columns specify whether $a_i \in s_i$ and $b_k \in s_k$ or not respectively. In the third column, $i = k$ implies that both a_i and b_k belong to the same matte. The fourth column lists the value of $\Delta_{ik}(a_i, b_k)$. The fifth and sixth columns show the value of the prior and the contrast terms respectively (computed using tables 4.2 and 4.3). The pairwise potential of the equivalent* CRF *model is shown in the seventh column and is computed as the sum of the corresponding elements in the fifth and sixth columns.*

Empirically, they were found to provide good segmentations for all our input videos.

Note that, for two points a_i and b_k, the minimum penalty of assigning them to different segments, i.e. $\min(\kappa_2 - \gamma_{ik}(a_i, b_k)) - \kappa_1$, depends only on the value of κ_1 and κ_2 (since $\gamma_{ik}(a_i, b_k)$ is always less than 1). Unlike the CRF framework, this fact comes across clearly in the CDRF formulation which forces us to treat the prior and the contrast term separately. In our case the values of κ_1 and κ_2 are chosen such that the minimum value of this penalty is always greater than 0.2. This prevents speckles appearing in the estimated segments by encouraging contiguous regions (i.e. regions which minimize the length of the boundary between segments). For example, consider the case where a_i differs in appearance from all its neighbours due to noise in the video frames. It would be undesirable to assign a_i to a different segment from all its neighbours. Such an assignment would be discouraged since a_i would have to incur a penalty of at least 0.2 from all its neighbours.

The next section builds on previous inference algorithms to learn the layered

representation. To this end, we describe a five stage approach to obtain $\mathbf{\Omega}$ given data $\mathbf{D}$ by minimizing the energy $Q(\mathbf{\Omega}; \mathbf{D}, \boldsymbol{\theta})$ (i.e. maximizing $\Pr(\mathbf{\Omega}|\mathbf{D}, \boldsymbol{\theta})$). The method described is applicable to any scene with piecewise parametric motion.

4.3. Learning Layered Segmentation

1. Rigidly moving components are identified between every pair of consecutive frames (§ 4.3.1). This is achieved by computing the image motion using a *novel efficient sum-product* BP *algorithm.*
2. An initial estimate of $\mathbf{\Omega}$ is obtained by combining these components (§ 4.3.2). The number of segments n_S is fixed to the initial estimate.
3. The parameters $\mathbf{\Omega}_A$, $\mathbf{\Omega}_T$ and $\mathbf{\Omega}_L$ are kept constant and the mattes $\mathbf{\Omega}_M$ are optimized *using $\alpha\beta$-swap and α-expansion algorithms* [15]. The occlusion numbers o_i are obtained (§ 4.3.3).
4. Using the refined values of $\mathbf{\Omega}_M$, the appearance parameters $\mathbf{\Omega}_A$ are updated. (§ 4.3.4).
5. Finally, the transformations $\mathbf{\Omega}_T$ and lighting variables $\mathbf{\Omega}_L$ are re-estimated, keeping $\mathbf{\Omega}_M$ and $\mathbf{\Omega}_A$ unchanged (§ 4.3.5).

Algorithm 1: *Estimating the parameters of the layered representation.*

Given a video, our objective is to estimate $\mathbf{\Omega}$ (i.e. the latent image, the transformations and the lighting variables) of the layered representation. Our approach takes inspiration from the highly successful interactive image segmentation algorithm of Boykov and Jolly [13] in which the user provides a small number of foreground and background seed pixels. The appearance model learnt from these seed pixels then provides sufficient information to obtain reliable segmentation by minimizing an objective function similar to equation (4.2.4) (see § 2.2.2). In our case, the seed pixels are provided by a rough motion segmentation obtained by computing the image motion. These seed pixels are sufficient to bootstrap the method to minimize equation (4.2.4) to obtain reliable segmentations. *This is one of the key intuitions behind our method.*

We obtain the layered representation $\mathbf{\Omega}$ in five stages. In the first stage, image motion is computed between every pair of consecutive frames to obtain rigidly moving *components.* An initial estimate of $\mathbf{\Omega}$ is then found in the second stage using these components. This provides us with the seed pixels for each segment. In the remaining stages, we alternate between holding some parameters constant and optimizing the rest as illustrated in algorithm 1.

Our method makes use of three inference algorithms for CRFs : sum-product BP, $\alpha\beta$-swap and α-expansion. Sum-product BP is particularly useful for appli-

cations such as estimating motion fields where each site of the CRF has a large number of labels (i.e. equal to the number of possible motions from one frame to the next). However, when refining the model, the number of labels is small (i.e. equal to the number of segments) and hence efficient inference can be performed using graph cuts based methods (i.e. $\alpha\beta$-swap and α-expansion). As will be seen, we take advantage of the strengths of all these algorithms. We begin by describing our approach for computing image motion.

4.3.1 Two Frame Motion Segmentation

In this section, we describe a novel, efficient algorithm to obtain rigidly moving *components* between a pair of frames by computing the image motion. Our method is robust to changes in appearance due to lighting and motion blur. We use the term *components* here to distinguish them from the *segments* finally found. Note that this is a simple two frame motion segmentation method that is used to initialize the more complex multiframe one as described later. The set of components obtained from all pairs of consecutive frames in a video sequence are combined to get the initial estimate of the segments (see § 4.3.2). Note that our method does not depend on any keyframe(s). This avoids the problems of previous approaches such as [94, 105], namely that they do not learn segments which were not present in the keyframe. For example, if the hand of a person walking was occluded by the torso in the keyframe then it would not be learnt as part of the layered representation using the methods described in [94, 105]. In contrast, our method will obtain the hand segment as long as it is visible in one of the frames of the video (and is moving non-rigidly with the other segments).

In order to identify points that move rigidly together from frame j to $j+1$ in the given video $\mathbf{D}$, we need to determine the transformation that maps each point in frame j to its position in frame $j+1$ (i.e. the image motion). However, at this stage we are only interested in obtaining a coarse estimate of the components as they will be refined later using graph cuts. This allows us to reduce the complexity of the problem by dividing frame j into uniform patches $\mathbf{r}_a$ of size $p \times p$ pixels and determining their transformations φ_i. However, a large value of p may merge two components. We use $p = 3$ for all our experiments which was found to offer a good compromise between efficiency and accuracy.

The components are obtained in two stages: (i) finding a set of putative transformations φ_i for each patch $\mathbf{r}_a$ in frame j; (ii) selecting from those initial transformations the best joint set of transformations over all patches in the frame. As the size of the patch is only 3×3 and we restrict ourselves to consecutive frames, it is sufficient to use transformations defined by a scale ρ_i, rotation ϕ_i and translation $\mathbf{t}_i = \{t_i^x, t_i^y\}$, i.e. $\varphi_i = \{\rho_i, \phi_i, \mathbf{t}_i\}$.

Finding putative transformations: We define a CRF whose variables correspond to the patches of frame j (i.e. each variable v_a of the CRF represents a patch $\mathbf{r}_a$). Each label l_i of variable v_a corresponds to a putative transformation φ_i. Note that this is a different CRF from the one described in the previous section which models the energy of the layered representation. It is a simpler one which we will solve in order to provide initialization for the layered representation.

In order to further differentiate this CRF from the one described in section 4.2, we represent its unary and pairwise potentials as $\psi^1_{a;i}$ and $\psi^2_{ab;ik}$ respectively. The unary potential $\psi^1_{a;i}$ of a label measures how well the patch $\mathbf{r}_a$ matches frame $j+1$ after undergoing transformation φ_i. The neighbourhood $\mathbf{N}_a$ of each variable v_a is defined as its 4-neighbourhood. As we are interested in finding rigidly moving components, we specify the pairwise potential $\psi^2_{ab;ik}$ such that it encourages neighbouring patches to move rigidly together. Let f represent a labelling of these variables (i.e. variable v_a takes label $l_{f(a)}$). The posterior probability of such a labelling given the j^{th} and $(j+1)^{th}$ frame (denoted by $\mathbf{D}^j$ and $\mathbf{D}^{j+1}$ respectively) is

$$\Pr(f|\mathbf{D}^j, \mathbf{D}^{j+1}, \boldsymbol{\psi}) = \frac{1}{Z(\boldsymbol{\psi})} \prod_k \exp(-\psi_{a;f(a)}) \prod_{v_b \in \mathbf{N}_a} \exp(-\psi_{ab;f(a)f(b)}), \qquad (4.3.1)$$

where $\boldsymbol{\psi}$ is the set of potentials and $Z(\boldsymbol{\psi})$ is the partition function of the CRF. Below, we describe the various terms used to define the above CRF in detail.

Transformations: By taking advantage of the fact that large scaling, translations and rotations are not expected between consecutive frames, we restrict ourselves to a small number of putative transformations. Specifically, we vary scale ρ_i from 0.7 to 1.3 in steps of 0.3, rotation ϕ_i from -0.3 to 0.3 radians in steps of 0.15 and translations $\mathbf{t}_i$ in horizontal (i.e. t^x_i) and vertical directions (i.e. t^y_i) from -10 to 10 pixels and -5 to 5 pixels respectively in steps of 1. Thus the total number of transformations is 3465.

Unary Potential: The unary potential associated with a patch $\mathbf{r}_a$ undergoing transformation φ_i is modelled as $\psi^1_{a;i} \propto -\mathcal{L}(\mathbf{r}_a, \varphi_i)$. The term $\mathcal{L}(\mathbf{r}_a, \varphi_i)$ is computed as follows. An $\omega \times \omega$ window centred around the patch $\mathbf{r}_a$ in frame j is transformed according to φ_i. The value of ω is chosen to be greater than the size of the patch (i.e. $\omega > p$). If φ_i is the true transformation of the patch $\mathbf{r}_a$ then the RGB values of the transformed $\omega \times \omega$ window should be similar to the RGB values of the corresponding pixels in frame $j+1$. The term $\mathcal{L}(\mathbf{r}_a, \varphi_i)$ measures this similarity between the window and the corresponding pixels of frame $j+1$ as their normalized cross-correlation. When calculating $\mathcal{L}(\mathbf{r}_a, \varphi_i)$ in this manner, the $\omega \times \omega$ window is subjected to different degrees of motion blur according to the motion specified by φ_i, and the best match score is chosen. *This, along with the use of normalized cross-correlation, makes the estimation of unary potential robust to lighting changes and motion blur.* In all our experiments, we used $\omega = 5$. Since the appearance of a patch does not change drastically between consecutive

frames, normalized cross-correlation provides reliable match scores. Unlike [54], we do not discard the transformations resulting in a low match score. However, it will be seen later that this does not significantly increase the amount of time required for finding the minimum mean squared error (MMSE) estimate of the transformations due to our computationally efficient method.

Pairwise Potential: We want to assign the pairwise potentials $\psi^2_{ab;ik}$ such that neighbouring patches $\mathbf{r}_a$ and $\mathbf{r}_b$ which do not move rigidly together are penalized. However, we would be willing to take the penalty when determining the MMSE estimate if it results in better match scores. Furthermore, we expect two patches separated by an edge to be more likely to move non-rigidly since they might belong to different segments. Thus we define the pairwise potentials by a Potts model such that

$$\psi^2_{ab;ik} = \begin{cases} -1 & \text{if rigid motion,} \\ -\zeta\nabla(\mathbf{r}_a, \mathbf{r}_b) & \text{otherwise,} \end{cases}$$

where $\nabla(\mathbf{r}_a, \mathbf{r}_b)$ is the average of the gradients of the neighbouring pixels $x \in \mathbf{r}_a$ and $y \in \mathbf{r}_b$, i.e. along the boundary shared by $\mathbf{r}_a$ and $\mathbf{r}_b$. The term ζ specifies how much penalty is assigned for two neighbouring patches not moving rigidly together. We choose ζ such that it scales $\zeta\nabla(\mathbf{r}_a, \mathbf{r}_b)$ to lie between 0 and 1.

Occlusion: To handle occlusion, an additional label l_o is introduced for each variable v_a which represents the patch $\mathbf{r}_a$ being occluded in frame $j+1$. The corresponding likelihoods and pairwise terms $\psi^1_{a;o}, \psi^2_{ab;io}, \psi^2_{ab;oi}$ and $\psi^2_{ab;oo}$ are modelled as constants for all patches $\mathbf{r}_a$ and $\mathbf{r}_b$. In our experiments, we used the values $0.1, 0.5, 0.5$ and 0.8 respectively. The higher value for $\psi^2_{ab;oo}$ specifies that two neighbouring patches tend to get occluded simultaneously.

Obtaining the transformations: Once the CRF has been specified, the best joint set of transformations for all patches is found as the (approximate) MMSE estimate defined by the probability $\Pr(f|\mathbf{D}^j, \mathbf{D}^{j+1}, \boldsymbol{\psi})$. We use the sum-product BP to find the posterior probability of a patch $\mathbf{r}_a$ undergoing transformation φ_i. This provides us with the MMSE estimate of transformations.

The two main limitations of sum-product BP are its large memory requirements and its computational inefficiency. We overcome these problems by developing a novel coarse to fine BP algorithm. This algorithm groups similar labels of the CRF to obtain a smaller number of *representative* labels, thereby reducing the memory requirements. The time complexity of sum-product BP is also reduced using the method described in [24]. Details of the algorithm can be found in Appendix A.

Once the MMSE estimate of the transformations for all the patches of frame j have been determined, we cluster the points moving rigidly together to obtain rigid components. Components with size less than 100 pixels are merged with surrounding components. We repeat this process for all pairs of consecutive

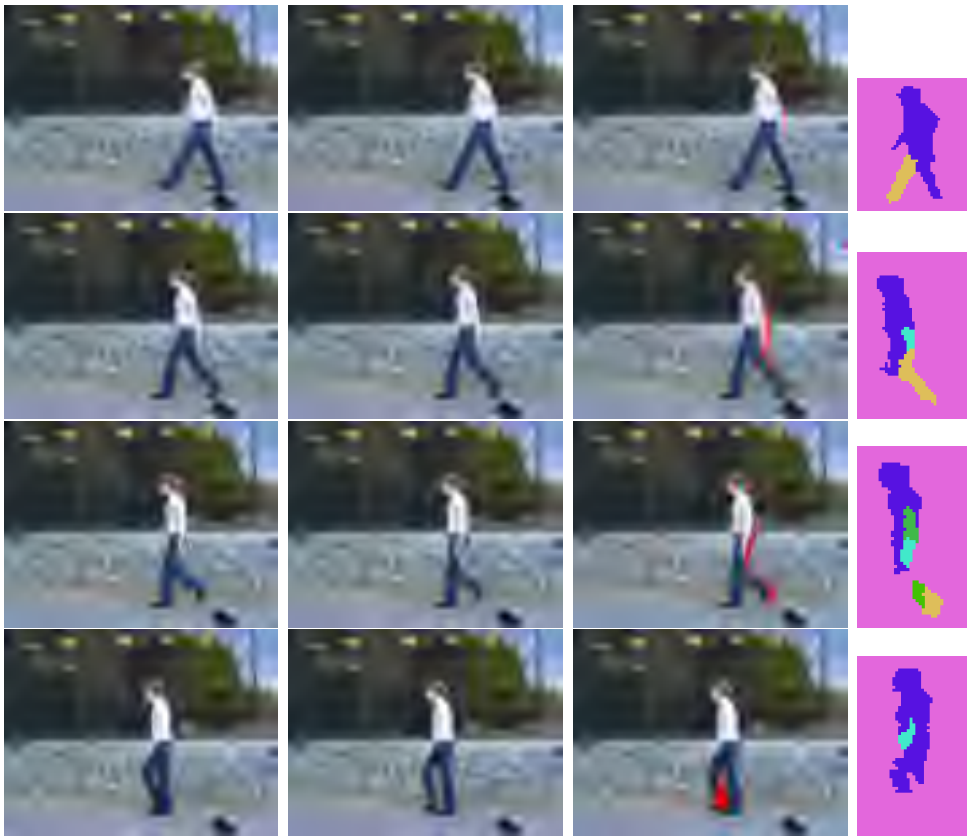


Figure 4.6: *Results of obtaining the* MMSE *estimates of the transformations. The first two columns show consecutive frames of a video. The third column shown the reconstruction of the second frame obtained by mapping the patches of the first frame according to the transformations obtained using coarse-to-fine efficient* LBP*. Points which are occluded from the first frame but are present in the second frame would be missing from the reconstruction. These points are shown in red. Points moving rigidly are clustered together to obtain the components shown in the fourth column.*

frames of the video. The m^{th} component of frame j is represented as a set of points $\mathcal{C}_m^j$. Fig. 4.6 shows the result of our approach on four pairs of consecutive frames. Fig. 4.7 shows the advantage of modelling motion blur when computing the unary potential of a patch $\mathbf{r}_a$ undergoing a transformation φ_i. In the next section, we describe how an initial estimate of the layered representation is obtained using the rigid pairwise components.

4.3.2 Initial Estimation of the Model over Multiple Frames

In this section, we describe a method to get an initial estimate of $\mathbf{\Omega}$. The method consists of two stages: (i) combining rigidly moving components to obtain the number of segments and the initial estimate of their shape parameter $\mathbf{\Omega}_{Mi}$; and

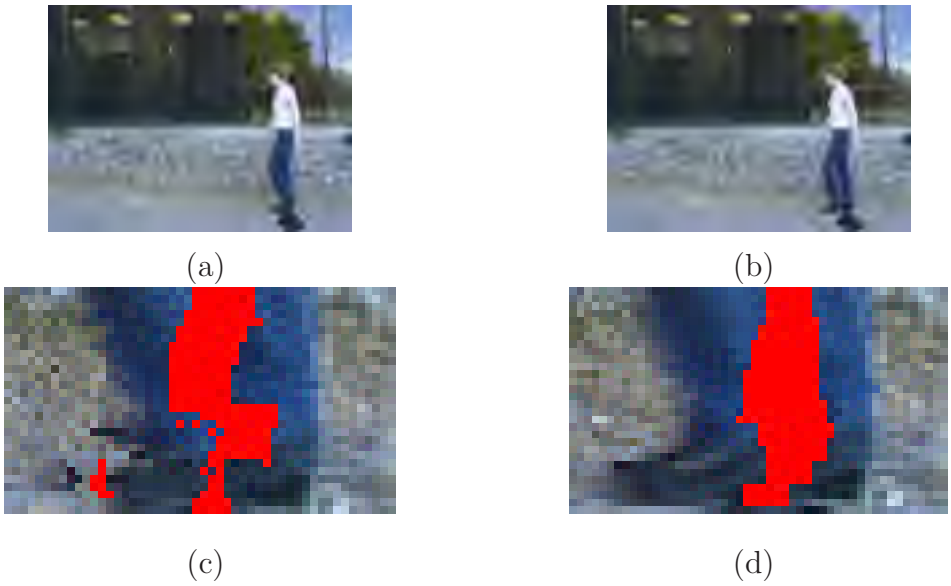
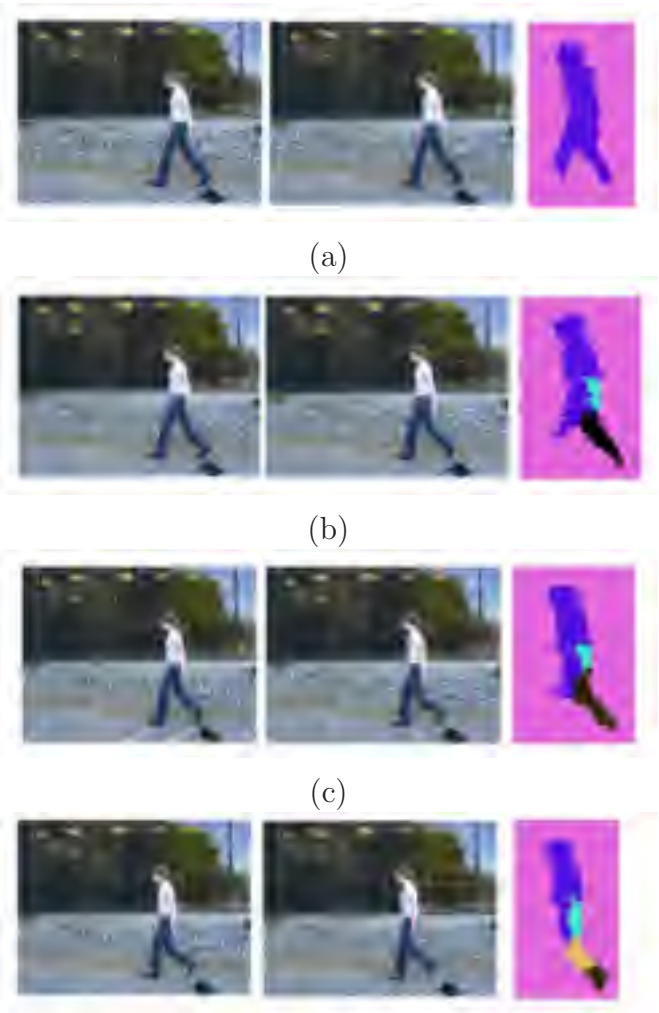


Figure 4.7: *Effects of modelling motion blur.* **(a)-(b)** *Two consecutive frames from the video shown in Fig. 4.1.* **(c)** *The image motion computed without modelling motion blur. The reconstruction of the second frame, obtained by mapping the patches of the first frame according to the transformations obtained, indicates that incorrect transformations are found around the feet (see e.g. the shoes) where there is considerable motion blur. Note that the pixels marked red are those that are occluded in the first frame but present in the second frame.* **(d)** *Results after modelling motion blur. Accurate transformations for the patches belonging to the shoes are obtained by accounting for the change in appearance due to motion blur.*

(ii) computing the remaining parameters, i.e. $\mathbf{\Omega}_{Ai}$, $\mathbf{\Omega}_{Ti}^{j}$ and $\mathbf{\Omega}_{Li}^{j}$.

Combining rigid components: Given the set of all pairwise components, we want to determine the number of segments n_S present in the entire video sequence and obtain an initial estimate of their shape. To this end, we make use of the MMSE estimate of the transformations, i.e. φ_i, obtained in the previous section. Note that, by definition, all the patches in frame j which belong to the same component $\mathcal{C}_m^j$ share the same transformation (since they move rigidly together). We denote the transformation of a component $\mathcal{C}_m^j$ (i.e. the transformation of its patches) by φ_m^j.

Note that each component may consists of points which belong to one of more segments. For example, consider the two components shown in the top row of Fig. 4.6. One of these components consists of the two half limb segments of a leg while the other component consists of the rest of the segments. However, we rely on each segment of the scene to be detected as an individual component in at least one frame. Empirically, this assumption is not found to be restrictive as it is satisfied for all the videos used in our experiments.



(a)

(b)

(c)

(d)

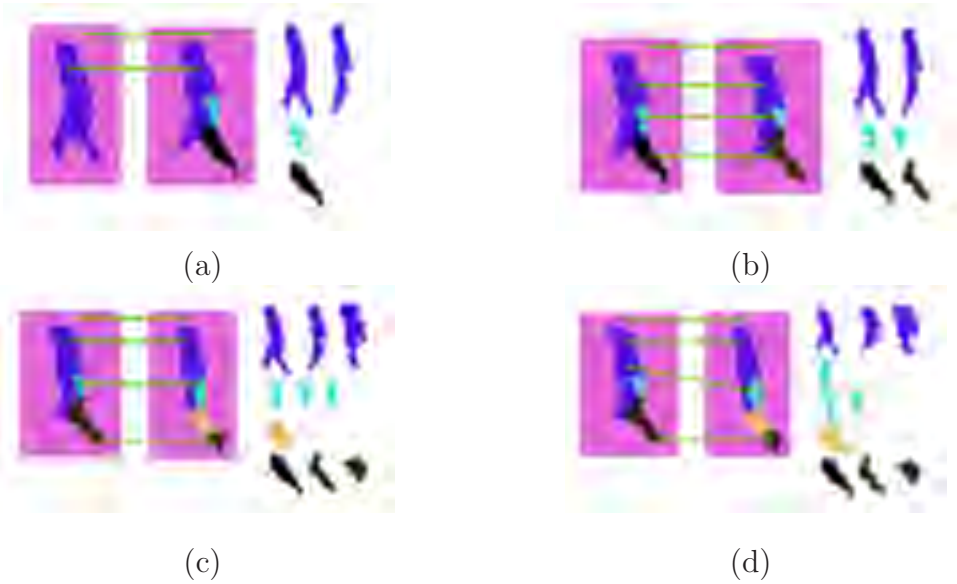
Figure 4.8: *Four sets of rigidly moving components found using five consecutive frames.* **(a)** *First two frames together with the set of components (i.e. the background and the body of the person).* **(b)-(d)** *Rigidly moving components for the remaining pairs of consecutive frames.*

Given the above assumption, one of the methods to obtain the segments would be to establish a correspondence between the components using the transformations φ_m^j. As an illustration, consider five consecutive frames of the video shown in Fig. 4.1. These frames result in four sets of components (i.e. one set each for four pairs of consecutive frames) as shown in Fig. 4.8. In order to establish a correspondence between these sets of components, we start with the first set (i.e. the set in Fig. 4.8(a) which contains only one component other than the background). We map this component using its transformation to the second frame which establishes a correspondence between the first two sets of components as shown in Fig. 4.9(a). This correspondences allows us to cluster the two sets of components such that each cluster contains corresponding components. The clusters obtained in this manner are also shown in Fig. 4.9(a).

We can continue this process by establishing a correspondence between the second and the third set of components. By assuming the correspondence to be transitive, we can obtain the clusters for the first three sets of components (see Fig. 4.9(b)). If one of the components in frame j, say $\mathcal{C}_m^j$, maps to more than one components in frame $j+1$ then we can split the cluster containing $\mathcal{C}_m^j$ as shown in Fig. 4.9(c). Similarly, if more than one components in frame j, say $\mathcal{C}_m^j$ and $\mathcal{C}_n^j$, correspond to one component in frame $j+1$, say $\mathcal{C}_p^{j+1}$, then we can simply let $\mathcal{C}_p^{j+1}$ to belong to one of the clusters (i.e. either belonging to the cluster containing $\mathcal{C}_m^j$ or the cluster containing $\mathcal{C}_n^j$). Hence, starting from the first frame and proceeding in this manner till the end of the video would result in clustering the components. The number of such clusters would provide us with the number of segments n_S. The initial estimate of the shape of a segment corresponding to a cluster can simply be chosen as the shape of one of the components belonging to that cluster.

However, the method described above does not take into account the errors that may be present in the transformations φ_m^j. Such errors would result in establishing an incorrect correspondence between two sets of components. This incorrect correspondence would in turn result in erroneous clustering of the components, e.g. see Fig. 4.9(d). Thus any method used for combining the components to obtain the segments should be robust to noise in the transformations φ_m^j. For this purpose, we use the agglomerative clustering algorithm as follows.

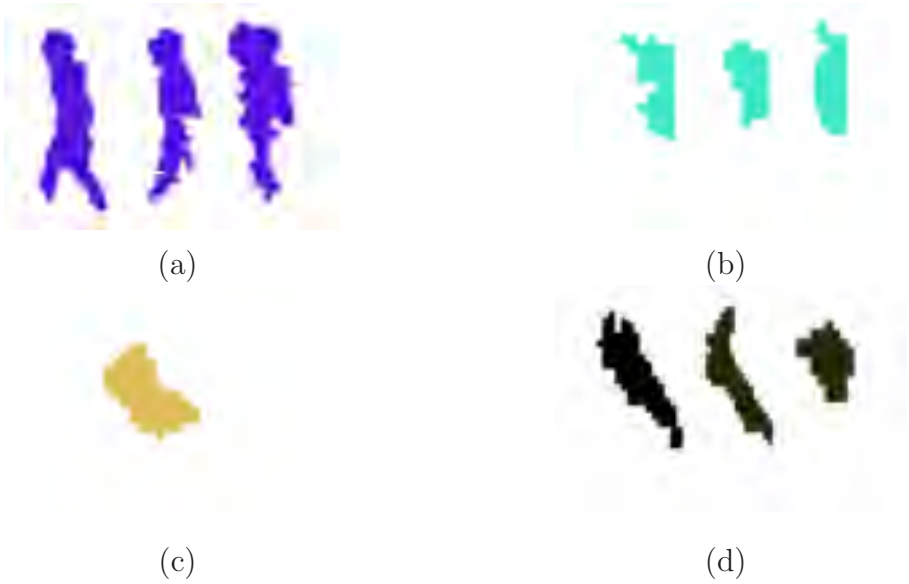
Agglomerative clustering starts by treating each component as a separate cluster. At each step, the two most *similar* clusters (i.e. the clusters containing the two most similar components) are combined together. The algorithm is terminated when the similarity of all pairs of components belonging to different clusters falls below a certain threshold. The similarity of two components, say $\mathcal{C}_m^j$ and $\mathcal{C}_n^k$, is measured as follows. If the corresponding component of $\mathcal{C}_m^j$ in frame k lies close to the component $\mathcal{C}_n^k$, then their similarity is measured as the normalized cross-correlation of the RGB values corresponding to these components. However, if the corresponding component of $\mathcal{C}_m^j$ lies far from $\mathcal{C}_n^k$, then their similarity mea-



(a) (b)

(c) (d)

Figure 4.9: *Correspondences are established between two consecutive sets of components by mapping a component $\mathcal{C}_m^j$ in frame j to frame $j+1$ using transformation φ_m^j. These correspondences are shown using green lines going from one set of components to the other. The corresponding components are said to belong to the same cluster.* **(a)** *The correspondence and the resulting clustering for the first two sets of components shown in Fig. 4.8. Each row in the right hand side represents one cluster. For example, the first row corresponds to the head+torso cluster, the second row corresponds to the arm cluster while the third tow corresponds to the leg cluster.* **(b)** *The correspondence and the resulting clustering for the second and third sets of components. The correspondence is considered transitive thereby resulting in a clustering of all three sets of components.* **(c)** *The correspondence and the resulting clustering for the third and fourth sets of components. Note that the leg cluster has been split into two half limb clusters.* **(d)** *Errors in estimating the transformations φ_m^j result in incorrect correspondences. Specifically, the arm of the person maps to the torso in the previous frame. Incorrect correspondence leads to erroneous clustering. For example, there are now two clusters corresponding to the arms of the person.*



(a) (b)

(c) (d)

Figure 4.10: *The clusters found using agglomerative clustering for the components shown in Fig. 4.8.* **(a)** *The cluster corresponding to the head+torso segment.* **(b)** *The cluster corresponding to the arm segment.* **(c)** *The cluster corresponding to the upper leg segment.* **(d)** *The cluster corresponding to the lower leg segment.*

sure is set to $-\infty$. The advantage of computing the similarity in this manner is two folds: (i) the clustering obtained makes efficient use of the correspondences provided by the transformations φ_m^j (computed as described above) while being robust to noise; and (ii) it encourages the agglomerative clustering algorithm to terminate quickly since the similarity measures between most pairs of components are $-\infty$.

Upon termination, the number of clusters provides us with the number of segments. The clusters obtained using agglomerative clustering for the set of components in Fig. 4.8 is shown in Fig. 4.10. We simply let components containing more than one segment lie in a cluster representing one of these segments. For example, the component in Fig. 4.8(b) which contains both the half limbs of a leg belongs to the cluster corresponding to the lower half limb in Fig. 4.10(d).

We are now only left with the decision of choosing one of the components in a cluster to represent the initial shape of the segment corresponding to that cluster. To this end, we choose the smallest component in the i^{th} cluster to specify the matte $\mathbf{\Omega}_{Mi}$ of segment s_i, as shown in Fig. 4.11 (top row). This avoids using a component containing more than one segment to define the shape of a segment. However, this implies that the initial estimate will often be smaller than (or equal to) the ground truth and thus, needs to be expanded as described in § 4.3.3.

Fig. 4.11 (bottom row) shows the initial shape estimates of all the segments (excluding the background) for the video in Fig. 4.1 obtained using our method. Note that all the segments of the person visible in the video have been found.

Figure 4.11: *Shape estimates of the segments found using agglomerative clustering. The top row shows the shape estimates corresponding to the clustering shown in Fig. 4.10. The bottom row shows the shape estimates for all the segments found in the entire video sequence shown in Fig. 4.1.*

Before moving further, we note here that the above method for combining rigid components to obtain segments is similar to the method described by Ramanan and Forsyth [74]. Note that [74] clusters rectangular fragments found in a video to obtain parts of an object. However, their method relies on finding parallel lines of contrast to define the fragments, which restricts their method to a small class of objects and videos. In contrast, our method obtains rigidly moving components by computing image motion and hence, is applicable to any video containing piecewise parametric motion.

Initial estimation of the model: Once the mattes $\mathbf{\Omega}_{Mi}$ are found, we need to determine the initial estimate of the remaining parameters of the model. The transformations $\mathbf{\Omega}_{Ti}^j$ are obtained using φ_m^j and the component clusters. The appearance parameter $\mathbf{\Omega}_{Ai}(a_i)$ is given by the mean of $\mathcal{I}^j(a_i)$ over all frames j. The histograms $\mathcal{H}_i$ are computed using the RGB values $\mathbf{\Omega}_{Ai}(a_i)$ for all points $a_i \in s_i$. As the size of the segment is small (and hence, the number of such RGB values is small), the histogram is implemented using only 15 bins each for R, G and B. The lighting parameters $\mathbf{\Omega}_i^j = \{\mathbf{W}_i^j, \mathbf{w}_i^j\}$ are calculated in a least squares manner using $\mathbf{\Omega}_{Ai}(a_i)$ and $\mathcal{I}^j(a_i)$, for all $a_i \in s_i$. The motion variables $\mathbf{m}_i^j$ are given by $\mathbf{\Omega}_{Ti}^j$ and $\mathbf{\Omega}_{Ti}^{j-1}$. This initial estimate of the model is then refined by optimizing each parameter while keeping others unchanged. We start by optimizing the shape parameters $\mathbf{\Omega}_M$ as described in the next section.

4.3.3 Refining Shape

In this section, we describe a method to refine the estimate of the shape parameters $\mathbf{\Omega}_M$ and determine the occlusion numbers o_i. Given an initial coarse estimate of the segments, we improve their shape using consistency of motion and

texture over the entire video sequence. The refinement is carried out such that it minimizes the energy $Q(\mathbf{\Omega}; \mathbf{D}, \boldsymbol{\theta})$ of the model given in equation (4.2.4).

There are several approaches that one can adopt to refine the shape parameters by minimizing the energy of the model. A standard way would be to fix the occlusion numbers of the segments and refine their shape by minimizing the resulting energy function. The process could be repeated for all possible assignments of occlusion numbers. The assignment which results in the lowest energy value would then be chosen to provide the refined estimates of $\mathbf{\Omega}_M$. However, this would be computationally inefficient because of the large number of possible occlusion numbers for each segment. In order to reduce the complexity of the algorithm, we make use of an iterative approach. As will be seen, this allows us to make use of the fact that only those segments which overlap with each other are required for determining the occlusion ordering (i.e. the layering of segments). In practice, we found that this iterative procedure provides a good estimate of the shape of the segments for all the videos used in this work.

Specifically, we make use of two algorithms: $\alpha\beta$-swap and α-expansion [15], which were briefly described in chapter 2. The $\alpha\beta$-swap algorithm iterates over pairs of segments, s_α and s_β. At each iteration, it refines the mattes of s_α and s_β by swapping the values of $\mathbf{\Omega}_{M\alpha}(a_\alpha)$ and $\mathbf{\Omega}_{M\beta}(a_\beta)$ for some points a_α and a_β in the mattes of the segments s_α and s_β respectively. The α-expansion algorithm iterates over segments s_α. At each iteration, it assigns $\mathbf{\Omega}_{M\alpha}(a_\alpha) = 1$ for some points a_α in the matte $\mathbf{\Omega}_{M\alpha}$. Note that α-expansion never reduces the number of points belonging to segment s_α.

In our previous work [54], we described an approach for refining the shape parameters of the model when all the segments are restricted to lie in one *reference* frame. In other words, each point on the reference frame has a unique label, i.e. it belongs to only one segment. In that case, it was sufficient to refine one segment at a time using the α-expansion algorithm alone to correctly relabel all the wrongly labelled points. For example, consider a point $a \in s_i$ which was wrongly labelled as belonging to s_k. During the expansion move where $\alpha = i$, the point a would be relabelled to s_i (and hence, it would not belong to s_k).

The question now arises as to why we need both the $\alpha\beta$-swap and the α-expansion algorithms for the layered representation? To answer this, we note that the shape of each segment in the layered representation is modelled using a separate matte (i.e. no reference frame is used). Hence the restriction that points in the reference frame belong to only one segment no longer holds true. In such a case, performing only α-expansion would incorrectly relabel the a^{th} point in both $\mathbf{\Omega}_{Mi}$ and $\mathbf{\Omega}_{Mk}$ to 1. We overcome this problem by performing $\alpha\beta$-swap over pairs of segments. During the swap move when $\alpha = i$ and $\beta = k$, the a^{th} point in matte $\mathbf{\Omega}_{Mi}$ (i.e. a_i) would be relabelled to 1 while the a^{th} point in $\mathbf{\Omega}_{Mk}$ (i.e. a_k) would be relabelled to 0. This case is illustrated in Fig. 4.12 using two segments corresponding to half limbs of the walking person in Fig. 4.1. The

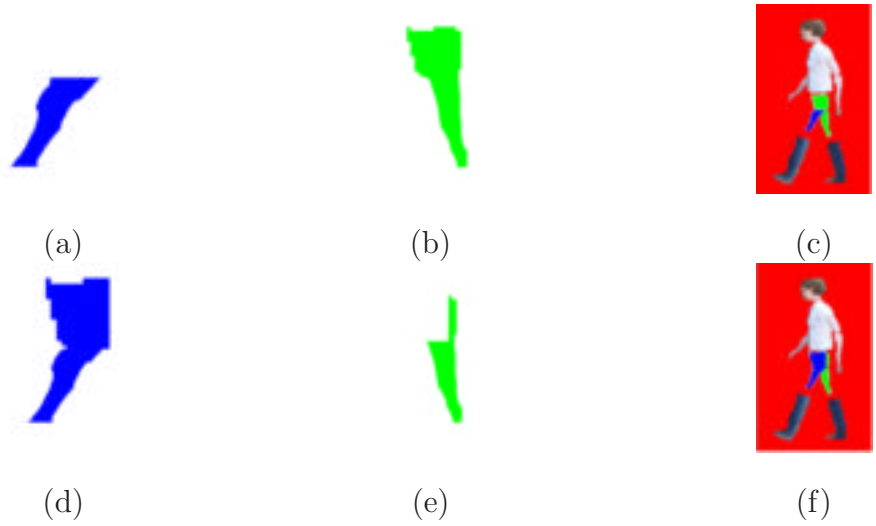


Figure 4.12: *An example of $\alpha\beta$-swap move for two segments, namely the upper half limbs of the two legs of the walking person shown in Fig. 4.1.* **(a)-(b)** *The initial shape estimate of the two half limbs which we denote by s_i (i.e. the blue segment) and s_k (i.e. the green segment) respectively. Note that some of the points which belong to s_i have been incorrectly assigned to the matte of s_k.* **(c)** *The projection of these mattes onto a frame of the video. Clearly the segmentation provided by these mattes is incomplete.* **(d)-(e)** *The refined shape estimates of the two half limbs obtained using the $\alpha\beta$-swap algorithm (when $\alpha = i$ and $\beta = k$). Note that the $\alpha\beta$-swap was computed by assuming that $o_i > o_k$ (i.e. s_i occludes s_k). We also compute an $\alpha\beta$-swap move by assuming that $o_i < o_k$ and choose those mattes which result in the lowest energy between the two assumptions.* **(f)** *The projection of the refined mattes onto the video frame.*

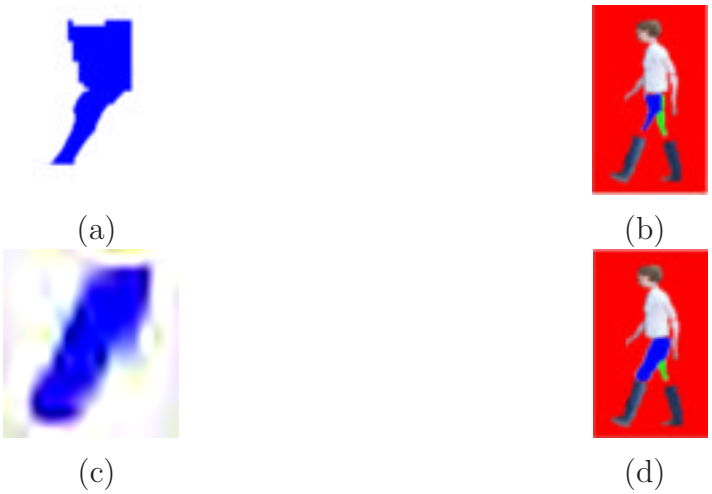


Figure 4.13: *An example of α-expansion move for the segment s_i in Fig. 4.12.* **(a)** *The shape estimate of s_i obtained using an $\alpha\beta$-swap move as shown in Fig. 4.12(d).* **(b)** *The projection of the matte onto a frame of the image. Note that this segmentation is the same as that shown in Fig. 4.12(f).* **(c)** *The refined shape estimate of s_i obtained after an α-expansion move (by assuming that $o_i > o_k$ where the segment s_k is shown in Fig. 4.12(e)). Note that the refined matte represents the complete shape of the half limb s_i. This can be seen by projecting the matte onto a video frame as shown in* **(d)**.

α-expansion algorithm would then grow the segments allowing them to overlap (e.g. see Fig. 4.13 for the result of growing the segment s_i in Fig. 4.12(d) using α-expansion). Therefore, refining the shape parameters $\mathbf{\Omega}_{Mi}$ of the layered representation requires both the $\alpha\beta$-swap and the α-expansion algorithm.

We now show how the iterative procedure described above allows us to determine the occlusion ordering using only those segments which overlap with each other. To this end, we define the *limit* $\mathcal{L}_i$ of a segment s_i as the set of points a_i whose distance from s_i is at most 40% of the current size of s_i. Given segment s_i, let s_k be a segment such that the limit $\mathcal{L}_i$ of s_i overlaps with s_k in at least one frame j of the video. Such a segment s_k is said to be *surrounding* the segment s_i. The number of surrounding segments s_k is quite small for objects such as humans and animals which are restricted in motion. For example, the head segment of the person shown in Fig. 4.1 is surrounded by only the torso and the background segments.

We iterate over segments and refine the shape of one segment s_i at a time. At each iteration, we perform an $\alpha\beta$-swap for s_i and each of its surrounding segments s_k. This relabels all the points which were wrongly labelled as belonging to s_i. We then perform an α-expansion algorithm to expand s_i to include those points a_i in its limit which move rigidly with s_i. During the iteration refining s_i, we consider three possibilities for s_i and its surrounding segment s_k: $o_i = o_k$, $o_i > o_k$

or $o_i < o_k$. Recall that if $o_i < o_k$, we assign $\Pr(\mathcal{I}^j(a_i)|\mathbf{\Omega}) = c_2$ for frames j where the image of a_i is occluded by the image of a point in s_k. We choose the option which results in the minimum value of $Q(\mathbf{\Omega}; \mathbf{D}, \boldsymbol{\theta})$. *This determines the occlusion ordering among surrounding segments.* We stop iterating when further reduction of $Q(\mathbf{\Omega}; \mathbf{D}, \boldsymbol{\theta})$ is not possible. This provides us with a refined estimate of $\mathbf{\Omega}_M$ along with the layer number o_i of the segments. Since the neighbourhood for each point a_i is small (see Fig. 4.3), graph cuts can be performed efficiently. The graph constructions for both the $\alpha\beta$-swap and α-expansion algorithms are provided in Appendix B.

Fig. 4.14 shows the refined shape parameters of the segments obtained by the above method using the initial estimates. Results indicate that reliable shape parameters can be learnt even while using a small neighbourhood. Note that, although the torso is partially occluded by the arm and the backleg is partially occluded by the front leg in every frame, their complete shape has been learnt using individual binary mattes for each segment. Next, the appearance parameters corresponding to the refined shape parameters are obtained.

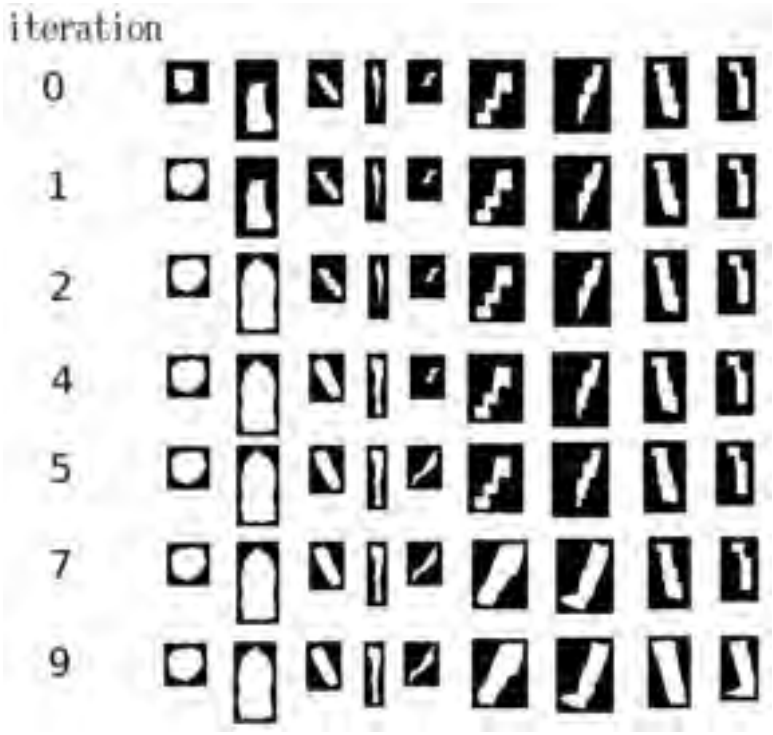


Figure 4.14: *Result of refining the mattes of the layered representation of a person using the iterative procedure described in § 4.3.3. The shape of the head is re-estimated after one iteration. The next iteration refines the torso segment. Subsequent iterations refine the half limbs one at a time. Note that the size of the mattes is equal to that of a frame of the video but smaller mattes are shown here for clarity.*

Figure 4.15: *Appearance of the parts learnt for the human model as described in § 4.3.4.*

4.3.4 Updating Appearance

Once the mattes $\mathbf{\Omega}_{Mi}$ of the segments are obtained, the appearance of a point $a_i \in s_i$, i.e. $\mathbf{\Omega}_{Ai}(a_i)$ is calculated as the mean of $\mathcal{I}^j(a_i)$ over all frames j. The histograms $\mathcal{H}_i$ are recomputed using the RGB values $\mathbf{\Omega}_{Ai}(a_i)$ for all points $a_i \in s_i$ (again using 15 bins each for R, G and B). Fig. 4.15 shows the appearance of the segments of the human model learnt using the video in Fig. 4.1. The refined shape and appearance parameters help in obtaining a better estimate for the transformations as described in the next section.

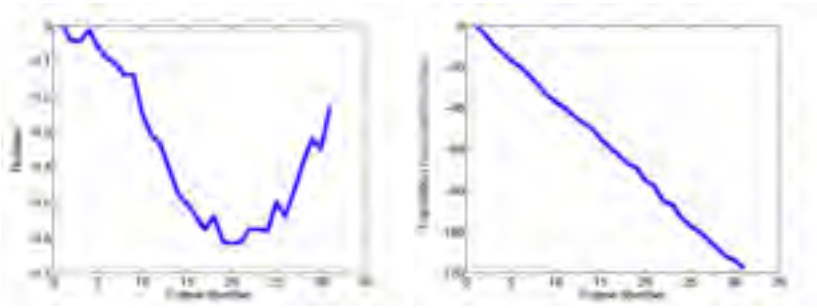
4.3.5 Refining the Transformations

Finally, the transformations $\mathbf{\Omega}_T$ and the lighting parameters $\mathbf{\Omega}_L$ are refined by searching over putative transformations around the initial estimate, for all segments at each frame j. For each putative transformation, the lighting parameters $\mathbf{\Omega}^j_{Li} = \{\mathbf{W}^j_i, \mathbf{w}^j_i\}$ are calculated in a least squares manner. The transformation $\mathbf{\Omega}^j_{Ti}$ which results in the smallest SSD is chosen. We search for putative transformations by considering translations upto 5 pixels in steps of 1, scales 0.9, 1.0 and 1.1 and rotations between -0.15 and 0.15 radians in steps of 0.15 radians around the initial estimate. Fig. 4.16 shows the rotation and translation in y-axis of the upper arm in Fig. 4.1, obtained after refining the transformations.

The model $\mathbf{\Omega}$, learnt using the five stage approach described above, can be used iteratively to refine the estimation of the layered representation. However, we found that it does not result in a significant improvement over the initial estimate as the parameters do not change much from one iteration to the other. Next, we describe a method to refine the segmentation of each frame.

4.3.6 Refining the Segmentation of Frames

Our model maps the segments onto a frame using only simple geometric transformations. This would result in gaps in the generated frame when the motion of segments cannot be accurately defined by such transformations (e.g. see Fig. 4.17(a)). In order to deal with this, we refine the segmentation of each frame by relabelling the points around the boundary of segments. Note that this step is



(a) (b)

Figure 4.16: **(a)** *Rotation of the upper arm obtained after refining the transformations as described in § 4.3.5. During the first half of the video, the arm swings away from the body while in the second half it rotates towards the body. Clearly, this motion has been captured in the learnt rotations.* **(b)** *Translation of the upper arm in the horizontal direction. The person moves from the right to the left of the scene with almost constant velocity as indicated by the learnt translations. Note that the transformations are refined individually for each frame and are therefore not necessarily smooth.*

performed only to obtain more accurate segmentations and does not change the values of any parameters. The relabelling is performed by using the α-expansion algorithm. The unary potential for assigning point a in frame j around the boundary of s_i to s_i is the inverse log likelihood of its observed RGB values in that frame given by the histogram $\mathcal{H}_i$.

The pairwise potential of assigning two neighbouring points a and b to the same segment is κ_1. The value of the pairwise potential is set to $\kappa_2 - \gamma^j(a,b)$ otherwise. The term $\gamma^j(a,b)$ is defined similar to $\gamma_{ik}(a_i,b_k)$ in equation (4.2.8), i.e.

$$\gamma^j(a,b) = \lambda\left(1 - \exp\left(\frac{-(\Delta^j(a,b))^2}{2\sigma^2}\right)\cdot\frac{1}{\texttt{dist}(a_i,b_k)}\right), \tag{4.3.2}$$

where

$$\Delta^j(a,b) = |\mathbf{D}^j(a) - \mathbf{D}^j(b)|, \tag{4.3.3}$$

i.e. $\Delta^j(a,b)$ is the difference in the observed RGB values of points a and b in frame j. Fig. 4.17 shows an example where the gaps in the segmentation are filled using the above method.

4.4. Results

We now present results for motion segmentation using the learnt layered representation of the scene. The method is applied to different types of object classes (such as jeep, humans and cows), foreground motion (pure translation, piecewise similarity transforms) and camera motion (static and translating). The background is assumed to be static. We use the same weight values (i.e. $\kappa_1 = 1$,

(a) (b)

Figure 4.17: *Result of refining the segmentation.* **(a)** *The segmentation obtained by compositing the transformed segments in descending order of the occlusion numbers.* **(b)** *The refined segmentation obtained using α-expansion (see text). Note that the gaps in the segmentation that appear in (a), e.g. between the upper and lower half limbs of the arm, have been filled.*

$\kappa_2 = 2.2$, $\lambda = 1$ and $\sigma = 5$) in all our experiments.

Fig. 4.18-4.20 show the segmentations obtained by generating frames using the learnt representation. All segments are composited in descending order of their occlusion numbers. Note that the segments belonging to the background (i.e. with occlusion number 0) are not projected onto the frames. Fig. 4.18(a) and 4.18(b) show the result of our approach on simple scenarios where each layer of the scene consists of segments which are undergoing pure translation. Despite having a lot of flexibility in the putative transformations (by allowing for various rotations and scales), the initial estimation recovers the correct transformations, i.e. those containing only translation. Note that the transparent windshield of the jeep is (correctly) not recovered in the M.A.S.H. sequence as the background layer can be seen through it. For the sequence shown in Fig. 4.18(a) the method proves robust to changes in lighting condition and it learns the correct layering for the segments corresponding to the two people.

Fig. 4.19(a) and 4.19(b) show the motion segmentation obtained for two videos, each of a person walking. In both cases, the body is divided into the correct number of segments (head, torso and seven visible half limbs). Our method recovers well from occlusion in these cases. For such videos, the feet of a person are problematic as they tend to move non-rigidly with the leg in some frames. Indeed the feet are missing from the segmentations. Note that the grass in Fig. 4.19(b) has similar intensity to the person's trousers which causes some errors in the transformations of the legs.

Fig. 4.20(a) and 4.20(b) are the segmentations of a cow walking. Again the body of the cow is divided into the correct number of segments (head, torso

(a)

(b)

Figure 4.18: *Motion segmentation results 1. In each case, the left image shows the various segments obtained in different colours. The top row shows the original video sequence while the segmentation results are shown in the bottom row.* **(a)** *A 40 frame sequence taken from a still camera (courtesy Nebojsa Jojic [38]). The scene contains two people undergoing pure translation in front of a static background. The results show that the layering is learnt correctly.* **(b)** *A 10 frame video sequence taken from 'M.A.S.H.'. The video contains a jeep undergoing translation and slight out of plane rotation against a static background while the camera pans to track the jeep.*

and eight half limbs). The cow in Fig. 4.20(a) undergoes a slight out of plane rotation in some frames, which causes some bits of grass to be pulled into the segmentation. The video shown in Fig. 4.20(b) is taken from a poor quality analog camera. However, our algorithm proves robust enough to obtain the correct segmentation. Note that when relabelling the points around the boundary of segments some parts of the background, which are similar in appearance to the cow, get included in the segmentation.

Our approach can also be used to segment objects present at different depths when the camera is translating. This is due to the fact that their transformations with respect to the camera will differ. Fig. 4.21 shows one such example using the well-known 'Garden' sequence. Note that the correct number of objects have been found and good segmentation is obtained.

Timing: The initial estimation takes approximately 5 minutes for every pair of frames: 3 minutes for computing the likelihood of the transformations and 2 minutes for MMSE estimation using coarse to fine BP. The shape parameters of the

(a)

(b)

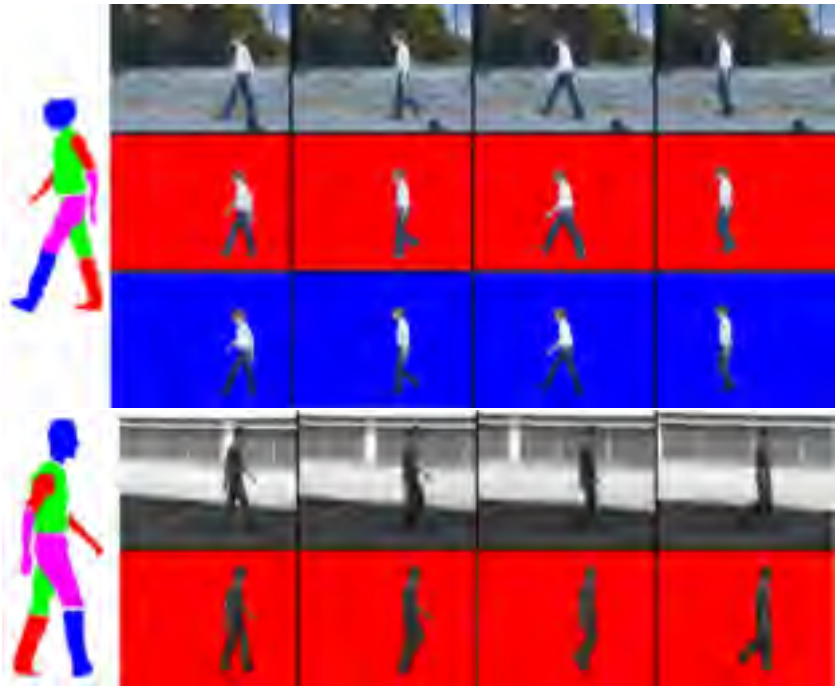


Figure 4.19: *Motion segmentation results 2.* **(a)** *A 31 frame sequence taken from a still camera (courtesy Hedvig Sidenbladh [86]). The scene consists of a person walking against a static background. The correct layering of various segments of the person is learnt. The ground truth used for comparison is also shown in the third row.* **(b)** *A 57 frame sequence taken from a translating camera of a person walking against a static background (courtesy Ankur Agarwal [1]). Again the correct layering of the segments is learnt.*

segments are refined by minimizing the energy $Q(\mathbf{\Omega}; \mathbf{D}, \boldsymbol{\theta})$ as described in § 4.3.3. The st-MINCUT algorithm used has, in practice, a time complexity which is linear in the number of points in the binary matte $\mathbf{\Omega}_{Mi}$. It takes less than 1 minute to refine the shape of each segment. Most of the time is taken up in calculating the various terms which define the energy shown in equation (4.2.4). Since the algorithm provides a good initial estimate, it converges after at most 2 iterations through each segment. All timings provided are for a C++ implementation on a 2.4 GHz processor.

Ground truth comparison: The segmentation performance of our method was assessed using eight manually segmented frames (four each from the challenging sequences shown in Fig. 4.19(a) and 4.20(b)). Out of 80901 ground truth foreground pixels and 603131 ground truth background pixels in these frames, 79198 (97.89%) and 595054 (98.66%) were present in the generated frames respectively. Most errors were due to the assumption of piecewise parametric motion and due to similar foreground and background pixels.

(a)

(b)

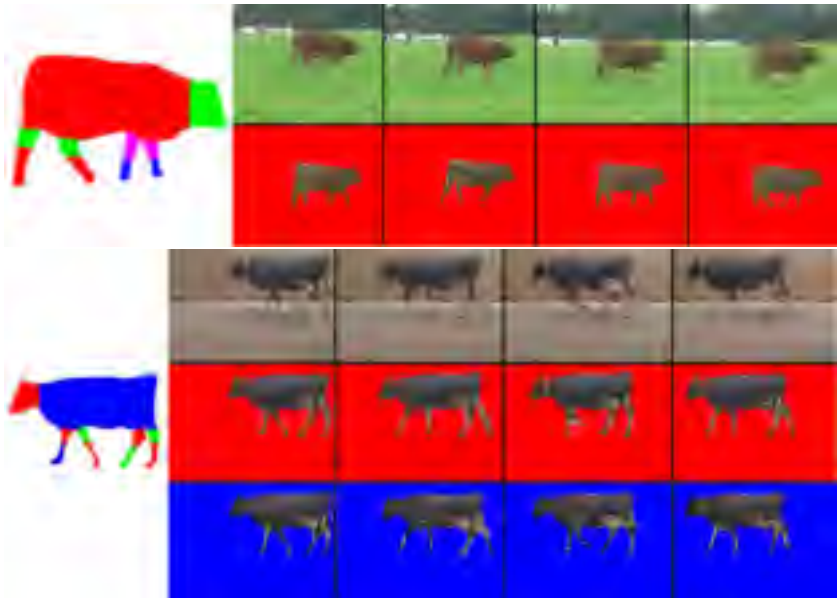


Figure 4.20: *Motion segmentation results 3.* **(a)** *A 44 frame sequence of a cow walking taken from a translating camera. All the segments, along with their layering, are learnt.* **(b)** *A 30 frame sequence of a cow walking against a static (homogeneous) background (courtesy Derek Magee [61]). The video is taken from a still analog camera which introduces a lot of noise. The results obtained using our approach (row 2) and the ground truth used for comparison (row 3) are also shown.*

Sensitivity to weights: Recall that, in order to obtain the rigidly moving components, we compute the image motion for each patch $\mathbf{r}_a$ in a frame j (i.e. the transformation that maps the pixels in $\mathbf{r}_a$ from frame j to $j+1$). Fig. 4.22 shows the effects of computing the image motion and obtaining the components without favouring rigid motion between neighbouring patches. This is achieved by using a higher value of ζ in equation (4.3.2), specifically one which scales $\zeta\nabla(\mathbf{r}_a, \mathbf{r}_b)$ to lie between 0 and 2 (i.e. twice the value of ζ used earlier). As expected, this oversegments the body of the person by learning a large number of rigidly moving components. This problem is overcome by favouring rigid motion for any two neighbouring patches using the CRF described in § 4.3.1.

Our model explicitly accounts for spatial continuity using the weights κ_1, κ_2 and λ. Fig. 4.23 shows the effects of setting these weights to zero, thereby not modelling spatial continuity. Note that this significantly deteriorates the quality of the segmentation when the background is homogeneous.

Sensitivity to length of sequence: We tested our approach by varying the number of frames used for the video sequence shown in Fig. 4.1. Since the number of segments (and their initial shape) is found using rigidly moving components,

Figure 4.21: **Segmenting objects.** *The top row shows some frames from the 29 frame 'Garden' sequence taken from a translating camera. The scene contains four objects, namely the sky, the house, the field and the tree, at different depths which are learnt correctly. The bottom row shows the appearance and shape of the segmented objects.*

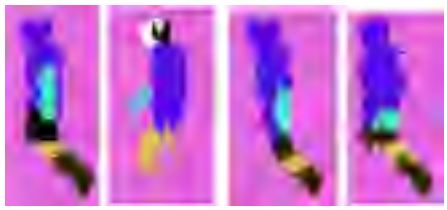
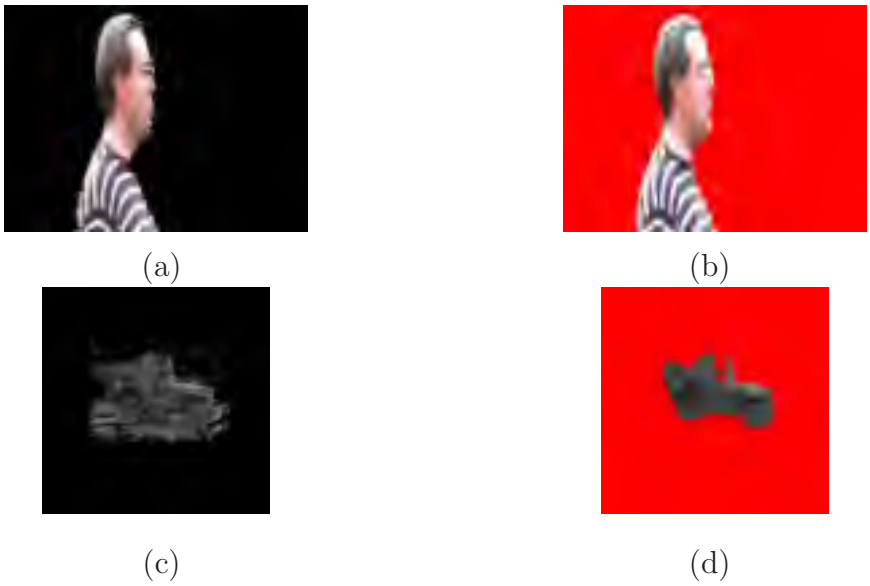


Figure 4.22: *Results of finding components for four frames from the video shown in Fig. 4.1 by using a greater tolerance for non-rigid motion between two neighbouring patches. A large number of components are found by clustering rigidly moving patches, i.e. the method oversegments the body of the person. For example, the head and the torso are oversegmented in the first and second image respectively, while the third and fourth images show the oversegmentation of a leg.*

using fewer frames tends to undersegment the object. For example, given 10 frames of a video where the two half limbs of an arm move rigidly together, our method would detect the arm as one segment. Fig. 4.24 shows the initial estimate of the segments obtained for a varying number of input frames. Note that the video sequence contains two half-periods of motion (i.e. the person takes two steps forward, first with the left leg and then with the right leg). As expected, the algorithm undersegments the body when the full period of motion is not considered. By the twenty fourth frame, i.e. just after the beginning of the second half-period, all visible segments are detected due to the difference in their transformations. Using more than twenty four frames does not change the number of segments obtained. However, the initial estimate of the segments changes as smaller components are found in subsequent frames (see § 4.3.2).



(a) (b)

(c) (d)

Figure 4.23: *Encouraging spatial continuity.* **(a)** *Result obtained by setting the weights κ_1, κ_2 and λ to zero. The method works well for the simple case of the video shown in Fig. 4.18(a) where the foreground and background differ significantly. When compared with ground truth, 93.1% of foreground pixels and 99.8% of background pixels were labelled correctly.* **(b)** *By encouraging spatial continuity, a small improvement is observed (95.4% of foreground pixels and 99.9% of background pixels were present in the generated frame).* **(c)** *For the more difficult case shown in Fig. 4.18(b), the segmentation starts to include parts of the homogeneous background when spatial continuity is not enforced. Only 91.7% of foreground pixels and 94.1% of background pixels are generated, compared to 95% of foreground pixels and 99.8% of background pixels correctly obtained when encouraging spatial continuity (shown in* **(d)***).*

4.5. Discussion

The algorithm proposed in this chapter achieves good motion segmentation results. Why is this? We believe that the reasons are two fold. Incremental improvements in the Computer Vision field have now ensured that: (i) we can use an appropriate model which accounts for motion, changes in appearance, layering and spatial continuity. The model is not too loose so as to undersegment, and not too tight so as to oversegment; (ii) we have more powerful modern algorithmic methods such as sum-product BP and graph cuts which avoid local minima better than previous approaches.

However, there is still more to do. As is standard in methods using layered representation, we have assumed that the visual aspects of the objects and the layering of the segments do not change throughout the video sequence. At the very least we need to extend the model to handle the varying visual aspects of

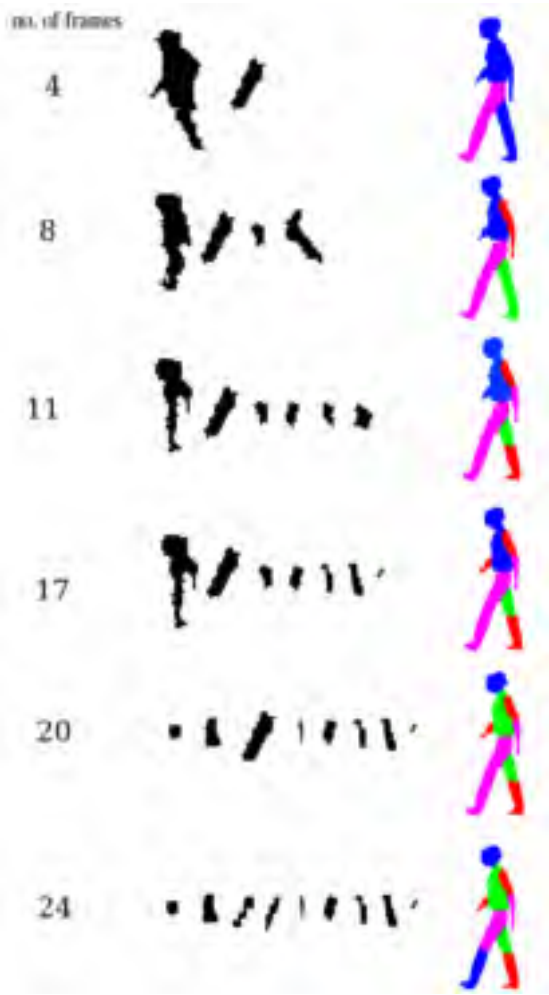


Figure 4.24: *Results of obtaining the initial estimate of the segments for a varying number of input frames. The refined estimates of the shape obtained using the method described in § 4.3.3 are also shown. During the first four frames only two segments are detected, i.e. the body and a leg. In the next four frames, the arm close to the camera and the other leg are detected. The half limbs which constitute this arm and leg are detected using 11 frames of the video sequence. When 24 frames are used, all 9 visible segments of the body are detected. The initial estimate of the segments and the refined estimate of their shapes for the entire video sequence is shown in Fig. 4.14.*

objects present in the scene, e.g. front, back and 3/4 views, in addition to the side views. The restriction of rigid motion within a segment can be relaxed using non-parametric motion models.

For our current implementation, we have set the values of the weights empirically. Although these values provide good results for a large class of videos, it would be interesting to learn them using ground truth segmentations (similar to [9] for image segmentation).

Chapter 5

OBJCUT

In this chapter, we present an application of optimization for object category specific segmentation. We propose a principled probabilistic method for detecting and segmenting instances of a particular object category within an image. Our approach overcomes the deficiencies of previous segmentation techniques based on traditional grid conditional random fields (CRF), namely that (i) they require the user to provide seed pixels for the foreground and the background; and (ii) they provide a poor prior for specific shapes due to the small neighbourhood size of grid CRF.

Specifically, we replace the manual interaction by automatic object detection. Furthermore, we propose a new probabilistic model which includes strong shape potentials for the object, which incorporate top-down information that is global across the image, in addition to the grid clique potentials, which provide the bottom-up information used in previous approaches.

The shape potentials are provided by a detection of the object using an object category model. We represent articulated object categories using a novel layered pictorial structures model. Non-articulated object categories are modelled using a set of exemplars. These object category models have the advantage that they can handle large intra-class shape, appearance and spatial variation.

We develop an efficient method, ObjCut, to obtain segmentations using our probabilistic framework. Novel aspects of this method include: (i) efficient algorithms for sampling the object category models of our choice; and (ii) the observation that the expected log likelihood of the model can be increased by a single graph cut. Results are presented on several articulated (e.g. animals) and non-articulated (e.g. fruits) object categories. We compare our method with the state of the art in object category specific image segmentation and demonstrate significant improvements.

5.1. Introduction

Image segmentation has seen renewed interest in the field of Computer Vision, in part due to the arrival of new efficient algorithms to perform the segmentation [13], and in part due to the resurgence of interest in object category detection [2, 25, 58]. Segmentation fell from favour due to an excess of papers attempting to solve ill posed problems with no means of judging the result. In contrast, interleaved object detection and segmentation [10, 58, 69, 84, 112] is both well posed and of practical use. Well posed in that the result of the segmentation can be quantitatively judged, e.g. how many pixels have been correctly and incorrectly assigned to the object. Of practical use because image editing tools can be designed that provide a *power assist* to cut out applications like 'Magic Wand', e.g. "I know this is a horse, please segment it for me, without the pain of having to manually delineate the boundary."

The conditional random field (CRF) framework [56] provides a useful model of images for segmentation and their prominence has been increased by the availability of efficient publically available code for their solution. The approach of Boykov and Jolly [13], and more recently its application in a number of systems including *GrabCut* by Rother *et al.* [76], strikingly demonstrates that with a minimum of user assistance objects can be rapidly segmented (e.g. by employing user-specified foreground and background seed pixels, see § 2.2.2). However samples from the distribution defined by the commonly used grid CRFs (e.g. with 4 or 8-neighbourhood) very rarely give rise to realistic shapes and on their own are ill suited to segmenting objects. For example, Fig. 5.1(c) shows the result of segmenting an image containing a cow using the method described in [13]. Note that the segmentation does not look like the object despite using a large number of seed pixels (see Fig. 5.1(b)) due to the small neighbourhood size of the grid CRF, which cannot provide global top-down information.

In contrast, models used for object detection utilize the global information of the object to localize it in the image. For example, it is common practice to represent an object using a set of shape and/or texture exemplars [29, 92, 97]. Given an image, the object can be detected by matching the exemplars to the image. Such a model is particularly suitable for non-articulated object categories where a sufficiently large set of exemplars can be used to handle intra-class shape and appearance variation.

For articulated objects, in addition to shape and appearance, there might also be a considerable spatial variation (e.g. see Fig. 5.19). In order to manage this variability there is a broad agreement that articulated object categories should be represented by a collection of spatially related parts each with its own shape and appearance. This sort of approach dates back to the pictorial structures model

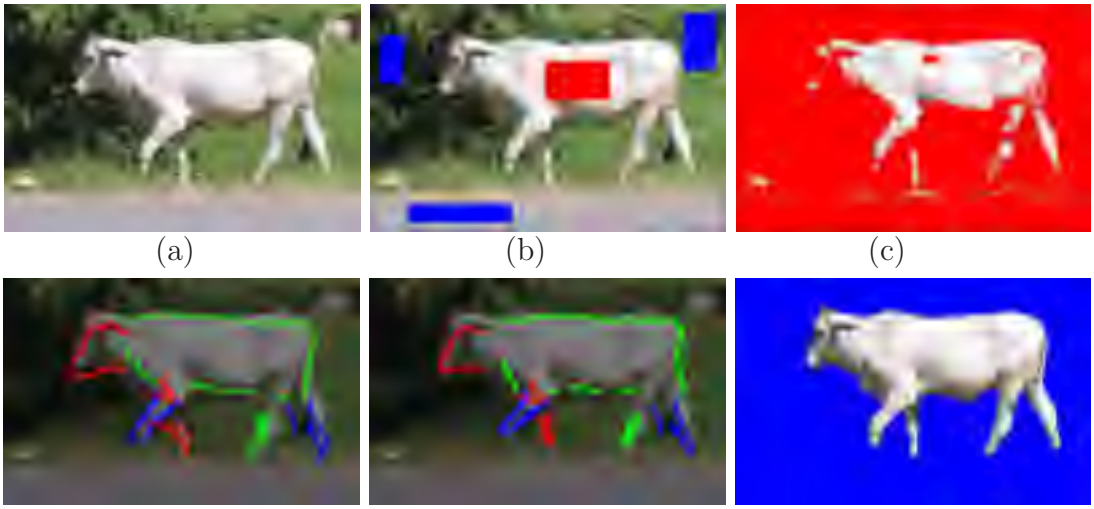


Figure 5.1: *Segmentation obtained using the* CRF *formulation.* **(a)** *Original image containing a cow.* **(b)** *The red and blue rectangles indicate the object and background seed pixels respectively which are provided by the user.* **(c)** *Segmentation obtained using the method described in [13]. Note that the segmentation is not object-like due to the poor prior provided by the grid* CRF. **(d)**,**(e)** *The cow is roughly localized using the pictorial structures model [23, 26]. The parts detected are shown overlaid on the image. Note that the position and shape of the parts differs between the two detections (e.g. the head and the torso).* **(f)** *The segmentation obtained using our method. Unlike previous methods [9, 13, 76], the segmentation is accurate and object-like.*

introduced by Fischler and Elschlager three decades ago [26]. Recently, pictorial structures [23] and other related models [25, 69, 84] have been shown to be very successful for the task of object recognition. Furthermore, pictorial structures have been highly effective in detecting fronto-parallel views of objects [23, 74]. In other words, they not only make a decision about the presence or absence of an object category in an image but they also provide a rough localization of the object. However, these models alone are not suitable for obtaining a pixel-wise segmentation of the image. For example, see Fig. 5.1(d) and (e) for two results of matching the pictorial structures model of a cow to an image [53].

In this work, we combine the models used for object detection with the grid CRF framework used for segmentation. The coarse localization of the object obtained by matching a model to an image provides us rough regions where the foreground (i.e. the object) and background are present. These regions are used to obtain the object and background seed pixels. The seed pixels could then be directly used to obtain the segmentation using CRF based methods. The result would be an automated Boykov-Jolly style segmentation algorithm [13]. However, such an approach would still suffer from the problem that the distribution of the grid CRF would not provide a good estimate for the shape of the object. In order to address this deficiency, we go beyond the probabilistic models of previous approaches. Specifically, we introduce a new framework that combines the grid CRF (which provides bottom-up information) with an object category model (which provides global top-down information across the image plane).

Using the above framework, pixels of an image can be labelled as belonging to the foreground or the background by jointly inferring the MAP estimate of the object detection and segmentation. However, it would be undesirable to depend only on the MAP detection since it may not localize some portion of the object well. We overcome this problem by marginalizing over various detections obtained for a given image. Fig. 5.1(d) and (e) show two such detections found using the pictorial structures model of a cow. Fig. 5.1(f) shows the segmentation obtained using the approach described in this chapter. Note that, unlike previous methods, the segmentation is accurate and object-like.

In more detail, we cast the problem of object category specific segmentation as that of estimating a probabilistic model which consists of an object category model in addition to the grid CRF. The object category model provides the top-down information which encourages the segmentation to resemble the object. For this work, we represent non-articulated objects using a set of exemplars. In order to handle spatial variation, we use the pictorial structures model (with suitable modifications) for articulated objects. However, it is worth noting that our method is general and can be extended to any object category model.

We develop an efficient method, ObjCut, to obtain segmentations using this framework. The basis of our method are two new theoretical/algorithmic contributions: (i) we provide a highly efficient algorithm for marginalizing or optimizing

the object category models of our choice; and (ii) we make the (not obvious) observation that the expectation of the log likelihood of an CRF under the distribution of some latent variables can be efficiently optimized by a single st-MINCUT.

Related Work: Many different approaches for segmentation using both top-down and bottom-up information have been reported in the literature. Huang *et al.* [36] describe an iterative algorithm which alternates between fitting an active contour to an image and segmenting it on the basis of the shape of the active contour. Cremers *et al.* [20] extend this by using multiple competing shape priors and identifying the image regions where shape priors can be enforced. However, the use of level sets in these methods makes them computationally inefficient. Freedman *et al.* [28] describe an algorithm based on st-MINCUT which uses a shape prior for segmentation. However, their framework lacks a probabilistic formulation. Furthermore, all these methods require manual interaction.

There are a few automatic methods for combining top-down and bottom-up information reported in the literature. For example, Borenstein and Ullman [10] describe an algorithm for segmenting instances of a particular object category from images using a patch-based model learnt from training images. Leibe and Schiele [58] provide a probabilistic formulation for this while incorporating spatial information of the relative locations of the patches. Winn and Jojic [111] describe a generative model which provides the segmentation by applying a smooth deformation field on a class specific mask. Shotton *et al.* [85] propose a novel texton-based feature which captures long range interactions to provide pixel-wise segmentation. However, all the above methods use a weak model for the shape of the object which does not provide realistic segmentations.

Winn and Shotton [112] present a segmentation technique using a parts-based model which incorporates spatial information between neighbouring parts. Their method allows for arbitrary scaling but it is not clear whether their model is applicable to articulated object categories. Levin and Weiss [59] describe an algorithm which learns a set of fragments for a particular object category that assist the segmentation. The learnt fragments provide only local cues for segmentation as opposed to the global information used in our work. The segmentation also relies on the maximum likelihood estimate of the position of these fragments on a given test image (found using normalized cross-correlation). This has two disadvantages:

- The spatial relationship between the fragments is not considered while matching them to an image (e.g. it is possible that the fragment corresponding to the legs of a horse is located above the fragment corresponding to the head). Thus the segmentation obtained would not be object-like. In contrast, we marginalize over the object category model while taking into account the spatial relationships between the parts of the model.

- The algorithm becomes susceptible to error in the presence of background clutter. Indeed, the segmentations provided by [59] assume that a rough localization of the object is known *a priori* in the image. It is not clear whether normalized cross-correlation would provide good estimates of the fragment positions in the absence of such knowledge.

Outline: The chapter is organized as follows. In section 5.2 the probabilistic model for object category specific segmentation is described. Section 5.3 gives an overview of an efficient method for solving this model for foreground-background labellings. We provide the details of our choice of representations for articulated and non-articulated objects in section 5.4. The important issue of automatic object detection is addressed in section 5.5. Results for several articulated and non-articulated object categories and a comparison with other methods is given in section 5.6.

5.2. Object Category Specific Segmentation

In this section we describe the model that forms the basis of our work. We formally specify and build upon previous work on segmentation, providing a probabilistic graphical model. There are three issues to be addressed in this section: (i) how to make the segmentation conform to foreground and background appearance models; (ii) how to encourage the segmentation to follow edges within the image; and (iii) how to encourage the segmentation to look like an object.

Markov random fields (MRF) offer a probabilistic formulation for the problem of segmentation. Recall that an MRF is characterized by pairwise prior terms which are independent of the data (see § 2.2.1). However, in segmentation, it is desirable to have pairwise data dependent terms [9, 13, 76] (e.g. see § 2.2.2). For this purpose, we make use of the discriminative CRF framework. Once again we will find it convenient to use the alternative formulation of CRF, i.e. the contrast dependent random field (CDRF) framework described in the previous chapter. A major drawback of previously used grid CDRF formulations is that, due to small neighbourhood size, the segmentations obtained do not form a realistic shape for a particular object category. In order to overcome this deficiency, we combine the grid CDRF with a strong model for the object category of interest. We now describe our probabilistic model in detail. For the sake of completeness, we start with an MRF formulation and build up to our model from there.

Markov random field: Given an image $\mathbf{D}$ containing an instance of a known object category, e.g. cows, we wish to segment the image into foreground, i.e. pixels belonging to the object, and background. In order to formulate the problem of segmentation, we define an MRF over random variables $\mathbf{v} = \{v_0, v_1, \cdots, v_{n-1}\}$,

where each random variable v_a corresponds to a pixel a of the image $\mathbf{D}$. The random variables can take labels from the set $\mathbf{l} = \{l_0, l_1\}$ where l_0 stands for foreground and l_1 stands for background. A segmentation can then be defined as a labelling $f : \{0, 1, \cdots, n-1\} \longrightarrow \{0, 1\}$ which assigns label $l_{f(a)}$ to v_a. In other words, if v_a is assigned label l_0 or l_1 then pixel a belongs to the foreground and background respectively. The joint probability of the labelling f and the data $\mathbf{D}$ is given by

$$\Pr(f, \mathbf{D}|\boldsymbol{\theta}) = \Pr(\mathbf{D}|f, \boldsymbol{\theta}) \Pr(f|\boldsymbol{\theta}) = \frac{1}{Z_1(\boldsymbol{\theta})} \exp(-Q_1(f; \mathbf{D}, \boldsymbol{\theta})), \tag{5.2.1}$$

where $\boldsymbol{\theta}$ is the parameter for the MRF and $Z_1(\boldsymbol{\theta})$ is the partition function. By assuming the Markovian property on the prior $\Pr(f|\boldsymbol{\theta})$, the energy $Q_1(f; \mathbf{D}, \boldsymbol{\theta})$ can be written as the summation of clique potentials, i.e.

$$Q_1(f; \mathbf{D}, \boldsymbol{\theta}) = \sum_{v_a \in \mathbf{v}} \theta^1_{a;f(a)} + \sum_{(a,b) \in \mathcal{E}} \theta^2_{ab;f(a)f(b)}, \tag{5.2.2}$$

where $\mathcal{E}$ is the neighbourhood relationship defined by the MRF. Typically, the neighbourhood of a is defined as its 4 or 8-neighbourhood which results in a grid MRF. The unary and pairwise potentials take a form which is similar to the interactive binary image segmentation MRF discussed in § 2.2.1. Below, we describe these potentials in detail.

Unary Potential: The unary potential $\theta^1_{a;f(a)}$ is the emission model for a pixel and is given by

$$\theta^1_{a;f(a)} = \begin{cases} -\log(\Pr(\mathbf{D}_a|\mathcal{H}_{obj})) & \text{if} \quad f(a) = 0 \\ -\log(\Pr(\mathbf{D}_a|\mathcal{H}_{bkg})) & \text{if} \quad f(a) = 1, \end{cases}$$

where $\mathcal{H}_{obj}$ and $\mathcal{H}_{bkg}$ are the appearance models for foreground and background respectively. For this work, $\mathcal{H}_{obj}$ and $\mathcal{H}_{bkg}$ modelled as RGB distributions. Recall that $\mathbf{D}_a$ denotes the RGB values of the pixel a.

Pairwise Potential: The pairwise potential $\theta^2_{ab;f(a)f(b)}$ takes the form of an Ising model:

$$\theta^2_{ab;f(a)f(b)} = \begin{cases} \kappa_1 & \text{if} \quad f(a) = f(b), \\ \kappa_2 & \text{if} \quad f(a) \neq f(b), \end{cases}$$

where $\kappa_1 < \kappa_2$. Thus the pairwise potential encourages smooth segmentation by (equally) favouring any two neighbouring pixels having the same label.

Contrast Dependent Random Field: The MRF formulation described above does not allow for data dependent pairwise terms. However, as noted in § 2.2.2, the models for image segmentation include a contrast term which is used to favour pixels with similar colour having the same label [9, 13, 76]. Together with a prior term, this encourages contiguous segmentations whose boundaries lie on image edges. The contrast and prior terms can be included within the CDRF framework

by assuming the Markovian property on the conditional distribution $\Pr(f|\mathbf{D},\boldsymbol{\theta})$ (instead of $\Pr(f|\boldsymbol{\theta})$ in the case of MRF) [56]. We provide the exact form of these terms below.

Prior Term: Let $f(a)$ and $f(b)$ be the labels for variables v_a and v_b respectively. Then the corresponding prior term is similar to the pairwise potential defined in equation (5.2.3), i.e.

$$\theta^{p}_{ab;f(a)f(b)} = \begin{cases} \kappa_1 & \text{if} \quad f(a) = f(b), \\ \kappa_2 & \text{if} \quad f(a) \neq f(b), \end{cases}$$

Contrast Term: The contrast term $\theta^{c}_{ab;f(a)f(b)}$ is given by

$$\theta^{c}_{ab;f(a)f(b)} = \begin{cases} 0 & \text{if} \quad f(a) = f(b), \\ -\gamma(a,b) & \text{if} \quad f(a) \neq f(b). \end{cases} \tag{5.2.3}$$

Note that the contrast term is similar to the ones used in chapters 2 and 4. Its form is inspired by previous work on segmentation [9, 13, 76]. We define $\gamma(a,b)$ such that it reduces the cost within the Ising model prior for $f(a) \neq f(b)$ in proportion to the difference in intensities of pixels a and b, i.e.

$$\gamma(a,b) = \lambda \left(1 - \exp\left(\frac{-\Delta^2(a,b)}{2\sigma^2}\right) \frac{1}{\text{dist}(a,b)}\right), \tag{5.2.4}$$

where $\Delta(a,b)$ measures the difference in the RGB values of pixels a and b, i.e. $\mathbf{D}_a$ and $\mathbf{D}_b$, and $\text{dist}(a,b)$ is the Euclidean distance between a and b.

We use the same weight values as the previous chapters, i.e. $\kappa_1 = 1$, $\kappa_2 = 2.2$, $\lambda = 1$ and $\sigma = 5$. These values are suitable for encouraging contiguous segments whose boundaries lie on image edges (see table 4.4). Empirically, these weights were found to provide good results for a large variety of images.

Reducing the cost of the Ising model prior term in this manner makes the pairwise terms, i.e. $\theta^{p}_{ab;f(a)f(b)} + \theta^{c}_{ab;f(a)f(b)}$, discontinuity preserving [48]. Recall that the contrast term $\theta^{c}_{ab;f(a)f(b)}$ cannot be included in the prior (since the prior term is not data dependent). Rather it leads to a pairwise linkage between neighbouring random variables and pixels as shown in the graphical model given in Fig. 5.2. The posterior probability of the labelling is defined as

$$\Pr(f|\mathbf{D},\boldsymbol{\theta}) = \frac{1}{Z_2(\boldsymbol{\theta})} \exp(-Q_2(f;\mathbf{D},\boldsymbol{\theta})), \tag{5.2.5}$$

where $Z_2(\boldsymbol{\theta})$ is the partition function. The energy $Q_2(f;\mathbf{D},\boldsymbol{\theta})$ is given by

$$Q_2(f;\mathbf{D},\boldsymbol{\theta}) = \sum_{v_a \in \mathbf{v}} \theta^{1}_{a;f(a)} + \sum_{(a,b)\in\mathcal{E}} \left(\theta^{p}_{ab;f(a)f(b)} + \theta^{c}_{ab;f(a)f(b)}\right). \tag{5.2.6}$$

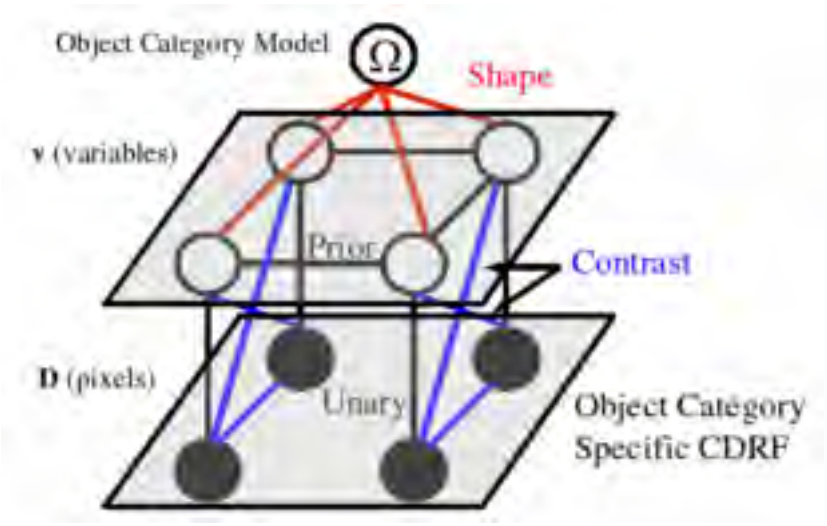


Figure 5.2: *Graphical model representation of the Object Category Specific* CDRF*. The random variables* $\mathbf{v}$ *are shown as unfilled circles, while the observed data* $\mathbf{D}$ *is shown using filled circles. The connections induced by the contrast term are shown as the blue edges below the random variables. Note that some of these connections (e.g. connecting the random variables on the left with pixels on the right) are not shown for the sake of clarity of the image. The random variables* $\mathbf{v}$ *lie in a plane. Together with the pixels shown below this plane, these form the* CDRF *used for segmentation. In addition to these terms, the Object Category Specific* CDRF *makes use of an object category model* $\mathbf{\Omega}$ *(shown lying above the plane). The model* $\mathbf{\Omega}$ *guides the segmentation towards a realistic shape closely resembling the object of interest.*

Object Category Specific CDRF**:** We introduce a strong object category model, parameterized by $\mathbf{\Omega}$, to the grid CDRF framework which will favour segmentations of a specific shape as shown in the graphical model depicted in Fig. 5.2. We refer to this extension of the grid CDRF model as the *Object Category Specific* CDRF. The Object Category Specific CDRF has the following energy function:

$$Q_3(f, \mathbf{\Omega}; \mathbf{D}, \boldsymbol{\theta}) = \sum_{v_a \in \mathbf{v}} \left(\theta^A_{a;f(a)} + \theta^S_{a;f(a)}\right) + \sum_{(a,b)\in\mathcal{E}} \left(\theta^p_{ab;f(a)f(b)} + \theta^c_{ab;f(a)f(b)}\right), \quad (5.2.7)$$

with posterior

$$\Pr(f, \mathbf{\Omega}|\mathbf{D}, \boldsymbol{\theta}) = \frac{1}{Z_3(\boldsymbol{\theta})} \exp(-Q_3(f, \mathbf{\Omega}; \mathbf{D}, \boldsymbol{\theta})). \quad (5.2.8)$$

Here $\boldsymbol{\theta}$ is the parameter of the Object Category Specific CDRF and $Z_3(\boldsymbol{\theta})$ is the partition function. The prior term $\theta^p_{ab;f(a)f(b)}$ and contrast term $\theta^c_{ab;f(a)f(b)}$ are as defined above. The potentials $\theta^A_{a;f(a)}$ and $\theta^S_{a;f(a)}$ are described below.

Appearance Potential: The appearance potential $\theta^A_{a;f(a)}$ is the same as the unary potential of the CDRF i.e.

$$\theta^A_{a;f(a)} = \begin{cases} -\log(\Pr(\mathbf{D}_a|\mathcal{H}_{obj})) & \text{if} \quad f(a) = 0 \\ -\log(\Pr(\mathbf{D}_a|\mathcal{H}_{bkg})) & \text{if} \quad f(a) = 1, \end{cases}$$

Shape Potential: We call the term $\theta^S_{a;f(a)}$ as the shape potential since it influences the shape of the segmentation to resemble the object. The shape potential $\theta^{\mathbf{S}}_{a;f(a)}$ is chosen such that, given $\mathbf{\Omega}$ (i.e. one possible localization of the object), the random variables corresponding to pixels that fall near to a detected object would be more likely to have foreground label (i.e. l_0) than random variables corresponding to pixels lying far from the object. It has the form:

$$\theta^S_{a;f(a)} = -\log \Pr(f(a)|\mathbf{\Omega}). \quad (5.2.9)$$

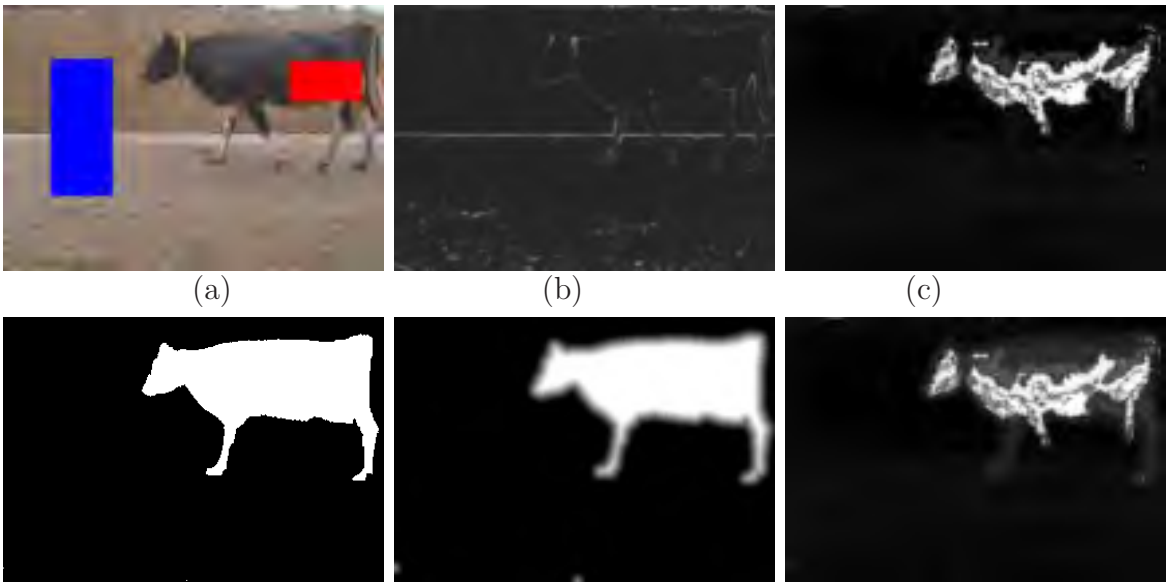
We choose to define $\Pr(f(a)|\mathbf{\Omega})$ as

$$\Pr(f(a) = 0|\mathbf{\Omega}) = \frac{1}{1 + \exp(\mu * \text{dist}(a, \mathbf{\Omega}))}, \quad (5.2.10)$$

$$\Pr(f(a) = 1|\mathbf{\Omega}) = 1 - \Pr(f(a) = 0|\mathbf{\Omega}), \quad (5.2.11)$$

where $\text{dist}(a, \mathbf{\Omega})$ is the spatial distance of a pixel a from the outline of the object defined by $\mathbf{\Omega}$ (being negative if inside the shape). The weight μ determines how much the pixels outside the shape are penalized compared to the pixels inside the shape.

Hence, the model $\mathbf{\Omega}$ contributes the unary term $\theta^S_{a;f(a)}$ for each pixel a in the image for a labelling f (see Fig. 5.2). Alternatively, $\mathbf{\Omega}$ can also be associated with the CDRF using pairwise terms as described in [28]. However, by reparameterizing the CDRF [34], both formulations can be shown to be equivalent. We prefer the use of unary terms since they do not effect the submodularity of the energy.



(a) (b) (c)

(d) (e) (f)

Figure 5.3: **(a)** *An example cow image. The red and blue rectangles show the seed pixels which are used to learn the* RGB *distribution of foreground ($\mathcal{H}_{obj}$) and background ($\mathcal{H}_{bkg}$) respectively.* **(b)** *The pairwise terms (prior+contrast) for a pixel summed over its entire neighbourhood. Pixels marked bright white indicate the presence of an image edge and are hence, more likely to define the segmentation boundary.* **(c)** *The unary potential ratio $\frac{\theta^1_{a;0}}{\theta^2_{a;1}}$ of a pixel computed using $\mathcal{H}_{obj}$ and $\mathcal{H}_{bkg}$. Pixels marked white are more likely to belong to foreground than the pixels marked black. Clearly, the likelihoods obtained using only* RGB *values are not sufficient to obtain a good segmentation.* **(d)** *The object category model $\mathbf{\Omega}$. White pixels are the points that lie inside the object while black pixels lie outside it.* **(e)** *The ratio $\frac{\theta^S_{a;0}}{\theta^S_{a;1}}$ corresponding to the model $\mathbf{\Omega}$. Again, pixels marked white are more likely to belong to foreground than background.* **(f)** *The ratio of the unary terms, i.e. $\frac{\theta^A_{a;0}+\theta^S_{a;0}}{\theta^A_{a;1}+\theta^S_{a;1}}$. Compared to (c), the unary terms in (f) provide more information about which pixels belong to the foreground and the background. Together with the pairwise terms shown in (b), this allows us to obtain a good segmentation of the object shown in (a).*

Hence, it can easily be shown that the energy function $Q_3(f, \mathbf{\Omega}; \mathbf{D}, \boldsymbol{\theta})$ can be minimized via st-MINCUT [48]. Fig. 5.3 shows the advantage of introducing an object category model in the CDRF.

In this work we use two types of object category models: (i) For non-articulated objects, $\mathbf{\Omega}$ is represented by a set of exemplars; (ii) For articulated objects, $\mathbf{\Omega}$ is specified by our extension of the pictorial structures model [23, 26]. However, we would like to emphasize that our methodology is completely general and could be combined with any sort of object category model (e.g. see [16]).

The Object Category Specific CDRF framework defined above provides the probability of the labelling f and the object category model $\mathbf{\Omega}$ as defined in equation (5.2.8). However, the optimal foreground-background labelling should be obtained by maximizing the posterior probability $\Pr(f|\mathbf{D})$. In order to achieve this, we must integrate out $\mathbf{\Omega}$ i.e.

$$\Pr(f|\mathbf{D}) = \int \Pr(f, \mathbf{\Omega}|\mathbf{D}) d\mathbf{\Omega}. \tag{5.2.12}$$

The surprising result of this work is that this rather intractable looking integral can in fact be optimized by a simple and computationally efficient set of operations, as described in the next section.

5.3. Roadmap of the Solution

We now provide a high-level overview of our approach. Given an image $\mathbf{D}$, the problem of segmentation requires us to obtain a labelling f^* which maximizes the posterior probability $\Pr(f|\mathbf{D})$, i.e.

$$f^* = \arg\max \Pr(f|\mathbf{D}) = \arg\max \log \Pr(f|\mathbf{D}). \tag{5.3.1}$$

We have dropped the term $\boldsymbol{\theta}$ from the above notation to make the text less cluttered. We note however that there is no ambiguity about $\boldsymbol{\theta}$ for the work described in this chapter (i.e. it always stands for the parameter of the Object Category Specific CDRF). In order to obtain realistic shapes, we would also like to influence the segmentation using an object category model $\mathbf{\Omega}$ (as described in the previous section). Given an Object Category Specific CDRF specified by one instance of $\mathbf{\Omega}$, the required posterior probability $\Pr(f|\mathbf{D})$ can be computed as

$$\Pr(f|\mathbf{D}) = \tfrac{\Pr(f, \mathbf{\Omega}|\mathbf{D})}{\Pr(\mathbf{\Omega}|f, \mathbf{D})}, \tag{5.3.2}$$

$$\Rightarrow \quad \log \Pr(f|\mathbf{D}) = \log \Pr(f, \mathbf{\Omega}|\mathbf{D}) - \log \Pr(\mathbf{\Omega}|f, \mathbf{D}), \tag{5.3.3}$$

where $\Pr(f, \mathbf{\Omega}|\mathbf{D})$ is given by equation (5.2.8) and $\Pr(\mathbf{\Omega}|f, \mathbf{D})$ is the conditional probability of $\mathbf{\Omega}$ given the image and its labelling. Note that we consider the log of the posterior probability $\Pr(f|\mathbf{D})$. As will be seen, this allows us to marginalize the object category model $\mathbf{\Omega}$ using the Expectation Maximization (EM) [30]

framework in order to obtaining the desired labelling f^*. By marginalizing $\mathbf{\Omega}$ we would ensure that the segmentation is not influenced by only one instance of the object category model (which may not localize the entire object correctly, leading to undesirable effects such as inaccurate segmentation).

We now briefly describe the EM framework which provides a natural way to deal with $\mathbf{\Omega}$ by treating it as missing (latent) data. The EM algorithm starts with an initial estimate f^0 of the labelling and iteratively refines it by marginalizing over $\mathbf{\Omega}$. It has the desirable property that during each iteration the posterior probability $\Pr(f|\mathbf{D})$ does not decrease (i.e. the algorithm is guaranteed to converge to a local maximum). Given the current guess of the labelling f', the EM algorithm consists of two steps: (i) E-step: where the probability distribution $\Pr(\mathbf{\Omega}|f', \mathbf{D})$ is obtained; and (ii) M-step: where a new labelling $\hat{f}$ is computed such that $\Pr(\hat{f}|\mathbf{D}) \geq \Pr(f'|\mathbf{D})$. We briefly describe how the two steps of the EM algorithm can be computed efficiently in order to obtain the segmentation. We subsequently provide the details for both the steps.

Efficiently Computing the E-step: Given the estimate of the labelling f', we approximate the desired distribution $\Pr(\mathbf{\Omega}|f', \mathbf{D})$ by sampling efficiently for $\mathbf{\Omega}$. For non-articulated objects, this involves computing similarity measures at each location in the image. In § 5.5.1, we show how this can be done efficiently. For the case of articulated objects, we make use of the efficient sum-product belief propagation (sum-product BP) algorithm described in the previous chapter, which efficiently computes the marginals for a non regular Potts model (i.e. when the labels are not specified by an underlying grid of parameters, complementing the result of Felzenszwalb and Huttenlocher [24]).

Efficiently Computing the M-step: Once the samples of $\mathbf{\Omega}$ have been obtained in the E-step, we need to compute a new labelling $\hat{f}$ such that $\Pr(\hat{f}|\mathbf{D}) \geq \Pr(f'|\mathbf{D})$. We show that such a labelling $\hat{f}$ can be computed by minimizing a weighted sum of energy functions of the form given in equation (5.2.7), where the summation is over the samples of $\mathbf{\Omega}$ (see below for details). The weights are given by the probability of the samples. This suggests that the labelling $\hat{f}$ can be obtained efficiently using a single st-MINCUT operation [48].

Details: We concentrate on the M-step first. We will later show how the E-step can be approximately computed using image features. Given the distribution $\Pr(\mathbf{\Omega}|f', \mathbf{D})$, we average equation (5.3.3) over $\mathbf{\Omega}$ to obtain

$$\log \Pr(f|\mathbf{D}) = E(\log \Pr(f, \mathbf{\Omega}|\mathbf{D})) - E(\log \Pr(\mathbf{\Omega}|f, \mathbf{D})), \tag{5.3.4}$$

where $E(\cdot)$ indicates the expectation under $\Pr(\mathbf{\Omega}|f', \mathbf{D})$. The key observation of the EM algorithm is that the second term on the right side of equation (5.3.4),

i.e.

$$E(\log \Pr(\mathbf{\Omega}|f,\mathbf{D})) = \int (\log \Pr(\mathbf{\Omega}|f,\mathbf{D})) \Pr(\mathbf{\Omega}|f',\mathbf{D}) d\mathbf{\Omega} \tag{5.3.5}$$

is maximized when $f = f'$. We obtain a labelling $\hat{f}$ such that it maximizes the first term on the right side of equation (5.3.4), i.e.

$$\hat{f} = \arg\max E(\log \Pr(f,\mathbf{\Omega}|\mathbf{D})) = \arg\max \int (\log \Pr(f,\mathbf{\Omega}|\mathbf{D})) \Pr(\mathbf{\Omega}|f',\mathbf{D}) d\mathbf{\Omega}. \tag{5.3.6}$$

since, if $\hat{f}$ is different from f', then it is guaranteed to increase the posterior probability $p(f|\mathbf{D})$. This is due to the following two reasons: (i) $\hat{f}$ increases the first term of equation (5.3.4), i.e. $E(\log \Pr(\mathbf{\Omega}, f|\mathbf{D}))$, as it is obtained by maximizing this term; and (ii) $\hat{f}$ decreases the second term of equation (5.3.4), i.e. $E(\log \Pr(\mathbf{\Omega}|f,\mathbf{D}))$, which is maximized when $f = f'$. If $\hat{f}$ is the same as f' then the algorithm is said to have converged to a local maximum of the distribution $\Pr(f|\mathbf{D})$. The expression in equation (5.3.6) is called the expected complete-data log-likelihood in the EM literature.

In section 5.5, it will be shown that we can efficiently sample from the object category model $\mathbf{\Omega}$ of our choice. This suggests a sampling based solution to maximizing equation (5.3.6). Let the set of s samples be $\mathbf{\Omega}_1, \ldots, \mathbf{\Omega}_s$, with weights $w_i = \Pr(\mathbf{\Omega}_i|f',\mathbf{D})$. Using these samples equation (5.3.6) can be approximated as

$$\hat{f} = \arg\max_f \textstyle\sum_{i=1}^{i=s} w_i(-Q_3(f,\mathbf{\Omega}_i;\mathbf{D})) - C, \tag{5.3.7}$$

$$\Rightarrow \quad \hat{f} = \arg\min_f \textstyle\sum_{i=1}^{i=s} w_i Q_3(f,\mathbf{\Omega}_i;\mathbf{D}) + C. \tag{5.3.8}$$

The form of the energy $Q_3(f,\mathbf{\Omega}_i;\mathbf{D})$ is given in equation (5.2.7). The term $C = \sum_i w_i \log Z_3(\boldsymbol{\theta})$ is a constant which does not depend on f or $\mathbf{\Omega}$ and can be therefore be ignored during the minimization. *This is the key equation of our approach.* We observe that this energy function is a weighted linear sum of the energies $Q_3(f,\mathbf{\Omega};\mathbf{D})$ which, being a linear combination with positive weights w_i, can also be optimized using a single st-MINCUT operation [48] (see Fig. 5.4). This demonstrates the interesting result that for CDRF (and MRF) with latent variables, it is computationally feasible to optimize the complete-data log-likelihood.

The EM algorithm converges to a local minima of $\Pr(f|\mathbf{D})$ and its success depends on the initial labelling f^0. In the last section a graphical model for pixel by pixel segmentation was set up. However, it would be computationally expensive to use this model straight off. Rather, we adopt an initialization stage in which we get a rough estimate of the posterior probability of $\mathbf{\Omega}$ from a set of image features $\mathbf{Z}$ (defined in § 5.4.1). Image features (such as textons and edges) can provide high discrimination at low computational cost. We approximate the initial distribution $\Pr(\mathbf{\Omega}|f^0,\mathbf{D})$ as $g(\mathbf{\Omega}|\mathbf{Z})$, where $\mathbf{Z}$ are some image features chosen to localize the object in a computationally efficient manner. The weights w_i required to evaluate equation (5.3.8) on the first EM iteration are obtained by sampling from the distribution $g(\mathbf{\Omega}|\mathbf{Z})$ (defined in section 5.4).

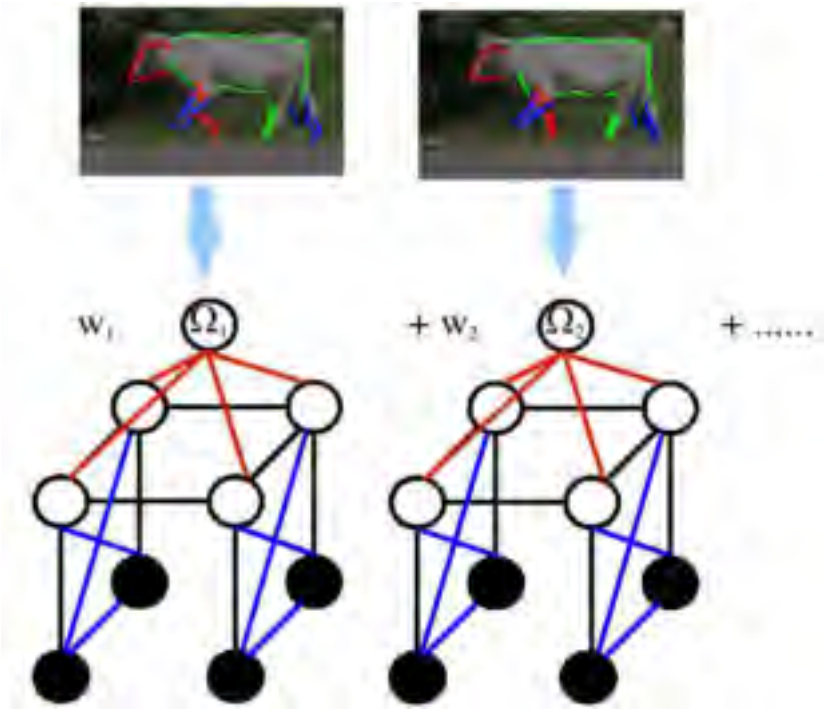


Figure 5.4: *The top row shows various samples of a cow model for a given image. Each sample* $\mathbf{\Omega}_i$ *gives rise to one instance of the Object Category Specific* CDRF *which can be solved to obtain a segmentation using a single st-*MINCUT *operation on a graph, say* $\mathcal{G}_i$ *(see § 2.4.3). The segmentation which increases the expected complete-data log-likelihood is found using a single st-*MINCUT *operation on the weighted average of all graphs* $\mathcal{G}_i$ *where the weights* w_i *are defined by the probability* $\Pr(\mathbf{\Omega}_i|f', \mathbf{D})$ *of the various samples.*

One might argue that if the MAP estimate of the object has a very high posterior probability compared to other poses, then equation (5.3.8) can be approximated using only the MAP estimate $\mathbf{\Omega}^*$ instead of the samples $\mathbf{\Omega}_1, \ldots, \mathbf{\Omega}_s$. However, we found that this is not the case especially when the RGB distribution of the background is similar to that of the object. For example, Fig. 5.17 shows various samples obtained by matching the model for a cow to two images. Note that different samples localize different parts of the object correctly and have similar posterior probabilities. Thus it is necessary to use multiple samples of the object category model.

The roadmap described above results in the OBJCUT algorithm, which obtains object category specific segmentation. Algorithms 1 and 2 summarize the main steps of OBJCUT for non-articulated and articulated object categories respectively.

- Input: An image $\mathbf{D}$ and a non-articulated object category model.
- Initial estimate of pose using edges and texture (§ 5.5.1.1):
  1. A set of candidate poses $t_o = (x_o, y_o, \theta_o, \sigma_o)$ for the object is identified using a tree cascade of classifiers which computes a set of image features $\mathbf{Z}$.
  2. The maximum likelihood estimate is chosen as initial estimate of the pose.
- Improved estimation of pose taking into account colour (§ 5.5.1.2):
  1. The appearance model of both foreground and background is updated.
  2. A new set of candidate poses is generated for the object by densely sampling pose space around the estimate found in the above step (again, using a tree cascade of classifiers for computing $\mathbf{Z}$).
- The samples $\mathbf{\Omega}_1, \ldots, \mathbf{\Omega}_s$ are obtained from the posterior $g(\mathbf{\Omega}|\mathbf{Z})$ of the object category model as described in § 5.5.1.3.
- OBJCUT
  1. The weights $w_i = g(\mathbf{\Omega}_i|\mathbf{Z})$ are computed.
  2. The energy in equation (5.3.8) is minimized using a single st-MINCUT operation to obtain the segmentation f.

Algorithm 1: *The* OBJCUT *algorithm for non-articulated object categories.*

- Input: An image $\mathbf{D}$ and an articulated object category model.
- Initial estimate of pose using edges and texture (§ 5.5.2.1):
  1. A set of candidate poses $t_i = (x_i, y_i, \theta_i, \sigma_i)$ for each part is identified using a tree cascade of classifiers which computes a set of image features $\mathbf{Z}$.
  2. An initial estimate of the poses of the parts is found without considering the layering of parts using an efficient sum-product BP algorithm.
- Improved estimation of pose taking into account colour and occlusion (§ 5.5.2.2):
  1. The appearance model of both foreground and background is updated.
  2. A new set of candidate poses is generated for each part by densely sampling pose space around the estimate found in the above step (again, using a tree cascade of classifiers for computing $\mathbf{Z}$).
  3. The pose of the object is estimated using efficient sum-product BP and the layering of the parts.
- The samples $\boldsymbol{\Omega}_1, \ldots, \boldsymbol{\Omega}_s$ are obtained from the posterior $g(\boldsymbol{\Omega}|\mathbf{Z})$ of the object category model as described in § 5.5.2.3.
- ObjCut
  1. The weights $w_i = g(\boldsymbol{\Omega}_i|\mathbf{Z})$ are computed.
  2. The energy in equation (5.3.8) is minimized using a single st-MINCUT operation to obtain the segmentation f.

Algorithm 2: *The* ObjCut *algorithm for articulated object categories.*

In the remainder of the chapter, we provide details of the object category model $\boldsymbol{\Omega}$ of our choice. We propose efficient methods to obtain the samples from the posterior probability distribution of $\boldsymbol{\Omega}$ required for the marginalization in equation (5.3.8). We demonstrate the results on a number of articulated and non-articulated object categories.

5.4. Object Category Models

When choosing the model $\boldsymbol{\Omega}$ for the Object Category Specific CDRF, two issues need to be considered: (i) whether the model can handle intra-class shape and appearance variation (and, in the case of articulated objects, spatial variation); and (ii) whether samples from the distribution $g(\boldsymbol{\Omega}|\mathbf{Z})$ (which are required for segmentation) can be obtained efficiently.

We represent the *shape* of an object (or a part, in the case of articulated objects) using multiple exemplars of the boundary. This allows us to handle the intra-class shape variation. The *appearance* of an object (or part) is represented using multiple texture exemplars. Again, this handles the intra-class appearance variation. Note that the exemplars model the shape and appearance of an object category. **These should not be confused with the shape and appearance potentials** of the Object Category Specific CDRF used to obtain the segmentation.

Once an initial estimate of the object is obtained, its appearance is known. Thus the localization of the object can be refined using a better appearance model (i.e. one which is specific to that instance of the object category). For this purpose, we use histograms which define the distribution of the RGB values of the foreground and the background.

We define the model $\mathbf{\Omega}$ for non-articulated objects as a set of shape and texture exemplars (see Fig. 5.5). In the case of articulated objects, one must also allow for considerable spatial variation. For this purpose, we use the pictorial structures (PS) model. However, the PS models used in previous work [23, 26] assume non-overlapping parts which are connected in a tree structure. We extend the PS by incorporating the layering information of parts and connecting them in a complete graph structure. We call the resulting representation as the *layered pictorial structures* (LPS) model (see Fig. 5.6). Below, we describe the object category models for non-articulated and articulated objects in detail.

5.4.1 Set of Exemplars Model

We represent non-articulated object categories as 2D patterns with a probabilistic model for their shape and appearance. The shape of the object category is represented using a set of shape exemplars $\mathcal{S} = \{\mathbf{S}_1, \mathbf{S}_2, \cdots, \mathbf{S}_e\}$. For this work, each shape exemplar $\mathbf{S}_i$ is given by a set of points $\{s_{i;1}, s_{i;2}, \cdots, s_{i;m}\}$ describing the outline of the object. Similarly, the appearance is represented using a set of texture exemplars $\mathcal{T} = \{\mathbf{T}_1, \mathbf{T}_2, \cdots, \mathbf{T}_e\}$, where each exemplar is an image patch (i.e. a set of intensity values). Note that we use multiple exemplars (i.e. $e > 1$) to handle the shape and appearance variations which are common in non-articulated object categories. We call this the set of exemplars (SOE) model. Note that similar models were used for object detection in [29, 92, 97].

5.4.1.1 Feature likelihood for object

Given the putative pose of an object, i.e. $\mathbf{t}_o = \{x_o, y_o, \phi_o, \rho_o\}$ (where $\{x_o, y_o\}$ is the location, ϕ_o is the rotation and ρ_o is the scale), we computed two features $\mathbf{Z} = \{z_1, z_2\}$ for the shape and appearance of the object respectively. Let $\mathbf{D}_o \subseteq \mathbf{D}$ be the set of pixels corresponding to the object at pose $\mathbf{t}_o$. The features z_1 and z_2 are computed using $\mathbf{D}_o$. Assuming independence of the two features, the likelihood based on the whole data is approximated as

$$\Pr(\mathbf{Z}|\mathbf{\Omega}) = \Pr(z_1)\Pr(z_2) \tag{5.4.1}$$

where $\Pr(z_1) \propto \exp(-z_1)$ and $\Pr(z_2) \propto \exp(-z_2)$. We also assume the prior $\Pr(\mathbf{\Omega})$ to be uniform. This provides us with the distribution $g(\mathbf{\Omega}|\mathbf{Z})$ as

$$g(\mathbf{\Omega}|\mathbf{Z}) \propto \Pr(\mathbf{Z}|\mathbf{\Omega})\Pr(\mathbf{\Omega}) \propto \Pr(\mathbf{Z}|\mathbf{\Omega}). \tag{5.4.2}$$

We now describe the features z_1 and z_2 in detail.

Outline (z_1): The likelihood of the object shape should be robust to outliers resulting from cluttered backgrounds. To this end, we define z_1 as the minimum of the truncated chamfer distances over all the exemplars of the object at pose $\mathbf{t}_o$. Let $\mathbf{U} = \{u_1, u_2, \cdots, u_m\}$ represent the edges of the image at $\mathbf{t}_o$. Then z_1 is computed as

$$z_1 = \min_{\mathbf{S}_i \in \mathcal{S}} d_{cham}(\mathbf{S}_i, \mathbf{U}). \tag{5.4.3}$$

The truncated chamfer distance $d_{cham}(\cdot, \cdot)$ is given by

$$d_{cham}(\mathbf{S}_i, \mathbf{U}) = \frac{1}{m} \sum_j \min\{\min_k ||u_k - s_{i;j}||, \tau_1\}, \tag{5.4.4}$$

where τ_1 is a threshold for truncation which reduces the effect of outliers and missing edges. Orientation information is included by computing $\min_k ||u_k - s_{i;j}||$ only over those edge points u_k which have a similar orientation to $s_{i;j}$. This makes the chamfer distance more robust [29]. We use 8 orientation groups for the outline points.

Texture (z_2): We use the VZ classifier [98] which provides a histogram representation $\mathcal{H}_i$ for each exemplar $\mathbf{T}_i$[1]. It also provides a histogram $\mathcal{H}_o$ for the image patch $\mathbf{D}_o$. The feature z_2 is computed as

$$z_2 = \min_{\mathbf{T}_i \in \mathcal{T}} d_{chi}(\mathcal{H}_i, \mathcal{H}_o), \tag{5.4.5}$$

where $d_{chi}(\cdot, \cdot)$ is the χ^2 distance[2].

5.4.1.2 Learning the exemplars

In order to learn the exemplars, we use manually segmented images. The outline of each segmented image provides us with an exemplars $\mathbf{S}_i \in \mathcal{S}$. The texture exemplars $\mathbf{T}_i$ are given by the subimage marked as foreground. We use 20 segmented images each for the 'banana' and the 'orange' categories. A subset of the shape exemplars $\mathcal{S}$ of these two categories is shown in Fig. 5.5.

We now describe an extension to the PS which is used as the model $\mathbf{\Omega}$ for articulated objects.

[1]The VZ classifier obtains a texton dictionary by clustering intensity values in an $N \times N$ neighbourhood of each pixel in $\mathbf{T}_i$ for all $\mathbf{T}_i \in \mathcal{T}$. The histogram $\mathcal{H}_i$ is given by the frequency of each entry of this texton dictionary in $\mathbf{T}_i$. We use $N = 3$ in our experiments.

[2]The feature z_2 described here handles the intra-class variation in appearance and is used to determine an initial estimate of the pose of the object. This estimate is then refined using a better appearance model (i.e. specific to a particular instance of the object category) as described in § 5.5.1.2.

Figure 5.5: *A selection of the multiple exemplars used to represent the model for bananas and oranges. Multiple shape exemplars are required to handle intra-class shape variability.*

5.4.2 Layered Pictorial Structures

In the case of articulated objects, we use the PS model to handle large deformations. PS are compositions of 2D patterns, termed *parts*, under a probabilistic model for their shape, appearance and spatial layout. However, the PS models used previously in [23, 26] are not directly suitable for applications such as efficient segmentation due to the following reasons: (i) they use a weak likelihood model which results in a large number of putative poses for each part; (ii) the parts are connected in a tree structure and hence, provide a weak spatial model; and (iii) they do not explicitly model self-occlusion. Hence, different parts with similar shape and appearance (e.g. the legs of cows or horses) are often incorrectly detected at the same pose (i.e. even in cases where they are actually at different poses in the given image).

We overcome the deficiencies of previous PS models by extending them in three ways: (i) similar to SOE, the likelihood of a part includes both its outline and its texture which results in a small number of putative poses for each part in a given image (see § 5.5.2.1); (ii) all parts are connected to each other to form a complete graph instead of a tree structure which provides a better spatial model; and (iii) similar to the model described in the previous chapter (and also in [1]), each part p_i is assigned an occlusion number o_i which determines its relative depth. The occlusion numbers allow us to explicitly model self-occlusion. Specifically, a part p_i can partially or completely occlude part p_j if and only if $o_i > o_j$. Note that several parts can have the same occlusion number if they are at the same depth. Such parts, which share a common occlusion number, are said to lie in the same layer. We call this model layered pictorial structures (LPS).

5.4.2.1 Posterior of the LPS

An LPS can also be viewed as an MRF where the random variables of the MRF correspond to the n_P parts. Each random variable takes one of n_L labels which encode the putative poses of the part. Similar to the pose of an object described in § 5.4.1, the pose of the i^{th} part is defined by a label $\mathbf{t}_i = \{x_i, y_i, \phi_i, \rho_i\}$. For a given pose $\mathbf{t}_i$ and image $\mathbf{D}$, the part p_i corresponds to the subset of the image pixels $\mathbf{D}$ which are used to calculate features $\mathbf{Z}_i$.

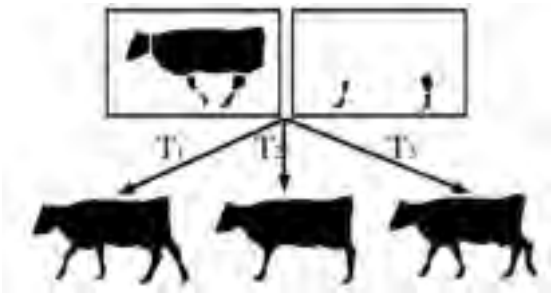


Figure 5.6: *Layered pictorial structures model of a cow. The various parts belonging to layers 2 and 1 are shown in the top left and right images respectively. Parts undergo different transformations and are composited in decreasing order of occlusion numbers to generate an instance of the object category (in this case, cows). The posterior of such an instance is given by equation (5.4.10).*

By slightly overloading the notation of chapter 2, the posterior of the LPS can be written as

$$g(\mathbf{\Omega}|\mathbf{Z}) = \Pr(\mathbf{Z}|\mathbf{\Omega})\Pr(\mathbf{\Omega}), \tag{5.4.6}$$

where $\mathbf{Z} = \{\mathbf{Z}_1, \ldots, \mathbf{Z}_{n_P}\}$ are the image features, $p(\mathbf{Z}|\mathbf{\Omega})$ is the feature likelihood and $p(\mathbf{\Omega})$ is the prior. Like the SOE model, the shape of an LPS is specified by a set of shape exemplars $\mathcal{S}_i$ for each part p_i. The appearance of an LPS is modelled using a set of texture exemplars $\mathcal{T}$ for the object category. Note that unlike the shape exemplars, which are specific to a part of an object category, the texture exemplars are specific to the object category of interest. Assuming that the features $\mathbf{Z}_i$ are computed by not including pixels accounted for by parts p_j for which $o_j > o_i$ (i.e. parts which can occlude p_i), the feature likelihood is given by

$$\Pr(\mathbf{Z}|\mathbf{\Omega}) = \prod_{i=1}^{i=n_P} \Pr(\mathbf{Z}_i|\mathbf{\Omega}). \tag{5.4.7}$$

The feature likelihood $\Pr(\mathbf{Z}_i|\mathbf{\Omega})$ for part p_i is computed as described in § 5.4.1.1. Specifically, the likelihood of the first feature, i.e. $\Pr(z_1)$, is computed using the minimum of the truncated chamfer distance, over the set $\mathcal{S}_i$ for the part, at pose $\mathbf{t}_i$. The texture likelihood, $p(z_2)$, is obtained from the VZ classifier using the set $\mathcal{T}$ for the object category.

LPS, like PS, are characterized by pairwise only dependencies between the random variables. These are modelled as a prior on the relative poses of parts:

$$\Pr(\mathbf{\Omega}) \propto \exp\left(-\sum_{i=1}^{i=n_P}\sum_{j=1,j\neq i}^{j=n_P} \alpha(\mathbf{t}_i, \mathbf{t}_j)\right). \tag{5.4.8}$$

Note that we use a completely connected MRF as this was found to provide a better localization of the object than a tree structured MRF [53]. In our approach, the

pairwise potentials $\alpha(\mathbf{t}_i, \mathbf{t}_j)$ of putative poses for each pair of parts are given by a non-regular Potts model, i.e.

$$\alpha(\mathbf{t}_i, \mathbf{t}_j) = \begin{cases} d_1 & \text{if valid configuration} \\ d_2 & \text{otherwise,} \end{cases}$$

where $d_1 < d_2$. In other words, all valid configurations are considered equally likely and have a smaller cost. A configuration is considered valid if the difference between the two poses $\mathbf{t}_i$ and $\mathbf{t}_j$ lies in an interval defined by $\mathbf{t}_{ij}^{min} = \{x_{ij}^{min}, y_{ij}^{min}, \theta_{ij}^{min}, \sigma_{ij}^{min}\}$ and $\mathbf{t}_{ij}^{max} = \{x_{ij}^{max}, y_{ij}^{max}, \theta_{ij}^{max}, \sigma_{ij}^{max}\}$, i.e.

$$\mathbf{t}_{ij}^{min} \leq |\mathbf{t}_i - \mathbf{t}_j| \leq \mathbf{t}_{ij}^{max}. \tag{5.4.9}$$

Note that the above inequalities should be interpreted component-wise (i.e. $x_{ij}^{min} \leq |x_i - x_j| \leq x_{ij}^{max}$ and so on). For each pair of parts p_i and p_j the terms $\mathbf{t}_{ij}^{min}$ and $\mathbf{t}_{ij}^{max}$ are learnt using training video sequences as described in § 5.4.2.2. Using equation (5.4.6), the posterior of the LPS parameters is given by

$$g(\mathbf{\Omega}|\mathbf{Z}) \propto \prod_{i=1}^{i=n_P} \Pr(\mathbf{Z}_i|\mathbf{\Omega}) \exp\left(-\sum_{j \neq i} \alpha(\mathbf{t}_i, \mathbf{t}_j)\right). \tag{5.4.10}$$

5.4.2.2 Learning the LPS

We now describe how we learn the various parameters of the LPS model for cows. To this end, we use 20 cow videos of 45 frames each and learn the segments using the motion segmentation method described in the previous chapter. Correspondence between the segments learnt from two different videos is established using shape context with continuity constraints [92] as shown in Fig. 5.7. The corresponding segments then define a part of the LPS model. The outline of the segments defines the shape exemplars $\mathcal{S}_i$ (see Fig. 5.8), while the intensity values of the segmented cows provides the set $\mathcal{T}$. Furthermore, an estimate of $|\mathbf{t}_i - \mathbf{t}_j|$ is also obtained (after rescaling the frames of the video such that the width of the cows is 230 pixels), for each frame and for all pairs of parts p_i and p_j. This is used to compute the parameters $\mathbf{t}_{ij}^{min}$ and $\mathbf{t}_{ij}^{max}$ that define valid configurations.

To obtain the LPS model for horses, we use 20 manually segmented images. The texture exemplars can be obtained using the segmented images. However, since these images do not provide us with any motion information, we cannot use the method in chapter 4 to obtain the shape exemplars of the LPS model. In order to overcome this problem, we establish a point to point correspondence between the outline of a cow from a training video and the outlines of the horses, again using shape context with continuity constraints [92] (see Fig. 5.9). Using this correspondence and the learnt parts of the cow, the parts of the horse are now easily determined (see Fig. 5.10). The part correspondence obtained also maps the parameters $\mathbf{t}_{ij}^{min}$ and $\mathbf{t}_{ij}^{max}$ that were learnt for cows to horses.

Figure 5.7: *Correspondence using shape context matching with continuity constraints. Outlines of two cows which need to be matched are shown. Lines are drawn to indicate corresponding points.*

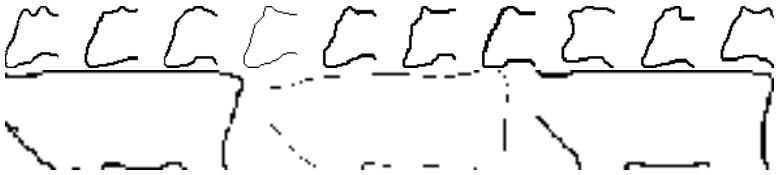


Figure 5.8: *The first row shows a subset of shape exemplars $\mathcal{S}_i$ for the head of a cow (obtained by establishing a correspondence between a set of segmented cow images as shown in Fig. 5.7). The second row shows shape exemplars of the torso part.*

Figure 5.9: *Correspondence using shape context matching with continuity constraints. Outlines of a horse and a cow are shown. Lines are drawn to indicate corresponding points.*

Figure 5.10: *The first and second row show the multiple exemplars of the head and the torso part respectively. The exemplars are obtained by establishing a correspondence between segmented images of cows and horses as shown in Fig. 5.9.*

In the next section, we address the important issue of developing efficient algorithms for matching the model $\mathbf{\Omega}$ to an image.

5.5. Sampling the Object Category Models

Given an image $\mathbf{D}$, our objective is to match the object category model to it in order to obtain samples from the distribution $g(\mathbf{\Omega}|\mathbf{Z})$. We achieve this in three stages:

- *Initialization*, where we fit the object category model to a given image $\mathbf{D}$ by computing features z_1 (i.e. chamfer) and z_2 (i.e. texture) using exemplars. This provides us with a rough object pose.
- *Refinement*, where the initial estimate is refined by computing z_2 using a better appearance model (i.e. the RGB distribution for the foreground and background learnt using the initial pose together with the shape) instead of the texture feature used during initialization. In the case of articulated objects, the layering information is also used.
- *Sampling*, where samples are obtained from the distribution $g(\mathbf{\Omega}|\mathbf{Z})$.

5.5.1 Sampling the SOE

We now describe the three stages for obtaining samples by matching the SOE model (for a non-articulated object category) to a given image.

5.5.1.1 Initial estimation of pose

In order to obtain the initial estimate of the pose of an object, we need to compute the feature likelihood for each pose using all exemplars. This would be computationally expensive due to the large number of possible poses and exemplars. However, most poses have a very low likelihood since they do not cover the pixels containing the object of interest. We require an efficient method which discards such poses quickly. To this end, we use a *tree cascade of classifiers* [88].

We term the rotated and scaled versions of the shape exemplars as *templates*. When matching many similar templates to an image, a significant speed-up is achieved by forming a template hierarchy and using a coarse-to-fine search. The idea is to group similar templates together with an estimate of the variance of the error within the cluster, which is then used to define a matching threshold. For each cluster, a prototype of the cluster is first compared to the image; the individual templates within the cluster are compared to the image only if the error is below the threshold. This clustering is done at various levels, resulting in

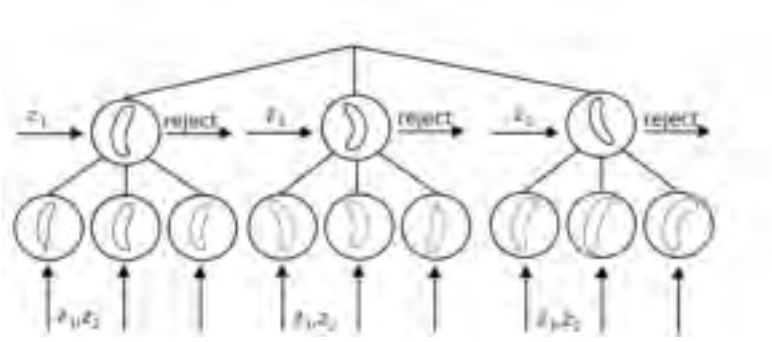


Figure 5.11: *The putative poses of the object, e.g. a banana, together with their likelihood are found using a cascade of classifiers. A tree structure is used to prune away the bad poses by thresholding on the chamfer distance. The texture (i.e. z_2) is measured only at the last level of the tree since the outline shape (i.e. z_1) is sufficient to discard poses with low likelihood.*

a hierarchy, with the templates at the leaf level covering the space of all possible templates (see Fig. 5.11).

In our experiments, we constructed a 3-level tree by clustering the templates using a cost function based on chamfer distance. We use 20 exemplars for each object. The templates are generated by transforming the exemplars using discrete rotations between $-\pi/4$ and $\pi/4$ radians in intervals of 0.1 radians and scales between 0.7 and 1.3 in intervals of 0.1.

The edge image of $\mathbf{D}$ is found using edge detection with embedded confidence [62] (a variation on Canny in which a confidence measure is computed from an ideal edge template). The feature z_1 (truncated chamfer distance) is computed efficiently by using a distance transform of the edge image. This transformation assigns to each pixel in the edge image, the minimum of τ_1 and the distance to its nearest edge pixel. The truncated chamfer distance of an exemplar at an image pose $\mathbf{t}_o = \{x_o, y_o, \phi_o, \rho_o\}$ is calculated efficiently as the mean of the distance transform values at the template point coordinates (using the template defined by rotation ϕ_o and scale ρ_o of the exemplar, see Fig. 5.12).

The feature z_2 (i.e. texture) is computed only at level 3 of the tree cascade by determining the nearest neighbour of the histogram of texton labelling of $\mathbf{D}_o$ among the histograms of texture exemplars. For this purpose, we use the efficient nearest neighbour method described in [32] (modified for χ^2 distance instead of Euclidean distance).

Associated with each node of the cascade is a threshold used to reject bad poses. The putative poses $\mathbf{t}_o$ of the object are found by rejecting bad poses by traversing through the tree cascade starting from the root node for each pixel $\{x, y\}$ of the image $\mathbf{D}$. The likelihoods $\Pr(\mathbf{Z}|\mathbf{\Omega})$ are computed using equation (5.4.1). The initial estimate of the pose is determined by the image location $\{x_o, y_o\}$, template orientation ϕ_o and template scale ρ_o which results in the high-

(a) (b) (c)

Figure 5.12: **(a)** *Original image containing bananas in a cluttered scene.* **(b)** *Edgemap of the image.* **(c)** *The distance transform of the edgemap along with an exemplar of banana. Brighter intensity values indicate points which are far away from the edges. Truncated chamfer distance for the exemplar is calculated as the mean of the distance transform values at the exemplar point coordinates.*

Figure 5.13: *The first column shows the initial estimate obtained for the pose of a banana in two images (see § 5.5.1.1). Samples of the model obtained using the* RGB *distribution of foreground and background are shown in the second and third column (see § 5.5.1.3). The detected poses are shown overlaid on the image. The fourth column shows the segmentation obtained using the* ObjCut *algorithm.*

est likelihood. Fig. 5.13 (column 1) shows the initial estimate for two banana images. This estimate is refined using a better appearance model as described below.

5.5.1.2 Refinement of pose

Once the initial estimate of the pose of the object is obtained, the object location is used to estimate the RGB distribution of the foreground and background (and texture exemplars are no longer used). These distributions, denoted as $\mathcal{H}_{obj}$ and $\mathcal{H}_{bkg}$ for the foreground and background respectively, are used to define a better appearance feature z_2, which is specific to the particular instance of the object category in the image. Specifically,

$$\Pr(z_2) = \prod_{\mathbf{x} \in \mathbf{D}_o} \frac{\Pr(\mathbf{x}|\mathcal{H}_{obj})}{\Pr(\mathbf{x}|\mathcal{H}_{bkg})}. \tag{5.5.1}$$

The refined estimate of the putative poses are obtained using the tree cascade of classifiers as described in § 5.5.1.1 by searching around the initial estimate. In our experiments, we consider locations $\{x, y\}$ which are at most at a distance of 15% of the size of the object as given by the initial estimate. When obtaining the refined estimate, all orientations ϕ and scales ρ are considered at each location $\{x, y\}$.

5.5.1.3 Obtaining samples of the SOE

We now obtain samples from the distribution $g(\mathbf{Z}|\mathbf{\Omega})$ for the SOE model. By assuming a uniform prior $\Pr(\mathbf{\Omega})$ for the model parameter $\mathbf{\Omega}$, this distribution is given by $g(\mathbf{Z}|\mathbf{\Omega}) \propto \Pr(z_1)\Pr(z_2)$. The samples are defined as the best s matches found in § 5.5.1.2 and are obtained by simply sorting over the various matches at all possible locations of the image $\mathbf{D}$. Fig. 5.13 (second and third column) shows some of the samples obtained using the above method for two banana images.

Next, we describe how to sample the distribution $g(\mathbf{Z}|\mathbf{\Omega})$ for an LPS model in the case of articulated object categories.

5.5.2 Sampling the LPS

When matching the LPS model to the image, the number of labels n_L per part has the potential to be very large. Consider the discretization of the putative poses $\mathbf{t} = \{x, y, \phi, \rho\}$ into 360×240 for $\{x, y\}$ with 15 orientations and 7 scales at each location. This results in 9,072,000 poses which causes some computational difficulty when obtaining the samples of the LPS.

Felzenszwalb and Huttenlocher [23] advocate maintaining all labels and suggest an $O(n_P n_L)$ algorithm for finding the samples of the PS by restricting the form of the prior $\exp(-\alpha(\mathbf{t}_i, \mathbf{t}_j))$ in equation (5.4.8). In their work, priors are specified by normal distributions. However, this approach would no longer be computationally feasible as the number of parameters used to represent a pose $\mathbf{t}_i$ increase (e.g. 6 parameters for affine or 8 parameters for projective).

In our approach, we consider the same amount of discretization as in [23] when we are finding candidate poses. However, as noted in § 5.4.2, using a strong shape and appearance model along with discriminative features allows us to consider only a small number of putative poses, n_L, per part by discarding the poses with low likelihood. We found that using a few hundred poses per part, instead of the millions of poses used in [23], was sufficient. The samples are found by a novel efficient algorithm of complexity $O(n_P n'_L)$ (where $n'_L \ll n_L^2$) which generalizes the method described in [24] to non-regular Potts model. Our approach is efficient even for affine and projective transformations due to the small number of putative poses n_L. We now described the three stages for obtaining samples of the LPS.

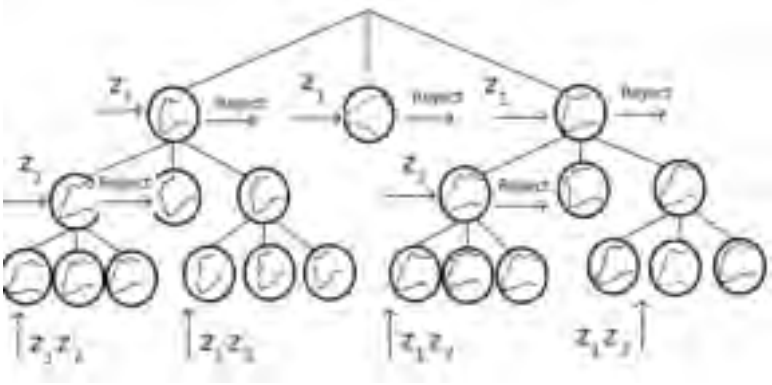


Figure 5.14: *The putative poses of a part, e.g. the head, together with their likelihood are found using a cascade of classifiers. Similar to the cascade shown in Fig. 5.11, a tree structure is used to prune away the bad poses. The texture (i.e. z_2) is measured only at the last level of the tree.*

5.5.2.1 Initial estimation of poses

We find the initial estimate of the poses of the LPS for an image $\mathbf{D}$ by first obtaining the putative poses for each part (along with the corresponding likelihoods) and then estimating posteriors of the putative poses. Note that we do not use occlusion numbers of the parts during this stage.

The putative poses are found using a tree cascade of classifiers for each part as described in § 5.5.1.1 (see Fig. 5.14). The first feature z_1 is computed using a 3-level tree cascade of classifiers for each part. Similar to the first stage of matching the SOE model, the appearance feature z_2 is computed using texture exemplars $\mathcal{T}$ of the object category at the third level of the tree cascade. Note that at this stage the RGB distributions $\mathcal{H}_{obj}$ and $\mathcal{H}_{bkg}$ for the foreground and background are not known. Hence, the feature z_2 is computed using only texture exemplars to overcome intra-class variation in appearance.

Next, an initial estimate of the model is obtained by computing the marginals of the putative poses. Note that, unlike the SOE model, LPS provides a prior over the relative poses of the parts which needs to be considered while computing the marginals. The pose of each part in the initial estimate is given by the putative pose which has the highest marginal probability.

We use sum-product BP to find the marginal probability of part p_i taking a label $\mathbf{t}_i$. Recall that the time complexity of sum-product BP is $O(n_P n_L^2)$ which makes it inefficient for large n_L. However, we take advantage of the fact that the pairwise potentials of the LPS are given by a non-regular Potts model (as shown in equation (5.4.9)). This allows us to reduce the time complexity of sum-product BP to $O(n_P n'_L)$, where $n'_L \ll n_L^2$, using the efficient sum-product BP algorithm described in Appendix A.

The beliefs for each part p_i and putative pose $\mathbf{t}_i$ computed using sum-product BP (denoted by $b_i(\mathbf{t}_i)$) allow us to determine the MMSE (minimum mean squared

(a) (b) (c)

Figure 5.15: **(a)** *An image containing a cow.* **(b)** *Initial estimate of poses of the parts obtained when they are connected using a tree structure. Note that the torso and the forelegs are not localized properly.* **(c)** *Result obtained using a complete graph. The connections between the half limbs and the torso provide better localization.*

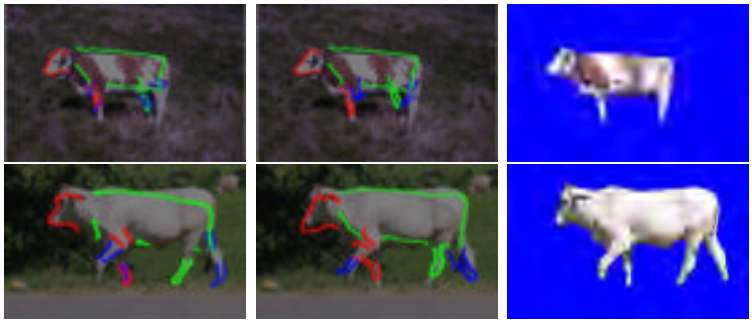


Figure 5.16: *The first column shows the initial estimate obtained for poses of parts of a cow in two images (see § 5.5.2.1). The half-limbs tend to overlap since occlusion numbers are not used. Refined estimates of the poses obtained using the* RGB *distribution of foreground and background together with the* LPS *model are shown in the second column (see § 5.5.2.2). The parts are shown overlaid on the image. The third column shows the segmentation obtained using the* OBJCUT *algorithm.*

error) estimate of the poses of the parts (by choosing the pose with the highest belief). In addition, it also allows us to compute the beliefs for putative poses of every pair of parts, i.e. $b_{ij}(\mathbf{t}_i, \mathbf{t}_j)$, which is later used for sampling (see section 5.5.2.3). Since the parts are connected to form a complete graph, we tend to find valid configurations of the object. Fig. 5.15 shows the advantage of a complete graph over the tree structure used in [23]. Fig. 5.16 (column 1) shows the initial estimate for two cow images. Note that the occlusion numbers are not used to obtain the initial estimate, as would be the case when using the PS model instead of the LPS model. Hence the half-limbs (which are similar to each other in shape and appearance) tend to overlap. The initial estimate is refined using the layering as described below.

5.5.2.2 Layerwise refinement

Using the initial estimate of the object obtained above, the RGB distribution of the foreground (i.e. $\mathcal{H}_{obj}$) and the background (i.e. $\mathcal{H}_{bkg}$) is estimated. A better appearance feature z_2 (i.e. specific to the particular instance of the object category in the images) is now computed as shown in equation (5.5.1). The refined estimate of the poses are obtained by compositing the parts of the LPS in descending order of their occlusion numbers as follows. When considering the layer with occlusion number o, putative poses of the parts p_j such that $o_j = o$ are found using the tree cascade of classifiers around the initial estimate of p_j. In our experiments, we consider locations $\{x, y\}$ which are at most at a distance of 15% of the size of the part as given by the initial estimate. At each location, all possible orientations ϕ and scales ρ are considered. When computing the likelihood of the part at a given pose, pixels which have already been accounted for by a previous layer are not used. Again, the beliefs of each putative pose of every part is computed using efficient sum-product BP. Fig. 5.16 (column 2) shows the MMSE estimate obtained using layerwise refinement for two cow images. However, for segmentation we are interested in samples of the LPS which are obtained in the third stage.

5.5.2.3 Obtaining samples of the LPS

We describe the method for sampling by considering only 2 layers (called layer 1 and layer 2). The extension to an arbitrary number of layers is trivial. The basic idea is to sample the parts in descending order of their occlusion numbers. In our case, this would imply that we sample the parts from layer 2 before we sample the parts from layer 1 (since layer 2 can occlude layer 1). Although this method is not optimal, it produces useful samples for segmentation in practice. To obtain a sample $\mathbf{\Omega}_i$, parts belonging to layer 2 are considered first. The beliefs of these parts are computed using efficient sum-product BP. The posterior for sample $\mathbf{\Omega}_i$ is approximated as

$$g(\mathbf{\Omega}_i|\mathbf{Z}) = \frac{\prod_{ij} b_{ij}(\mathbf{t}_i, \mathbf{t}_j)}{\prod_i b_i(\mathbf{t}_i)^{q_i - 1}}, \tag{5.5.2}$$

where q_i is the number of neighbouring parts of p_i. Since we use a complete graph, $q_i = n_P - 1$, for all parts p_i. Note that the posterior is exact only for a singly connected graph. However, using this approximation sum-product BP has been shown to converge to stationary points of the Bethe free energy [114].

The posterior is then sampled for poses, one part at a time (i.e. Gibbs sampling), such that the pose of the part being sampled forms a valid configuration with the poses of the parts previously sampled. The process is repeated to obtain multiple samples $\mathbf{\Omega}_i$ (which do not include the poses of parts belonging to layer 1). This method of sampling is efficient since often very few pairs of poses form a valid configuration. Further, these pairs are pre-computed during the efficient sum-product BP algorithm as described in the Appendix A. The best n_S samples,

Figure 5.17: *Each row shows three samples obtained by matching the* LPS *model of a cow to an image. Beliefs over the putative poses of parts are calculated using sum-product* BP*. The resulting posterior probability is then sampled to obtain instances of the object (see § 5.5.2.3). Note that different half-limbs are detected correctly in different samples.*

with the highest belief, are chosen.

To obtain the poses of parts in layer 1 for sample $\mathbf{\Omega}_i$, we fix the poses of parts belonging to layer 2 as given by $\mathbf{\Omega}_i$. We calculate the posterior over the poses of parts in layer 1 using sum-product BP. We sample this posterior for poses of parts such that they form a valid configuration with the poses of the parts in layer 2 and with those in layer 1 that were previously sampled. As in the case of layer 2, multiple samples are obtained and the best n_S samples are chosen. The process is repeated for all samples $\mathbf{\Omega}_i$ for layer 2, resulting in a total of $s = n_S^2$ samples.

However, computing the likelihood of the parts in layer 1 for each $\mathbf{\Omega}$ is expensive as their overlap with parts in layer 2 needs to be considered. We use an approximation by considering only those poses whose overlap with layer 2 is below a threshold τ_2. Fig. 5.17 shows some of the samples obtained using the above method for the cows in Fig. 5.16. These samples are the input for the ObjCut algorithm.

5.6. Results

We present several results of the ObjCut algorithm and compare it with a state of the art method and ground truth. In all our experiments, we used the same weight values. As will be seen, ObjCut provides reliable segmentation using both: (i) modelled deformations, using a set of exemplars model for non-articulated objects and the LPS model for articulated objects; and (ii) unmodelled deformations, by merging pixels surrounding the detected object into the segmentation via an st-MINCUT operation.

The results for non-articulated objects are shown for two categories: bananas

and oranges. Fig. 5.13 (column 4) shows the results of the ObjCut algorithm for two banana images. Fig. 5.18 show the segmentations obtained for images containing oranges. Note that the samples of the soe model correctly localize the object in the image. The distinctive shape and appearance of the object then allows us to obtain an accurate segmentation using a single st-mincut operation.

We also tested the ObjCut algorithm on two articulated object categories: cows and horses. Fig 5.16 (column 3) shows the results of our approach for two cow images. Fig. 5.19 and Fig. 5.20 show the segmentation of various images of cows and horses respectively. The 8 cow images and 5 horse images were manually segmented to obtain ground truth for comparison. For the cow images, out of the 125,362 foreground pixels and 472,670 background pixels present in the ground truth, 120,127 (95.82%) and 466,611 (98.72%) were present in the segmentations obtained. Similarly, for the horse images, out of the 79,860 foreground pixels and 151,908 background pixels present in the ground truth, 71,397 (89.39%) and 151,185 (99.52%) were obtained in the segmentations computed by our approach. In the case of horses, most errors are due to unmodelled mane and tail parts. Results indicate that, by considering both modelled and unmodelled deformations, excellent segmentations were obtained by ObjCut.

Figure 5.21 shows a comparison of the segmentation results obtained when using ObjCut with a state of the art method for object category specific segmentation proposed by Leibe and Schiele [58]. Note that a similar approach was described in [10]. The ObjCut algorithm provides better segmentations using a significantly smaller number of exemplars. It achieves this by exploiting the ability of st-mincut for providing excellent segmentations using a good initialization obtained by the object category model.

Figure 5.22 shows the effects of using only the shape potential $\theta^S_{a;f(a)}$ and only the appearance potential $\theta^A_{a;f(a)}$ by discarding the other completely. Results indicate that good segmentations depend on combining both the potentials, as is the case with the ObjCut algorithm.

5.7. Discussion

The approach presented in this work overcomes the problems of previous methods. Specifically, it efficiently provides accurate segmentation which resembles the object. The accuracy of the segmentation can be attributed to the novel probabilistic model, i.e. Object Category Specific cdrf. Object Category Specific cdrf combines the grid cdrf models previously used with an object category model. While the grid cdrf provides bottom-up information, the object category model incorporates top-down information about the shape of the object. For non-articulated object categories, we showed how a set of exemplars model can be effectively used to obtain good segmentations. For articulated object categories,

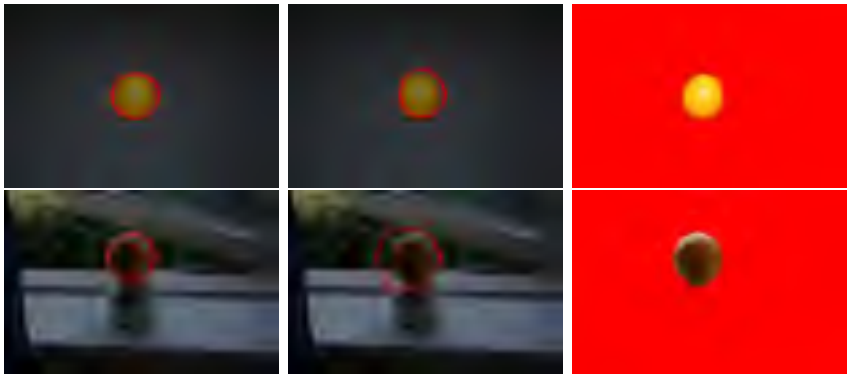


Figure 5.18: *Image segmentation results 1. The* SOE *model for oranges was used to obtain segmentations of previously unseen orange images. The first two images in each column show some of the samples of the* SOE *model. The segmentation obtained using the* OBJCUT *algorithm is shown in the last column.*

we proposed a novel extension of the pictorial structures model which is suitable for the segmentation application.

We presented a three stage approach to match the object category models of our choice to a given image. In the first stage, a rough initial estimate is obtained while taking into account the intra-class variation in shape, appearance and spatial layout. In the second stage, the initial estimate is refined using more descriptive features (i.e. features that are more specific to that instance of the object category) and the layering information (in the case of articulated objects). In the third stage, the samples of the object category model are obtain which are then used to segment the image.

The efficiency of the method is due to two reasons: (i) we showed how the samples of the object category models of our choice can be quickly obtained using a tree cascade of classifiers and efficient sum-product BP; and (ii) our observation that, within the EM framework, the complete data log-likelihood can be optimized using a single st-MINCUT.

However, our method may not scale well when the number of exemplars is huge, e.g. when we want to handle multiple object categories simultaneously. In such cases, the advantage of using feature sharing methods such as [96] within our approach needs to be explored.

Currently, the shape potential provided by the object category model $\mathbf{\Omega}$ is incorporated as a unary term in the Object Category Specific CDRF. An interesting direction for future work would be to use higher order clique potentials provided by $\mathbf{\Omega}$. Some promising work in this area [43] already seems to indicate that vast improvements are possible by using more complex potentials.

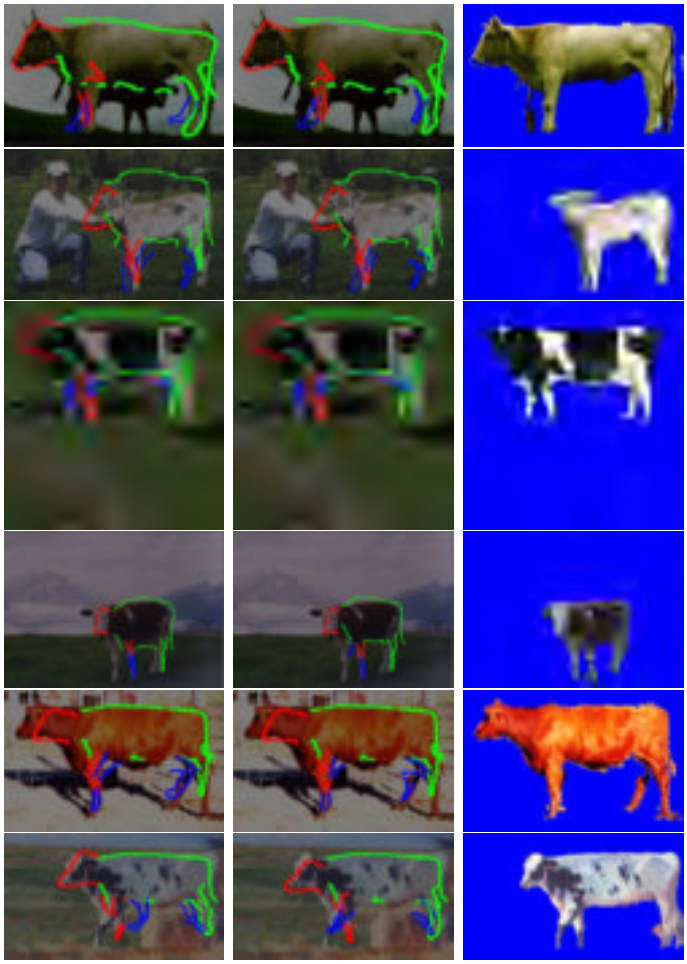


Figure 5.19: *Image segmentation results 2. The first two images in each row show some of the samples of the* LPS *model. The segmentation obtained using the Object Category Specific* CDRF *is shown in the last column. Most of the errors were caused by the tail (which was not a part of the* LPS *model) and parts of the background which were close and similar in colour to the object.*

Figure 5.20: *Image segmentation results 3. The* LPS *model for the horse learnt using manually segmented images was used to obtain the labelling of previously unseen images. Most of the errors were caused by unmodelled parts i.e. the mane and the tail.*

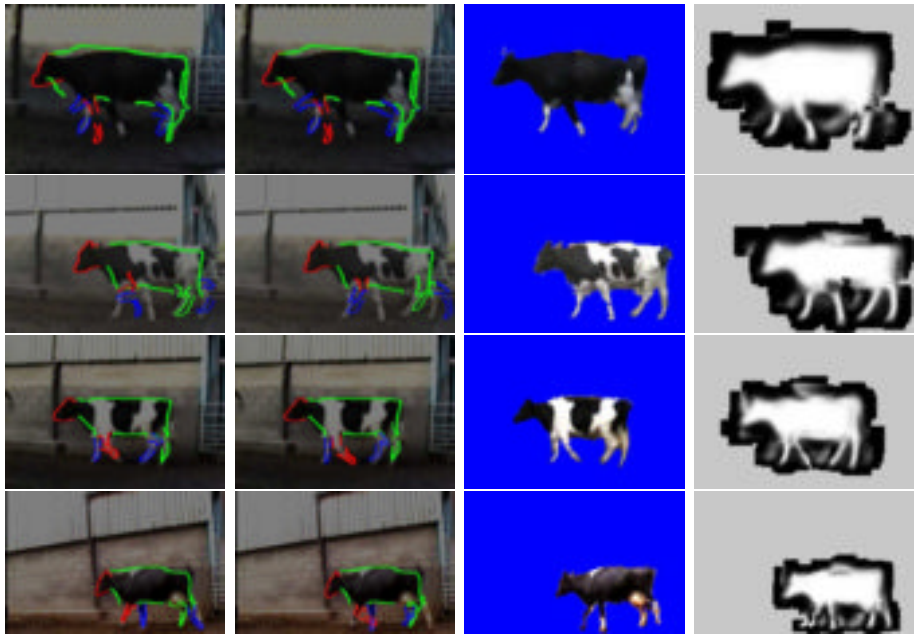


Figure 5.21: *Comparison with Leibe and Schiele. The first two images of each row show some of the samples obtained by matching the* LPS *model to the image. The third image is the segmentation obtained using the* OBJCUT *algorithm. The fourth image shows the result obtained using [58] (provided by the authors). Note that* OBJCUT *provides a better segmentation of the torso and head without detecting extra half limbs.*

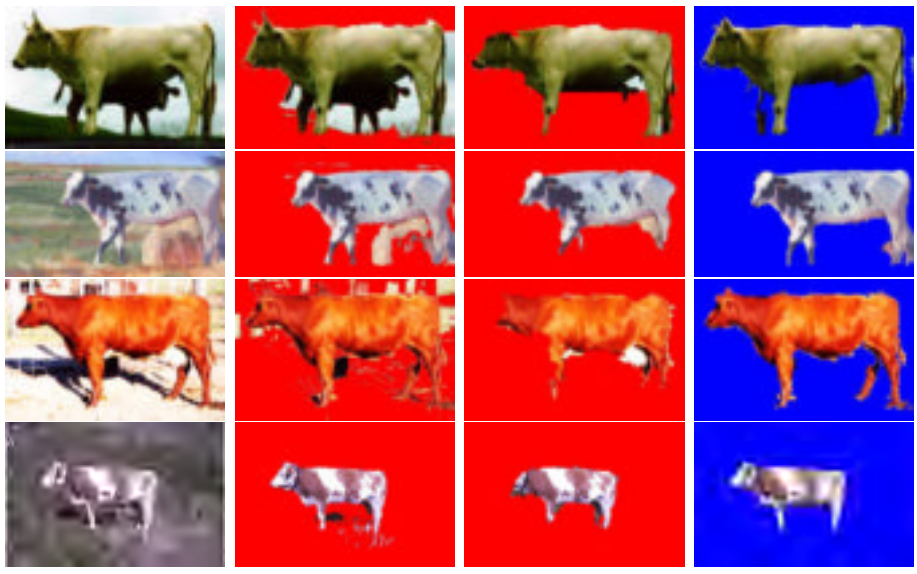


Figure 5.22: *Effects of shape and appearance potentials. The first column of each row shows an image containing a cow. The segmentation results obtained by using only the* RGB *histograms for the foreground and the background provided by the* LPS *model are shown in the second column. The results obtained by using only the shape potential provided by the* LPS *model is shown in the third column. The fourth column shows the segmentations we get using the* OBJCUT *algorithm. Results indicate that good segmentation is obtained only when both shape and appearance potentials are used.*

Chapter 6

An Analysis of Convex Relaxations

The problem of obtaining the maximum a posteriori estimate of a general discrete random field (i.e. a random field defined using a finite and discrete set of labels) is known to be NP-hard. However, due to its central importance in many applications, several approximate algorithms have been proposed in the literature. In this chapter, we present an analysis of three such algorithms based on convex relaxations which were described in chapter 3. Specifically, we consider (i) LP-S: the linear programming (LP) relaxation proposed by Schlesinger [79] for a special case and independently in [17, 51, 101] for the general case; (ii) QP-RL: the quadratic programming (QP) relaxation by Ravikumar and Lafferty [75]; and (iii) SOCP-MS: the second order cone programming (SOCP) relaxation first proposed by Muramatsu and Suzuki [64] for two label problems and later extended in [55] for a general label set.

We show that the SOCP-MS and the QP-RL relaxations are equivalent. Furthermore, we prove that despite the flexibility in the form of the constraints/objective function offered by QP and SOCP, the LP-S relaxation *strictly dominates* (i.e. provides a better approximation than) QP-RL and SOCP-MS. We generalize these results by defining a large class of SOCP (and equivalent QP) relaxations which is dominated by the LP-S relaxation. Based on these results we propose some novel SOCP relaxations which strictly dominate the previous approaches.

6.1. Introduction

Discrete random fields (i.e. random fields defined using a finite and discrete set of labels) are a powerful tool to obtain a probabilistic formulation for various applications in Computer Vision and related areas [15, 18]. Hence, developing accurate and efficient algorithms for performing inference on a given discrete random field is of fundamental importance. In this chapter, we will focus on the problem of maximum a posteriori (MAP) estimation. As seen in the previous chapters, MAP estimation allows us to obtain solutions to many applications such as motion segmentation and image segmentation (for more examples such as stereo and image stitching, see [89]). Further, MAP estimation on discrete random fields is closely related to many important Combinatorial Optimization problems such as : (i) MAXCUT [31] which can be seen as an MAP estimation problem with two putative labels and uniform unary potentials; (ii) multi-way cut [21] (MAP estimation with Potts model pairwise potentials); (iii) metric labelling [15, 42] (MAP estimation with metric pairwise potentials); and (iv) 0-extension [15, 39] (MAP estimation with semi-metric pairwise potentials).

Recall that, given data $\mathbf{D}$, a discrete random field models the distribution (i.e. either the joint or the conditional probability) of a labelling of a set of random variables. Each of these variables $\mathbf{v} = \{v_0, v_1, \cdots, v_{n-1}\}$ can take a label from a discrete set $\mathbf{l} = \{l_0, l_1, \cdots, l_{h-1}\}$. A particular labelling of variables $\mathbf{v}$ is specified by a function f such that

$$f : \{0, 1, \cdots, n-1\} \rightarrow \{0, 1, \cdots, h-1\}, \tag{6.1.1}$$

i.e. variable v_a takes label $l_{f(a)}$. For convenience, we assume the model to be a CRF while noting that all the results of this chapter also apply to MRFs.

Within the CRF framework, the conditional probability of a labelling f given data $\mathbf{D}$ is specified as

$$\Pr(f|\mathbf{D}, \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \exp(-Q(f; \mathbf{D}, \boldsymbol{\theta}). \tag{6.1.2}$$

Here $\boldsymbol{\theta}$ represents the parameters of the CRF, $Z(\boldsymbol{\theta})$ is the partition function and the energy $Q(f; \mathbf{D}, \boldsymbol{\theta})$ is given by

$$Q(f; \mathbf{D}, \boldsymbol{\theta}) = \sum_{v_a \in \mathbf{v}} \theta^1_{a;f(a)} + \sum_{(a,b) \in \mathcal{E}} \theta^2_{ab;f(a)f(b)}. \tag{6.1.3}$$

Recall that $\mathcal{E}$ specifies the neighbourhood relationship of the CRF. The terms $\theta^1_{a;f(a)}$ and $\theta^2_{ab;f(a)f(b)}$ are called the unary and pairwise potentials respectively. For simplicity, we assume that $\theta^2_{ab;f(a)f(b)} = w(a,b)d(f(a), f(b))$ where $w(a,b)$ is the weight that indicates the strength of the pairwise relationship between variables v_a and v_b, with $w(a,b) = 0$ if $(a,b) \notin \mathcal{E}$, and $d(\cdot,\cdot)$ is a distance function

on the labels[1]. As will be seen later, this formulation of the pairwise potentials would allow us to concisely describe our results.

We note that a subclass of this problem where $w(a,b) \geq 0$ and the distance function $d(\cdot,\cdot)$ is a semi-metric or a metric has been well-studied in the literature [15, 17, 42]. However, we will focus on the general MAP estimation problem. In other words, unless explicitly stated, we do not place any restriction on the form of the unary and pairwise potentials.

The problem of MAP estimation is well known to be NP-hard in general. Since it plays a central role in several applications, many approximate algorithms have been proposed in the literature (see chapters 2 and 3). In this chapter, we analyze three such algorithms which are based on convex relaxations of the integer programming (IP) formulation (described in § 3.3.1). Specifically, we consider: (i) LP-S, the linear programming (LP) relaxation of [17, 51, 79, 101] (see § 3.3.2); (ii) QP-RL, the quadratic programming (QP) relaxation of [75] (see § 3.3.3); and (iii) SOCP-MS, the second order cone programing (SOCP) relaxation of [55, 64] (see § 3.3.5). In order to compare these relaxations, we require the following definitions.

6.1.1 Comparing Relaxations

We say that a relaxation A *dominates* the relaxation B (alternatively, B is dominated by A) if and only if

$$\min_{(\mathbf{x},\mathbf{X})\in\mathcal{F}(\mathrm{A})} e(\mathbf{x},\mathbf{X};\boldsymbol{\theta}) \geq \min_{(\mathbf{x},\mathbf{X})\in\mathcal{F}(\mathrm{B})} e(\mathbf{x},\mathbf{X};\boldsymbol{\theta}), \forall \boldsymbol{\theta}, \tag{6.1.4}$$

where $\mathcal{F}(\mathrm{A})$ and $\mathcal{F}(\mathrm{B})$ are the feasibility regions of the relaxations A and B respectively. The term $e(\mathbf{x},\mathbf{X};\boldsymbol{\theta})$ denotes the value of the objective function at $(\mathbf{x},\mathbf{X})$ (i.e. the energy of the possibly fractional labelling $(\mathbf{x},\mathbf{X})$) for the MAP estimation problem defined over the CRF with parameter $\boldsymbol{\theta}$. Thus the optimal value of the dominating relaxation A is always greater than or equal to the optimal value of relaxation B. We note here that the concept of domination has been used previously in [17] (to compare LP-S with the linear programming relaxation in [42]).

Relaxations A and B are said to be *equivalent* if A dominates B and B dominates A, i.e. their optimal values are equal to each other for all CRFs. A relaxation A is

[1]The pairwise potentials for any CRF can be represented in the form $\theta^2_{ab;ij} = w(a,b)d(i,j)$. This can be achieved by using a larger set of labels $\hat{\mathbf{l}} = \{l_{0;0}, \cdots, l_{0;h_1}, \cdots, l_{n-1;h_1}\}$ such that the unary potential of v_a taking label $l_{b;i}$ is $\theta^1_{a;i}$ if $a = b$ and ∞ otherwise. In other words, a variable v_a can only take labels from the set $\{l_{a;0}, \cdots, l_{a;h-1}\}$ since all other labels will result in an energy value of ∞. The pairwise potential for variables v_a and v_b taking labels $l_{a;i}$ and $l_{b;j}$ respectively can then be represented in the form $w(a,b)d(a;i,b;j)$ where $w(a,b) = 1$ and $d(a;i,b;j) = \theta^2_{ab;ij}$. Note that using a larger set of labels $\hat{\mathbf{l}}$ will increase the time complexity of MAP estimation algorithms, but does not effect the analysis presented in this chapter.

said to *strictly dominate* relaxation B if A dominates B but B does not dominate A. In other words there exists at least one CRF with parameter $\boldsymbol{\theta}$ such that

$$\min_{(\mathbf{x},\mathbf{X})\in\mathcal{F}(\mathrm{A})} e(\mathbf{x},\mathbf{X};\boldsymbol{\theta}) > \min_{(\mathbf{x},\mathbf{X})\in\mathcal{F}(\mathrm{B})} e(\mathbf{x},\mathbf{X};\boldsymbol{\theta}). \tag{6.1.5}$$

Note that, by definition, the optimal value of any relaxation would always be less than or equal to the energy of the optimal (i.e. the MAP) labelling. Hence, the optimal value of a strictly dominating relaxation A is closer to the optimal value of the MAP estimation IP compared to that of relaxation B. In other words, A provides a better approximation for MAP estimation than B.

We now describe two special cases of domination which are used extensively in the remainder of this chapter.

Case I: Consider two relaxations A and B which share a common objective function. For example, the objective functions of the LP-S and the SOCP-MS relaxations described in chapter 3 have the same form. Further, let A and B differ in the constraints that they specify such that $\mathcal{F}(\mathrm{A}) \subseteq \mathcal{F}(\mathrm{B})$, i.e. the feasibility region of A is a subset of the feasibility region of B.

Given two such relaxations, we claim that A dominates B. This can be proved by contradiction. To this end, we assume that A does not dominate B. Therefore, by definition of domination, there exists at least one parameter $\boldsymbol{\theta}$ for which B provides a greater value of the objective function than A. Let an optimal solution of A be $(\mathbf{x}_A, \mathbf{X}_A)$. Similarly, let $(\mathbf{x}_B, \mathbf{X}_B)$ be an optimal solution of B. By our assumption, the following holds true:

$$e(\mathbf{x}_A, \mathbf{X}_A; \boldsymbol{\theta}) < e(\mathbf{x}_B, \mathbf{X}_B; \boldsymbol{\theta}). \tag{6.1.6}$$

However, since $\mathcal{F}(\mathrm{A}) \subseteq \mathcal{F}(\mathrm{B})$ it follows that $(\mathbf{x}_A, \mathbf{X}_A) \in \mathcal{F}(\mathrm{B})$. Hence, from equation (6.1.6), we see that $(\mathbf{x}_B, \mathbf{X}_B)$ cannot be an optimal solution of B. This proves our claim.

We can also consider a case where $\mathcal{F}(\mathrm{A}) \subset \mathcal{F}(\mathrm{B})$, i.e. the feasibility region of A is a strict subset of the feasibility region of B. Using the above argument we see that A dominates B. Further, assume that there exists a parameter $\boldsymbol{\theta}$ such that the intersection of the set of all optimal solutions of A and the set of all optimal solutions of B is null. In other words if $(\mathbf{x}_B, \mathbf{X}_B)$ is an optimal solution of B then $(\mathbf{x}_B, \mathbf{X}_B) \notin \mathcal{F}(\mathrm{A})$. Clearly, if such a parameter $\boldsymbol{\theta}$ exists then A strictly dominates B.

Case II: Consider two relaxations A and B such that they share a common objective function. Further, let the constraints of B be a subset of the constraints of A. We claim that A dominates B. This follows from the fact that $\mathcal{F}(\mathrm{A}) \subseteq \mathcal{F}(\mathrm{B})$ and the argument used in Case I above.

6.1.2 Our Results

We prove that LP-S strictly dominates SOCP-MS (see section 6.2). Further, in section 6.3, we show that QP-RL is equivalent to SOCP-MS. This implies that LP-S strictly dominates the QP-RL relaxation. In section 6.4 we generalize the above results by proving that a large class of SOCP (and equivalent QP) relaxations is dominated by LP-S.

Based on these results, we propose a novel set of constraints which result in SOCP relaxations that dominate LP-S, QP-RL and SOCP-MS. These relaxations introduce SOC constraints on cycles and cliques formed by the neighbourhood relationship of the CRF. In chapter 7, we propose approximate algorithms for solving these relaxations and show that the empirical results conform with our analysis.

6.2. LP-S vs. SOCP-MS

We now show that for the MAP estimation problem the linear constraints of LP-S, i.e.

$$\mathbf{x} \in [-1,1]^{nh}, \mathbf{X} \in [-1,1]^{nh \times nh}, \tag{6.2.1}$$
$$\textstyle\sum_{l_i \in \mathbf{l}} x_{a;i} = 2 - h, \tag{6.2.2}$$
$$\textstyle\sum_{l_j \in \mathbf{l}} X_{ab;ij} = (2-h) x_{a;i}, \tag{6.2.3}$$
$$X_{ab;ij} = X_{ba;ji}, \tag{6.2.4}$$
$$1 + x_{a;i} + x_{b;j} + X_{ab;ij} \geq 0. \tag{6.2.5}$$

are stronger than the SOCP-MS constraints, i.e.

$$\mathbf{x} \in [-1,1]^{nh}, \mathbf{X} \in [-1,1]^{nh \times nh}, \tag{6.2.6}$$
$$\textstyle\sum_{l_i \in \mathbf{l}} x_{a;i} = 2 - h, \tag{6.2.7}$$
$$(x_{a;i} - x_{b;j})^2 \leq 2 - 2X_{ab;ij}, \tag{6.2.8}$$
$$(x_{a;i} + x_{b;j})^2 \leq 2 + 2X_{ab;ij}, \tag{6.2.9}$$
$$X_{ab;ij} = X_{ba;ji}, \tag{6.2.10}$$
$$1 + x_{a;i} + x_{b;j} + X_{ab;ij} \geq 0. \tag{6.2.11}$$

In other words the feasibility region of LP-S is a strict subset of the feasibility region of SOCP-MS (i.e. $\mathcal{F}(\text{LP-S}) \subset \mathcal{F}(\text{SOCP-MS})$). This in turn would allow us to prove the following theorem.

Theorem 1: The LP-S relaxation strictly dominates the SOCP-MS relaxation.

Proof: The LP-S and the SOCP-MS relaxations differ only in the way they relax the non-convex constraint $\mathbf{X} = \mathbf{x}\mathbf{x}^\top$. While LP-S relaxes $\mathbf{X} = \mathbf{x}\mathbf{x}^\top$ using the marginalization constraint (6.2.3), SOCP-MS relaxes it to constraints (6.2.8)

and (6.2.9). The SOCP-MS constraints provide the supremum and infimum of $X_{ab;ij}$ as

$$\frac{(x_{a;i}+x_{b;j})^2}{2}-1\leq X_{ab;ij}\leq 1-\frac{(x_{a;i}-x_{b;j})^2}{2}. \tag{6.2.12}$$

Consider a pair of neighbouring variables v_a and v_b and a pair of labels l_i and l_j. Recall that SOCP-MS specifies the constraints (6.2.8) and (6.2.9) for all such pairs of random variables and labels, i.e. for all $x_{a;i}, x_{b;j}, X_{ab;ij}$ such that $(a,b)\in\mathcal{E}$ and $l_i, l_j \in \mathbf{l}$. In order to prove this theorem we use the following two lemmas.

Lemma 2.1: If $x_{a;i}$, $x_{b;j}$ and $X_{ab;ij}$ satisfy the LP-S constraints, i.e. constraints (6.2.1)-(6.2.5), then

$$|x_{a;i}-x_{b;j}|\leq 1-X_{ab;ij}. \tag{6.2.13}$$

The above result holds true for all $(a,b)\in\mathcal{E}$ and $l_i, l_j\in\mathbf{l}$.

Proof: From the LP-S constraints, we get

$$\begin{aligned}\frac{1+x_{a;i}}{2}&=\sum_{l_k\in\mathbf{l}}\frac{1+x_{a;i}+x_{b;k}+X_{ab;ik}}{4},\\ \frac{1+x_{b;j}}{2}&=\sum_{l_k\in\mathbf{l}}\frac{1+x_{a;k}+x_{b;j}+X_{ab;kj}}{4}.\end{aligned} \tag{6.2.14}$$

Therefore,

$$\begin{aligned}|x_{a;i}-x_{b;j}| &= 2\left|\tfrac{1+x_{a;i}}{2}-\tfrac{1+x_{b;j}}{2}\right|,\\ &= 2\left|\textstyle\sum_{l_k\in\mathbf{l}}\tfrac{1+x_{a;i}+x_{b;k}+X_{ab;ik}}{4}-\sum_{l_k\in\mathbf{l}}\tfrac{1+x_{a;k}+x_{b;j}+X_{ab;kj}}{4}\right|,\\ &= 2\left|\textstyle\sum_{l_k\in\mathbf{l},k\neq j}\tfrac{1+x_{a;i}+x_{b;k}+X_{ab;ik}}{4}-\sum_{l_k\in\mathbf{l},k\neq i}\tfrac{1+x_{a;k}+x_{b;j}+X_{ab;kj}}{4}\right|,\\ &\leq 2\left(\textstyle\sum_{l_k\in\mathbf{l},k\neq j}\tfrac{1+x_{a;i}+x_{b;k}+X_{ab;ik}}{4}+\sum_{l_k\in\mathbf{l},k\neq i}\tfrac{1+x_{a;k}+x_{b;j}+X_{ab;kj}}{4}\right),\\ &= 2\left(\tfrac{1+x_{a;i}}{2}-\tfrac{1+x_{a;i}+x_{b;j}+X_{ab;ij}}{4}+\tfrac{1+x_{b;j}}{2}-\tfrac{1+x_{a;i}+x_{b;j}+X_{ab;ij}}{4}\right),\\ &= 1-X_{ab;ij}. \quad \blacksquare\end{aligned} \tag{6.2.15}$$

Using the above lemma, we get

$$(x_{a;i}-x_{b;j})^2\leq(1-X_{ab;ij})(1-X_{ab;ij}), \tag{6.2.16}$$

$$\Rightarrow\quad (x_{a;i}-x_{b;j})^2\leq 2(1-X_{ab;ij}), \tag{6.2.17}$$

$$\Rightarrow\quad (x_{a;i}-x_{b;j})^2\leq 2-2X_{ab;ij}. \tag{6.2.18}$$

The inequality (6.2.17) is obtained using the fact that $-1\leq X_{ab;ij}\leq 1$ and hence, $1-X_{ab;ij}\leq 2$. Using inequality (6.2.16), we see that the necessary condition for the equality to hold true is $(1-X_{ab;ij})(1-X_{ab;ij})=2-2X_{ab;ij}$, i.e. $X_{ab;ij}=-1$. Note that inequality (6.2.18) is equivalent to the SOCP-MS constraint (6.2.8). Thus LP-S provides a smaller supremum of $X_{ab;ij}$ when $X_{ab;ij}>-1$.

Lemma 2.2: If $x_{a;i}$, $x_{b;j}$ and $X_{ab;ij}$ satisfy the LP-S constraints then

$$|x_{a;i}+x_{b;j}|\leq 1+X_{ab;ij}. \tag{6.2.19}$$

This holds true for all $(a,b) \in \mathcal{E}$ and $l_i, l_j \in \mathbf{l}$.

Proof: According to constraint (6.2.5),

$$-(x_{a;i} + x_{b;j}) \leq 1 + X_{ab;ij}. \tag{6.2.20}$$

Using Lemma 2.1, we get the following set of inequalities:

$$|x_{a;i} - x_{b;k}| \leq 1 - X_{ab;ik}, l_k \in \mathbf{l}, k \neq j \tag{6.2.21}$$

Adding the above set of inequalities, we get

$$\sum_{l_k \in \mathbf{l}, k \neq j} |x_{a;i} - x_{b;j}| \leq \sum_{l_k \in \mathbf{l}, k \neq j} (1 - X_{ab;ik}), \tag{6.2.22}$$

$$\Rightarrow \quad \sum_{l_k \in \mathbf{l}, k \neq j} (x_{a;i} - x_{b;k}) \leq \sum_{l_k \in \mathbf{l}, k \neq j} (1 - X_{ab;ik}), \tag{6.2.23}$$

$$\Rightarrow \quad (h-1)x_{a;i} - \sum_{l_k \in \mathbf{l}, k \neq j} x_{b;k} \leq (h-1) - \sum_{l_k \in \mathbf{l}, k \neq j} X_{ab;ik}, \tag{6.2.24}$$

$$\Rightarrow \quad (h-1)x_{a;i} + (h-2) + x_{b;j} \leq (h-1) + (h-2)x_{a;i} + X_{ab;ij}. \tag{6.2.25}$$

The last step is obtained using constraints (6.2.2) and (6.2.3), i.e.

$$\sum_{l_k \in \mathbf{l}} x_{b;k} = (2-h), \sum_{l_k \in \mathbf{l}} X_{ab;ik} = (2-h)x_{a;i}. \tag{6.2.26}$$

Rearranging the terms, we get

$$(x_{a;i} + x_{b;j}) \leq 1 + X_{ab;ij}. \tag{6.2.27}$$

Thus, according to inequalities (6.2.20) and (6.2.27)

$$|x_{a;i} + x_{b;j}| \leq 1 + X_{ab;ij}. \quad \blacksquare \tag{6.2.28}$$

Using the above lemma, we obtain

$$(x_{a;i} + x_{b;j})^2 \leq (1 + X_{ab;ij})(1 + X_{ab;ij}), \tag{6.2.29}$$

$$\Rightarrow \quad (x_{a;i} + x_{b;j})^2 \leq 2 + 2X_{ab;ij}. \tag{6.2.30}$$

where the necessary condition for the equality to hold true is $1 + X_{ab;ij} = 2$ (i.e. $X_{ab;ij} = 1$). Note that the above constraint is equivalent to the SOCP-MS constraint (6.2.9). Together with inequality (6.2.18), this proves that the LP-S relaxation provides smaller supremum and larger infimum of the elements of the matrix $\mathbf{X}$ than the SOCP-MS relaxation. Thus, $\mathcal{F}(\text{LP-S}) \subset \mathcal{F}(\text{SOCP-MS})$.

One can also construct a parameter $\boldsymbol{\theta}$ for which the set of all optimal solutions of SOCP-MS do not lie in the feasibility region of LP-S. In other words the optimal solutions of SOCP-MS belong to the non-empty set $\mathcal{F}(\text{SOCP-MS}) - \mathcal{F}(\text{LP-S})$. Using the argument of Case I in § 6.1.1, this implies that LP-S strictly dominates SOCP-MS. $\blacksquare$

Note that the above theorem does not apply to the variation of SOCP-MS described in [55, 64] which include *triangular inequalities*. However, since triangular inequalities are linear constraints, LP-S can be extended to include them (as will

be seen in the next chapter). The resulting LP relaxation would strictly dominate the SOCP-MS relaxation with triangular inequalities.

6.3. QP-RL vs. SOCP-MS

We now prove that QP-RL and SOCP-MS are equivalent (i.e. their optimal values are equal for MAP estimation problems defined over all CRFs). Specifically, we consider a vector $\mathbf{x}$ which lies in the feasibility regions of the QP-RL and SOCP-MS relaxations, i.e. $\mathbf{x} \in [-1,1]^{nh}$. For this vector, we show that the values of the objective functions of the QP-RL and SOCP-MS relaxations are equal. This would imply that if $\mathbf{x}^*$ is an optimal solution of QP-RL for some CRF with parameter $\boldsymbol{\theta}$ then there exists an optimal solution $(\mathbf{x}^*, \mathbf{X}^*)$ of the SOCP-MS relaxation. Further, if e^Q and e^S are the optimal values of the objective functions obtained using the QP-RL and SOCP-MS relaxation, then $e^Q = e^S$.

Theorem 2: The QP-RL relaxation and the SOCP-MS relaxation are equivalent.

Proof: Recall that the QP-RL relaxation is defined as follows:

$$\text{QP-RL:} \quad \mathbf{x}^* = \arg\min_{\mathbf{x}} \left(\tfrac{\mathbf{1}+\mathbf{x}}{2}\right)^\top \hat{\boldsymbol{\theta}}^1 + \left(\tfrac{\mathbf{1}+\mathbf{x}}{2}\right)^\top \hat{\boldsymbol{\theta}}^2 \left(\tfrac{\mathbf{1}+\mathbf{x}}{2}\right), \tag{6.3.1}$$

$$\text{s.t.} \quad \textstyle\sum_{l_i \in \mathbf{l}} x_{a;i} = 2 - h, \forall v_a \in \mathbf{v}, \tag{6.3.2}$$

$$\mathbf{x} \in \{-1,1\}^{nh}, \tag{6.3.3}$$

where the unary potential vector $\hat{\boldsymbol{\theta}}^1$ and the pairwise potential matrix $\hat{\boldsymbol{\theta}}^2 \succeq 0$ are defined as

$$\hat{\theta}^1_{a;i} = \theta^1_{a;i} - \sum_{v_c \in \mathbf{v}} \sum_{l_k \in \mathbf{l}} |\theta^2_{ac;ik}|, \tag{6.3.4}$$

$$\hat{\theta}^2_{ab;ij} = \begin{cases} \sum_{v_c \in \mathbf{v}} \sum_{l_k \in \mathbf{l}} |\theta^2_{ac;ik}|, & \text{if} \quad a = b, i = j, \\ \theta^2_{ab;ij} & \text{otherwise.} \end{cases}$$

Here, the terms $\theta^1_{a;i}$ and $\theta^2_{ac;ik}$ are the (original) unary potentials and pairwise potentials for the given CRF. Consider a feasible solution $\mathbf{x}$ of the QP-RL and the SOCP-MS relaxations. Further, let $\mathbf{X}$ be the solution obtained when minimizing the objective function of the SOCP-MS relaxation whilst keeping $\mathbf{x}$ fixed. We prove that the value of the objective functions for both relaxations at the above feasible solution is the same by equating the coefficient of $\theta^1_{a;i}$ and $\theta^2_{ab;ij}$ for all $v_a \in \mathbf{v}$, $(a,b) \in \mathcal{E}$ and $l_i, l_j \in \mathbf{l}$ in both formulations. Using equation (6.3.4), we see that $\theta^1_{a;i}$ is multiplied by $\frac{1+x_{a;i}}{2}$ in the objective function of the QP-RL. Similarly, $\theta^1_{a;i}$ is multiplied by $\frac{1+x_{a;i}}{2}$ in the SOCP-MS relaxation. Therefore the coefficients of $\theta^1_{a;i}$ in both relaxations are equal for all $v_a \in \mathbf{v}$ and $l_i \in \mathbf{l}$.

We now consider the pairwise potentials, i.e. $\theta^2_{ab;ij}$ and show that their coefficients are the same when obtaining the minimum of the objective function. We consider the following two cases.

Case I: Let $\theta^2_{ab;ij} = \theta^2_{ba;ji} \geq 0$. Using equation (6.3.5) we see that, in the QP-RL relaxation, $\theta^2_{ab;ij} + \theta^2_{ba;ji}$ is multiplied by the following term:

$$\left(\frac{1+x_{a;i}}{2}\right)^2 + \left(\frac{1+x_{b;j}}{2}\right)^2 + 2\left(\frac{1+x_{a;i}}{2}\right)\left(\frac{1+x_{b;j}}{2}\right) - \frac{1+x_{a;i}}{2} - \frac{1+x_{b;j}}{2}. \tag{6.3.5}$$

In the case of SOCP-MS relaxation, since $\theta^2_{ab;ij} \geq 0$, the minimum of the objective function is obtained by using the minimum value that $X_{ab;ij}$ would take given the SOC constraints. Since $\mathbf{X}$ is symmetric, we see that $\theta^2_{ab;ij} + \theta^2_{ba;ji}$ is multiplied by the following term:

$$\frac{1+x_{a;i}+x_{b;j}+\inf\{X_{ab;ij}\}}{2} \tag{6.3.6}$$

$$= \frac{1+x_{a;i}+x_{b;j}+(x_{a;i}+x_{b;j})^2/2-1}{2}, \tag{6.3.7}$$

where the infimum of $X_{ab;ij}$ is defined by constraint (6.2.9) in the SOCP-MS relaxation. It can easily be verified that the terms (6.3.5) and (6.3.7) are equal.

Case II: Now consider the case where $\theta^2_{ab;ij} = \theta^2_{ba;ji} < 0$. In the QP-RL relaxation, the term $\theta^2_{ab;ij} + \theta^2_{ba;ji}$ is multiplied by

$$\frac{1+x_{a;i}}{2} + \frac{1+x_{b;j}}{2} + 2\left(\frac{1+x_{a;i}}{2}\right)\left(\frac{1+x_{b;j}}{2}\right) - \left(\frac{1+x_{a;i}}{2}\right)^2 - \left(\frac{1+x_{b;j}}{2}\right)^2. \tag{6.3.8}$$

In order to obtain the minimum of the objective function, the SOCP-MS relaxation uses the maximum value that $X_{ab;ij}$ would take given the SOC constraints. Thus, $\theta^2_{ab;ij} + \theta^2_{ba;ji}$ is multiplied by

$$\frac{1+x_{a;i}+x_{b;j}+\sup\{X_{ab;ij}\}}{2} \tag{6.3.9}$$

$$= \frac{1+x_{a;i}+x_{b;j}+1-(x_{a;i}-x_{b;j})^2/2}{2}, \tag{6.3.10}$$

where the supremum of $X_{ab;ij}$ is defined by constraint (6.2.8). Again, the terms (6.3.8) and (6.3.10) can be shown to be equivalent. ∎

Theorems 1 and 2 prove that the LP-S relaxation strictly dominates the QP-RL and SOCP-MS relaxations. A natural question that now arises is whether the additive bound of QP-RL (proved in [75]) is applicable to the LP-S and SOCP-MS relaxations. Our next theorem answers this question in an affirmative. To this end, we use the rounding scheme proposed in [75] to obtain the labelling f for all the random variables of the given CRF. This rounding scheme is summarized below:

- Pick a variable v_a which has not been assigned a label.
- Assign the label l_i to v_a (i.e. f(a) = i) with probability $\frac{1+x_{a;i}}{2}$.
- Continue till all variables have been assigned a label.

Recall that $\sum_{i=0}^{h-1} \frac{1+x_{a;i}}{2} = 1$ for all $v_a \in \mathbf{v}$. Hence, once v_a is picked it is guaranteed to be assigned a label. In other words the above rounding scheme terminates after $n = |\mathbf{v}|$ steps. To the best of our knowledge, this additive bound was previously known only for the QP-RL relaxation [75].

Theorem 3: For the above rounding scheme, LP-S and SOCP-MS provide the same additive bound as the QP-RL relaxation of [75], i.e. $\frac{S}{4}$ where $S = \sum_{(a,b)\in\mathcal{E}} \sum_{l_i,l_j\in\mathbf{l}} |\theta^2_{ab;ij}|$ (i.e. the sum of the absolute values of all pairwise potentials). Furthermore, this bound is tight.

Proof: The QP-RL and SOCP-MS relaxations are equivalent. Thus the above theorem holds true for SOCP-MS. We now consider the LP-S relaxation of [17, 51, 79, 101]. We denote the energy of the optimal labelling as e^*. Recall that e^L denotes the optimal value of the LP-S which is obtained using possibly fractional variables $(\mathbf{x}^*, \mathbf{X}^*)$. Clearly, $e^L \leq e^*$. The energy of the labelling $\hat{\mathbf{x}}$, obtained after rounding the solution of the LP-S relaxation, is represented by the term $\hat{e}^L$,

Using the above notation, we now show that the LP-S relaxation provides an additive bound of $\frac{S}{4}$ for the above rounding scheme. We first consider the unary potentials and observe that

$$E\left(\theta^1_{a;i}\left(\frac{1+\hat{x}_{a;i}}{2}\right)\right) = \theta^1_{a;i}\left(\frac{1+x^*_{a;i}}{2}\right), \tag{6.3.11}$$

where $E(\cdot)$ denotes the expectation of its argument under the above rounding scheme. Similarly, for the pairwise potentials,

$$E\left(\theta^2_{ab;ij}\left(\frac{1+\hat{x}_{a;i}}{2}\right)\left(\frac{1+\hat{x}_{b;j}}{2}\right)\right) = \theta^2_{ab;ij}\left(\frac{1+x^*_{a;i}+x^*_{b;ij}+x^*_{a;i}x^*_{b;j}}{4}\right). \tag{6.3.12}$$

We analyze the following two cases:

(i) $\theta^2_{ab;ij} \geq 0$: Using the fact that $X^*_{ab;ij} \geq |x^*_{a;i} + x^*_{b;j}| - 1$ (see Lemma 2.2), we get

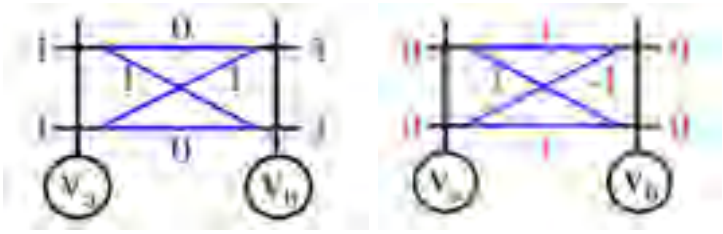
$$\begin{aligned} & 1 + x^*_{a;i} + x^*_{b;j} + x^*_{a;i}x^*_{b;j} - (1 + x^*_{a;i} + x^*_{b;j} + X^*_{ab;ij}) \\ = \;& x^*_{a;i}x^*_{b;j} - X^*_{ab;ij} \\ \leq \;& x^*_{a;i}x^*_{b;j} + 1 - |x^*_{a;i} + x^*_{b;j}| \\ \leq \;& 1, \end{aligned} \tag{6.3.13}$$

where the equality holds when $x^*_{a;i} = x^*_{b;j} = 0$. Therefore,

$$E\left(\theta^2_{ab;ij}\left(\frac{1+\hat{x}_{a;i}}{2}\right)\left(\frac{1+\hat{x}_{b;j}}{2}\right)\right) \leq \theta^2_{ab;ij}\frac{(1+x^*_{a;i}+x^*_{b;ij}+X^*_{ab;ij})}{4} + \frac{|\theta^2_{ab;ij}|}{4}. \tag{6.3.14}$$

(ii) $\theta^2_{ab;ij} < 0$: Using the fact that $X^*_{ab;ij} \leq 1 - |x^*_{a;i} - x^*_{b;j}|$ (see Lemma 2.1), we get

$$\begin{aligned} & 1 + x^*_{a;i} + x^*_{b;j} + x^*_{a;i}x^*_{b;j} - (1 + x^*_{a;i} + x^*_{b;j} + X^*_{ab;ij}) \\ \geq \;& x^*_{a;i}x^*_{b;j} - 1 + |x^*_{a;i} - x^*_{b;j}| \\ \geq \;& -1, \end{aligned} \tag{6.3.15}$$



(a) (b)

Figure 6.1: *An example* CRF *for proving the tightness of the* LP-S *additive bound of* $\frac{S}{4}$. **(a)** *The two random variables* v_a *and* v_b *are shown as unfilled circles. Their two putative labels are shown as branches (i.e. the horizontal lines) of the trellises (i.e. the vertical lines). The value of the unary potential* $\theta^1_{a;i}$ *is shown next to the* i^{th} *branch of the trellis on top of* v_a*. The pairwise potential* $\theta^2_{ab;ij}$ *is shown next to the connection between the* i^{th} *and* j^{th} *branches of the trellises on top of* v_a *and* v_b *respectively. Note that the unary potentials are uniform while the pairwise potentials form an Ising model.* **(b)** *An optimal solution of the* LP-S *relaxation for the* CRF *shown in (a). This solution is shown in red to differentiate it from the potentials shown in (a). The values of the variables* $x_{a;i}$ *are shown next to the* i^{th} *branch of the trellis of* v_a*. Note that all variables* $x_{a;i}$ *have been assigned to* 0*. The values of the variables* $X_{ab;ij}$ *are shown next to the connection between the* i^{th} *and* j^{th} *branch of the trellises of* v_a *and* v_b*. Note that* $X_{ab;ij} = -1$ *if* $\theta^2_{ab;ij} > 0$ *and* $X_{ab;ij} = 1$ *otherwise.*

where the equality holds when $x^*_{a;i} = x^*_{b;j} = 0$. Therefore,

$$E\left(\theta^2_{ab;ij}\left(\frac{1+\hat{x}_{a;i}}{2}\right)\left(\frac{1+\hat{x}_{b;j}}{2}\right)\right) \leq \theta^2_{ab;ij}\frac{(1+x^*_{a;i}+x^*_{b;ij}+X^*_{ab;ij})}{4} + \frac{|\theta^2_{ab;ij}|}{4}. \tag{6.3.16}$$

Summing the expectation of the unary and pairwise potentials for all $v_a \in \mathbf{v}$, $(a,b) \in \mathcal{E}$ and $l_i, l_j \in \mathbf{l}$, we get

$$e^* \leq E(\hat{e}^L) \leq e^L + \frac{S}{4} \leq e^* + \frac{S}{4}, \tag{6.3.17}$$

which proves the additive bound for LP-S.

This additive bound can indeed be shown to be tight by using the following simple example. Consider an instance of the MAP estimation problem for a CRF defined on two variables $\mathbf{v} = \{v_a, v_b\}$ each of which can take one of two labels from the set $\mathbf{l} = \{l_0, l_1\}$. Let the unary and pairwise potentials be as defined in Fig. 6.1(a), i.e. the unary potentials are uniform and the pairwise potentials follow the Ising model.

An optimal solution of the LP-S relaxation is given in Fig. 6.1(b). Clearly, $e^* = 2$ (e.g. for the labelling $f = \{0,0\}$ or $f = \{1,1\}$) while $E(\hat{e}^L) = 2 + \frac{2}{4} = e^* + \frac{S}{4}$. Thus the additive bounds obtained for the LP-S, QP-RL and SOCP-MS algorithms are the same. In fact, one can construct arbitrarily large CRFs (i.e. CRF defined

over a large number of variables) with uniform unary potentials and Ising model pairwise potentials for which the bound can be shown to be tight. We note, however, that better bounds can be obtained for some special cases of the MAP estimation problem using the LP-S relaxation together with more clever rounding schemes (such as those described in [17, 42]). ∎

6.4. QP and SOCP Relaxations over Trees and Cycles

We now generalize the results of Theorem 1 by defining a large class of SOCP relaxations which is dominated by LP-S. Specifically, we consider the SOCP relaxations which relax the non-convex constraint $\mathbf{X} = \mathbf{x}\mathbf{x}^\top$ using a set of second order cone (SOC) constraints of the form

$$||(\mathbf{U}^k)^\top \mathbf{x}|| \leq \mathbf{C}^k \bullet \mathbf{X}, k = 1, \cdots, n_C \tag{6.4.1}$$

where $\mathbf{C}^k = \mathbf{U}^k(\mathbf{U}^k)^\top \succeq 0$, for all $k = 1, \cdots, n_C$. In order to make the proofs of the subsequent theorems easier, we make two assumptions. However, the theorems would hold true even without these assumptions as discussed below.

Assumption 1: We assume that the integer constraints

$$\mathbf{x} \in \{-1, +1\}^{nh}, \mathbf{X} \in \{-1, +1\}^{nh \times nh}, \tag{6.4.2}$$

are relaxed to

$$\mathbf{x} \in [-1, +1]^{nh}, \mathbf{X} \in [-1, +1]^{nh \times nh}, \tag{6.4.3}$$

with $X_{aa;ii} = 1$, for all $v_a \in \mathbf{v}, l_i \in \mathbf{l}$. The constraints (6.4.3) provide the convex hull for all the points defined by the integer constraints (6.4.2). Recall that the convex hull of a set of points is the smallest convex set which contains all the points. We now discuss how the above assumption is not restrictive with respect to the results that follow. Let A be a relaxation which contains constraints (6.4.3). Further, let B be a relaxation which differs from A only in the way it relaxes the integer constraints. Then by definition of convex hull $\mathcal{F}(\text{A}) \subset \mathcal{F}(\text{B})$. In other words A dominates B (see Case I in § 6.1.1). Hence, if A is dominated by the LP-S relaxation, then LP-S would also dominate B.

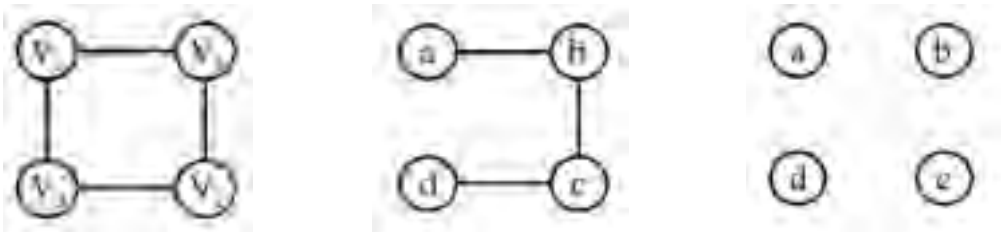
Assumption 2: We assume that the set of constraints (6.4.1) contains all the constraints specified in the SOCP-MS relaxation. Recall that for a given pair of neighbouring random variables, i.e. $(a, b) \in \mathcal{E}$, and a pair of labels $l_i, l_j \in \mathbf{l}$, SOCP-MS specifies SOC constraints using two matrices (say $\mathbf{C}^1$ and $\mathbf{C}^2$) which are 0 everywhere expect for the following 2×2 submatrices:

$$\begin{pmatrix} C^1_{aa;ii} & C^1_{ab;ij} \\ C^1_{ba;ji} & C^1_{bb;jj} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, \begin{pmatrix} C^2_{aa;ii} & C^2_{ab;ij} \\ C^2_{ba;ji} & C^2_{bb;jj} \end{pmatrix} = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}. \tag{6.4.4}$$

In the case where a given relaxation A does not contain the SOCP-MS constraints, we can define a new relaxation B. This new relaxation B is obtained by adding all the SOCP-MS constraints to A. By definition, B dominates A (although not necessarily strictly, see Case II in § 6.1.1). Hence, if B is dominated by the LP-S relaxation then it follows that LP-S would also dominate A. Hence, our assumption about including the SOCP-MS constraints is not restrictive for the results presented in this section.

Note that each SOCP relaxation belonging to this class would define an equivalent QP relaxation (similar to the equivalent QP-RL relaxation defined by the SOCP-MS relaxation). Hence, all these QP relaxations will also be dominated by the LP-S relaxation. Before we begin to describe our results in detail, we need to set up some notation as follows.

6.4.1 Notation



(a) (b) (c)

Figure 6.2: **(a)** *An example* CRF *defined over four variables which form a cycle. Note that the observed nodes are not shown for the sake of clarity of the image.* **(b)** *The set* E^k *specified by the matrix* $\mathbf{C}^k$ *shown in equation (6.4.6), i.e.* $E^k = \{(a,b),(b,c),(c,d)\}$. **(c)** *The set* $V^k = \{a,b,c,d\}$. *See text for definitions of these sets.*

We consider an SOC constraint which is of the form described in equation (6.4.1), i.e.

$$||(\mathbf{U}^k)^\top \mathbf{x}|| \leq \mathbf{C}^k \bullet \mathbf{X}, \tag{6.4.5}$$

where $k \in \{1, \cdots, n_C\}$. In order to help the reader understand the notation better, we use an example CRF shown in Fig. 6.2(a). This CRF is defined over four variables $\mathbf{v} = \{v_a, v_b, v_c, v_d\}$ (connected to form a cycle of length 4), each of which take a label from the set $\mathbf{l} = \{l_0, l_1\}$. For this CRF we specify a constraint using a matrix $\mathbf{C}^k \succeq 0$ which is 0 everywhere, except for the following 4×4 submatrix:

$$\begin{pmatrix} C^k_{aa;00} & C^k_{ab;00} & C^k_{ac;00} & C^k_{ad;00} \\ C^k_{ba;00} & C^k_{bb;00} & C^k_{bc;00} & C^k_{bd;00} \\ C^k_{ca;00} & C^k_{cb;00} & C^k_{cc;00} & C^k_{cd;00} \\ C^k_{da;00} & C^k_{db;00} & C^k_{dc;00} & C^k_{dd;00} \end{pmatrix} = \begin{pmatrix} 2 & 1 & 1 & 0 \\ 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 0 & 1 & 1 & 2 \end{pmatrix} \tag{6.4.6}$$

Using the SOC constraint shown in equation (6.4.5) we define the following sets:

- The set E^k is defined such that $(a, b) \in E^k$ if, and only if, it satisfies the following conditions:

$$(a, b) \in \mathcal{E}, \tag{6.4.7}$$

$$\exists l_i, l_j \in \mathbf{l} \text{ such that } C^k_{ab;ij} \neq 0. \tag{6.4.8}$$

  Recall that $\mathcal{E}$ specifies the neighbourhood relationship for the given CRF. In other words E^k is the subset of the edges in the graphical model of the CRF such that $\mathbf{C}^k$ specifies constraints for the random variables corresponding to those edges. For the example CRF (shown in Fig. 6.2(a)) and $\mathbf{C}^k$ matrix (in equation (6.4.6)), the set E^k obtained is shown in Fig. 6.2(b).

- The set V^k is defined as $a \in V^k$ if, and only if, there exists a $v_b \in \mathbf{v}$ such that $(a, b) \in E^k$. In other words V^k is the subset of hidden nodes in the graphical model of the CRF such that $\mathbf{C}^k$ specifies constraints for the random variables corresponding to those hidden nodes. Fig. 6.2(c) shows the set V^k for our example SOC constraint.

- The set $\mathcal{T}^k$ consists of elements $a; i \in \mathcal{T}^k$ which satisfy

$$a \in V^k, l_i \in \mathbf{l}, \tag{6.4.9}$$

$$\exists b \in V^k, l_j \in \mathbf{l}, \text{ such that } C^k_{ab;ij} \neq 0. \tag{6.4.10}$$

  In other words the set $\mathcal{T}^k$ consists of the set of indices for the vector $\mathbf{x}$ which are constrained by inequality (6.4.5), i.e. the coefficient of $x_{a;i}$ where $a; i \in \mathcal{T}^k$ are non-zero in the LHS of inequality (6.4.5). Note that in equation (6.4.6) the constraint is specified using only the label l_0 for all the random variables $\mathbf{v}$. Thus the set $\mathcal{T}^k$ is given by

$$\mathcal{T}^k = \{(a; 0), (b; 0), (c; 0), (d; 0)\}. \tag{6.4.11}$$

  For each set $\mathcal{T}^k$ we define three disjoint subsets of $\mathcal{T}^k \times \mathcal{T}^k$ as follows.

- The set $\mathcal{T}^k_0$ is defined as

$$\mathcal{T}^k_0 = \{(a; i, b; j) | (a; i, b; j) \in \mathcal{T}^k \times \mathcal{T}^k, (a, b) \in \mathcal{E}, (a, b) \notin E^k\}. \tag{6.4.12}$$

  Note that by definition $C^k_{ab;ij} = 0$ if $(a; i, b; j) \in \mathcal{T}^k_0$. Thus $\mathcal{T}^k_0$ indexes the elements of matrix $\mathbf{X}$ which are not constrained by inequality (6.4.5) but are present in the set $\mathcal{T}^k \times \mathcal{T}^k$. For the matrix $\mathbf{C}^k$ in equation (6.4.6), the set $\mathcal{T}^k_0$ is given by

$$\mathcal{T}^k_0 = \{(a; 0, d; 0)\} \tag{6.4.13}$$

- The set $\mathcal{T}_1^k$ is defined as

$$\mathcal{T}_1^k = \{(a;i,b;j)|(a;i,b;j) \in \mathcal{T}^k \times \mathcal{T}^k, (a,b) \notin \mathcal{E}\}. \tag{6.4.14}$$

  In other words the set $\mathcal{T}_1^k$ indexes the elements of matrix $\mathbf{X}$ which are constrained by inequality (6.4.5) but do not belong to any pair of neighbouring random variables. Note that the variables $X_{ab;ij}$ such that $(a;i,b;j) \in \mathcal{T}_1^k$ were not present in the LP-S relaxation. For the matrix $\mathbf{C}^k$ in equation (6.4.6), the set $\mathcal{T}_1^k$ is given by

$$\mathcal{T}_1^k = \{(a;0,c;0),(b;0,d;0)\} \tag{6.4.15}$$

- The set $\mathcal{T}_2^k$ is defined as

$$\mathcal{T}_2^k = \{(a;i,b;j)|(a;i,b;j) \in \mathcal{T}^k \times \mathcal{T}^k, (a,b) \in E^k\}. \tag{6.4.16}$$

  In other words the set $\mathcal{T}_2^k$ indexes the elements of matrix $\mathbf{X}$ which are constrained by inequality (6.4.5) and belong to a pair of neighbouring random variables. For the matrix $\mathbf{C}^k$ in equation (6.4.6), the set $\mathcal{T}_1^k$ is given by

$$\mathcal{T}_2^k = \{(a;0,b;0),(b;0,c;0)(c;0,d;0)\} \tag{6.4.17}$$

  Note that $\mathcal{T}_0^k \bigcup \mathcal{T}_1^k \bigcup \mathcal{T}_2^k = \mathcal{T}^k \times \mathcal{T}^k$. For a given set of pairwise potentials $\theta^2_{ab;ij}$ we define two disjoint sets of $\mathcal{T}_2^k$ as follows.

- The set $\mathcal{T}_{2+}^k$ corresponds to non-negative pairwise potentials, i.e.

$$\mathcal{T}_{2+}^k = \{(a;i,b;j)|(a;i,b;j) \in \mathcal{T}_2^k, \theta^2_{ab;ij} \geq 0\}, \tag{6.4.18}$$

  Thus the set $\mathcal{T}_{2+}^k$ indexes the elements of matrix $\mathbf{X}$ which belong to $\mathcal{T}_2^k$ and are multiplied by a non-negative pairwise potential in the objective function of the relaxation.

- The set $\mathcal{T}_{2-}^k$ corresponds to negative pairwise potentials, i.e.

$$\mathcal{T}_{2-}^k = \{(a;i,b;j)|(a;i,b;j) \in \mathcal{T}_2^k, \theta^2_{ab;ij} < 0\}, \tag{6.4.19}$$

  Thus the set $\mathcal{T}_{2-}^k$ indexes the elements of matrix $\mathbf{X}$ which belong to $\mathcal{T}_2^k$ and are multiplied by a negative pairwise potential in the objective function of the relaxation. Note that $\mathcal{T}_2^k = \mathcal{T}_{2+}^k \bigcup \mathcal{T}_{2-}^k$. For the purpose of illustration let us assume that, for the example CRF in Fig. 6.2(a), $\theta^2_{ab;00} \geq 0$ while $\theta^2_{bc;00} < 0$ and $\theta^2_{cd;00} < 0$. Then,

$$\mathcal{T}_{2+}^k = \{(a;0,b;0)\}, \tag{6.4.20}$$

$$\mathcal{T}_{2-}^k = \{(b;0,c;0),(c;0,d;0)\}, \tag{6.4.21}$$

We also define a weighted graph $G^k = (V^k, E^k)$ whose vertices are specified by the set V^k and whose edges are specified by the set E^k. The weight of an edge $(a, b) \in E^k$ is given by $w(a, b)$. Recall that $w(a, b)$ specifies the strength of the pairwise relationship between two neighbouring variables v_a and v_b. Thus, for our example SOC constraint, the vertices of this graph are given in Fig. 6.2(c) while the edges are shown in Fig. 6.2(b). This graph can be viewed as a subgraph of the graphical model representation for the given CRF.

Further, we define the submatrices $\mathbf{x}_T^k$ and $\mathbf{X}_T^k$ of $\mathbf{x}$ and $\mathbf{X}$ respectively such that

$$\mathbf{x}_T^k = \{x_{a;i} | a; i \in \mathcal{T}^k\}, \tag{6.4.22}$$

$$\mathbf{X}_T^k = \{X_{ab;ij} | (a; i, b; j) \in \mathcal{T}^k \times \mathcal{T}^k\}. \tag{6.4.23}$$

For our example, these submatrices will be given by

$$\mathbf{x}_T^k = \begin{pmatrix} x_{a;0} \\ x_{b;0} \\ x_{c;0} \\ x_{d;0} \end{pmatrix}, \mathbf{X}_T^k = \begin{pmatrix} X_{aa;00} & X_{ab;00} & X_{ac;00} & X_{ad;00} \\ X_{ba;00} & X_{bb;00} & X_{bc;00} & X_{bd;00} \\ X_{ca;00} & X_{cb;00} & X_{cc;00} & X_{cd;00} \\ X_{da;00} & X_{db;00} & X_{dc;00} & X_{dd;00} \end{pmatrix}. \tag{6.4.24}$$

Using the above notation, we are now ready to describe our results in detail.

6.4.2 QP and SOCP Relaxations over Trees

We begin by considering those relaxations where the SOC constraints are defined such that the graphs $G^k = (V^k, E^k)$ form trees. For example, the graph G^k defined by the SOC constraint in equation (6.4.6) forms a tree as shown in Fig. 6.2(b). We denote such a relaxation, which specifies SOC constraints only over trees, by SOCP-T. Note that SOCP-MS (and hence, QP-RL) can be considered a special case of this class of relaxations where the number of vertices in each tree is equal to two (since the constraints are defined for all $(a, b) \in \mathcal{E}$).

We will remove this restriction by allowing the number of vertices in each tree to be arbitrarily large (i.e. between 1 and n). We consider one such tree $G = (V, E)$. Note that for a given relaxation SOCP-T, there may be several SOC constraints defined using this tree G (or its subtree). Without loss of generality, we assume that the constraints

$$||(\mathbf{U}^k)^\top \mathbf{x}|| \leq \mathbf{C}^k \bullet \mathbf{X}, k = 1, \cdots, n'_C \tag{6.4.25}$$

are defined on the tree G. In other words,

$$G^k \subseteq G, k = 1, \cdots, n'_C, \tag{6.4.26}$$

where $G^k \subseteq G$ implies that G^k is a subtree of G. In order to make the notation less cluttered, **we will drop the superscript k from the sets defined in the previous section** (since we will consider only one tree G in our analysis).

We will now show that SOCP-T is dominated by the LP-S relaxation. This result is independent of the choice of trees G and matrices $\mathbf{C}^k$. To this end, we define the term $e_1(\mathbf{x}_T)$ for a given value of $\mathbf{x}_T$ as

$$e_1(\mathbf{x}_T) = \sum_{(a;i)\in\mathcal{T}} \left(\theta^1_{a;i} + \sum_{(b;j)\in\mathcal{T}} \frac{\theta^2_{ab;ij}}{2} \right) x_{a;i}. \tag{6.4.27}$$

Further, for a fixed $\mathbf{x}_T$ we also define the following two terms:

$$e^S_2(\mathbf{x}_T) = \min_{(\mathbf{x}_T,\mathbf{X}_T)\in\mathcal{F}(\text{SOCP-T})} \sum_{(a;i,b;j)\in\mathcal{T}_2} \theta^2_{ab;ij} X_{ab;ij}, \tag{6.4.28}$$

$$e^L_2(\mathbf{x}_T) = \min_{(\mathbf{x}_T,\mathbf{X}_T)\in\mathcal{F}(\text{LP-S})} \sum_{(a;i,b;j)\in\mathcal{T}_2} \theta^2_{ab;ij} X_{ab;ij}, \tag{6.4.29}$$

where $\mathcal{F}$(SOCP-T) and $\mathcal{F}$(LP-S) are the feasibility regions of SOCP-T and LP-S respectively. We use the notation $(\mathbf{x}_T, \mathbf{X}_T) \in \mathcal{F}$(SOCP-T) loosely to mean that we can obtain a feasible solution $(\mathbf{x}, \mathbf{X})$ of SOCP-T such that the values of the variables $x_{a;i}$ where $a;i \in \mathcal{T}$ and $X_{ab;ij}$ where $(a;i,b;j) \in \mathcal{T}\times\mathcal{T}$ are equal to the values specified by $\mathbf{x}_T$ and $\mathbf{X}_T$. The notation $(\mathbf{x}_T, \mathbf{X}_T) \in \mathcal{F}$(LP-S) is used similarly. Note that for a given $\mathbf{x}_T$ the possible values of $\mathbf{X}_T$ are constrained such that $(\mathbf{x}_T, \mathbf{X}_T) \in \mathcal{F}$(SOCP-T) and $(\mathbf{x}_T, \mathbf{X}_T) \in \mathcal{F}$(LP-S) (in the case of SOCP-T and LP-S respectively). Hence different values of $\mathbf{x}_T$ will provide different values of $e^S_2(\mathbf{x}_T)$ and $e^L_2(\mathbf{x}_T)$.

The contribution of the tree G to the objective function of SOCP-T and LP-S is given by

$$e^S = \min_{\mathbf{x}_T} \frac{e_1(\mathbf{x}_T)}{2} + \frac{e^S_2(\mathbf{x}_T)}{4}, \tag{6.4.30}$$

$$e^L = \min_{\mathbf{x}_T} \frac{e_1(\mathbf{x}_T)}{2} + \frac{e^L_2(\mathbf{x}_T)}{4} \tag{6.4.31}$$

respectively. Assuming that the trees G do not overlap, the total value of the objective function would simply be the sum of e^S (for SOCP-T) or e^L (for LP-S) over all trees G. However, since we use an arbitrary parameter $\boldsymbol{\theta}$ in our analysis, it follows that the results do not depend on this assumption of non-overlapping trees. In other words if two trees G^1 and G^2 share an edge $(a,b) \in \mathcal{E}$ then we can simply consider two MAP estimation problems defined using arbitrary parameters $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ such that $\boldsymbol{\theta}_1 + \boldsymbol{\theta}_2 = \boldsymbol{\theta}$. We can then add the contribution of G_1 for the MAP estimation problem with parameter $\boldsymbol{\theta}_1$ to the contribution of G_2 for the MAP estimation problem with parameter $\boldsymbol{\theta}_2$. This would then provide us with the total contribution of G_1 and G_2 for the original MAP estimation defined using parameter $\boldsymbol{\theta}$.

Using the above argument it follows that if, for all G and for all $\boldsymbol{\theta}$, the following holds true:

$$\tfrac{e_1(\mathbf{x}_T)}{2} + \tfrac{e^S_2(\mathbf{x}_T)}{4} \leq \tfrac{e_1(\mathbf{x}_T)}{2} + \tfrac{e^L_2(\mathbf{x}_T)}{4}, \forall \mathbf{x}_T \in [-1,1]^{|\mathcal{T}|} \tag{6.4.32}$$

$$\Rightarrow \quad e^S_2(\mathbf{x}_T) \leq e^L_2(\mathbf{x}_T), \forall \mathbf{x}_T \in [-1,1]^{|\mathcal{T}|}, \tag{6.4.33}$$

then LP-S dominates SOCP-T (since this would imply that $e^S \leq e^L$, for all G and for all $\boldsymbol{\theta}$). This is the condition that we will use to prove that LP-S dominates all SOCP relaxations whose constraints are defined over trees. To this end, we define a vector $\boldsymbol{\omega} = \{\omega_k, k = 1, \cdots, n'_C\}$ of non-negative real numbers such that

$$\sum_k \omega_k C^k_{ab;ij} = \theta^2_{ab;ij}, \forall (a;i,b;j) \in \mathcal{T}_2. \tag{6.4.34}$$

Due to the presence of the matrices $\mathbf{C}^k$ defined in equation (6.4.4) (which result in the SOCP-MS constraints for all $(a,b) \in \mathcal{E}$ and $l_i, l_j \in \mathbf{l}$), such a vector $\boldsymbol{\omega}$ would always exist for any CRF parameter $\boldsymbol{\theta}$. We denote the matrix $\sum_k \omega_k \mathbf{C}^k$ by $\mathbf{C}$. Clearly, $\mathbf{C} \succeq 0$, and hence can be written as $\mathbf{C} = \mathbf{U}\mathbf{U}^\top$.

Using the constraints $||(\mathbf{U}^k)^\top \mathbf{x}||^2 \leq \mathbf{C}^k \bullet \mathbf{X}_T$ together with the the fact that $\omega_k \geq 0$, we get the following inequality[1]:

$$\begin{aligned}
& \textstyle\sum_k \omega_k ||(\mathbf{U}^k)^\top \mathbf{x}||^2 \leq \sum_k \omega_k \mathbf{C}^k \bullet \mathbf{X}, \\
\Rightarrow \quad & ||\mathbf{U}^\top \mathbf{x}||^2 \leq \mathbf{C} \bullet \mathbf{X}, \\
\Rightarrow \quad & ||\mathbf{U}^\top \mathbf{x}||^2 \leq \textstyle\sum_{a;i \in \mathcal{T}} C_{aa;ii} X_{aa;ii} + \sum_{(a;i,b;j) \in \mathcal{T}_1} C_{ab;ij} X_{ab;ij} + \sum_{(a;i,b;j) \in \mathcal{T}_2} C_{ab;ij} X_{ab;ij} \\
\Rightarrow \quad & \|\mathbf{U}^\top \mathbf{x}\|^2 - \textstyle\sum_{a;i \in \mathcal{T}} C_{aa;ii} - \sum_{(a;i,b;j) \in \mathcal{T}_1} C_{ab;ij} X_{ab;ij} \leq \sum_{(a;i,b;j) \in \mathcal{T}_2} \theta^2_{ab;ij} X_{ab;ij},
\end{aligned} \tag{6.4.35}$$

where the last expression is obtained using the fact that $C_{ab;ij} = \theta^2_{ab;ij}$ for all $(a;i,b;j) \in \mathcal{T}_2$ and $X_{aa;ii} = 1$ for all $v_a \in \mathbf{v}$ and $l_i \in \mathbf{l}$. Note that, in the absence of any other constraint (which is our assumption), the value of $e^S_2(\mathbf{x}_T)$ after the minimization would be exactly equal to the LHS of the inequality given above (since the objective function containing $e^S_2(\mathbf{x}_T)$ is being minimized). In other words,

$$\begin{aligned}
e^S_2(\mathbf{x}_T) &= \min \sum_{(a;i,b;j) \in \mathcal{T}_2} \theta^2_{ab;ij} X_{ab;ij}, \\
&= \min \|\mathbf{U}^\top \mathbf{x}\|^2 - \sum_{a;i \in \mathcal{T}} C_{aa;ii} - \sum_{(a;i,b;j) \in \mathcal{T}_1} C_{ab;ij} X_{ab;ij}.
\end{aligned} \tag{6.4.36}$$

For the LP-S relaxation, from Lemmas 2.1 and 2.2, we obtain the following value of $e^L_2(\mathbf{x}_T)$:

$$|x_{a;i} + x_{b;j}| - 1 \leq X_{ab;ij} \leq 1 - |x_{a;i} - x_{b;j}|, \tag{6.4.37}$$

$$\begin{aligned}
\Rightarrow \quad & e^L_2(\mathbf{x}_T) = \min \textstyle\sum_{(a;i,b;j) \in \mathcal{T}_2} \theta^2_{ab;ij} X_{ab;ij}, \\
= \; & \textstyle\sum_{(a;i,b;j) \in \mathcal{T}_{2+}} \theta^2_{ab;ij} (|x_{a;i} + x_{b;j}|) - \sum_{(a;i,b;j) \in \mathcal{T}_{2-}} \theta^2_{ab;ij} (|x_{a;i} - x_{b;j}|) - \\
& \textstyle\sum_{(a;i,b;j) \in \mathcal{T}_2} |\theta^2_{ab;ij}|
\end{aligned} \tag{6.4.38}$$

We are now ready to prove the following theorem.

[1]Note that there are no terms corresponding to $(a;i,b;j) \in \mathcal{T}_0$ in inequality (6.4.35) since $C_{ab;ij} = 0$ if $(a;i,b;j) \in \mathcal{T}_0$. In other words, $X_{ab;ij}$ vanishes from the above inequality if $(a;i,b;j) \in \mathcal{T}_0$.

Theorem 4: SOCP relaxations (and the equivalent QP relaxations) which define constraints only using graphs $G = (V, E)$ which form (arbitrarily large) trees are dominated by the LP-S relaxation.

Proof: We begin by assuming that $d(i, j) \geq 0$ for all $l_i, l_j \in \mathbf{l}$ and later drop this constraint on the distance function[2]. We will show that for any arbitrary tree G and any matrix $\mathbf{C}$, the value of $e_2^L(\mathbf{x}_T)$ is greater than the value of $e_2^S(\mathbf{x}_T)$ for all $\mathbf{x}_T$. This would prove inequality (6.4.33) which in turn would show that the LP-S relaxation dominates SOCP-T (and the equivalent QP relaxation which we call QP-T) whose constraints are defined over trees.

It is assumed that we do not specify any additional constraints for all the variables $X_{ab;ij}$ where $(a; i, b; j) \in \mathcal{T}_1$ (i.e. for $X_{ab;ij}$ not belonging to any of our trees). In other words these variables $X_{ab;ij}$ are bounded only by the relaxation of the integer constraint, i.e. $-1 \leq X_{ab;ij} \leq +1$. Thus in equation (6.4.36) the minimum value of the RHS (which is equal to the value of $e_2^S(\mathbf{x}_T)$) is obtained by using the following value of $X_{ab;ij}$ where $(a; i, b; j) \in \mathcal{T}_1$:

$$X_{ab;ij} = \begin{cases} 1 & \text{if} \quad C_{ab;ij} \geq 0, \\ -1 & \text{otherwise.} \end{cases} \tag{6.4.39}$$

Substituting these values in equation (6.4.36) we get

$$\begin{aligned} & e_2^S(\mathbf{x}_T) = ||\mathbf{U}^\top \mathbf{x}||^2 - \textstyle\sum_{a;i \in \mathcal{T}} C_{aa;ii} - \sum_{(a;i,b;j) \in \mathcal{T}_1} |C_{ab;ij}|, \\ \Rightarrow \quad & e_2^S(\mathbf{x}_T) = \textstyle\sum_{a;i \in \mathcal{T}} C_{aa;ii} x_{a;i}^2 + \sum_{(a;i,b;j) \in \mathcal{T}_1} C_{ab;ij} x_{a;i} x_{b;j} + \sum_{(a;i,b;j) \in \mathcal{T}_2} \theta_{ab;ij}^2 x_{a;i} x_{b;j} \\ & \qquad - \textstyle\sum_{a;i \in \mathcal{T}} C_{aa;ii} - \sum_{(a;i,b;j) \in \mathcal{T}_1} |C_{ab;ij}|, \end{aligned} \tag{6.4.40}$$

where the last expression is obtained using the fact that $\mathbf{C} = \mathbf{U}^\top \mathbf{U}$. Consider the term $\sum_{(a;i,b;j) \in \mathcal{T}_1} C_{ab;ij} x_{a;i} x_{b;j}$ which appears in the RHS of the above equation. For this term, clearly the following holds true

$$\sum_{(a;i,b;j) \in \mathcal{T}_1} C_{ab;ij} x_{a;i} x_{b;j} \leq \sum_{(a;i,b;j) \in \mathcal{T}_1} \frac{|C_{ab;ij}|}{2} (x_{a;i}^2 + x_{b;j}^2), \tag{6.4.41}$$

since for all $(a; i, b; j) \in \mathcal{T}_1$

$$C_{ab;ij} \leq |C_{ab;ij}|, \tag{6.4.42}$$

$$x_{a;i} x_{b;j} \leq \tfrac{(x_{a;i}^2 + x_{b;j}^2)}{2}. \tag{6.4.43}$$

Inequality (6.4.41) provides us with an upper bound on the value of $e_2^S(\mathbf{x}_T)$ as follows:

$$\begin{aligned} e_2^S(\mathbf{x}_T) \leq & \textstyle\sum_{a;i \in \mathcal{T}} C_{aa;ii} x_{a;i}^2 + \sum_{(a;i,b;j) \in \mathcal{T}_1} \frac{|C_{ab;ij}|}{2} (x_{a;i}^2 + x_{b;j}^2) + \sum_{(a;i,b;j) \in \mathcal{T}_2} \theta_{ab;ij}^2 x_{a;i} x_{b;j} \\ & - \textstyle\sum_{a;i \in \mathcal{T}} C_{aa;ii} - \sum_{(a;i,b;j) \in \mathcal{T}_1} |C_{ab;ij}|. \end{aligned} \tag{6.4.44}$$

[2]Recall that $d(\cdot, \cdot)$ is a distance function on the labels. Together with the weights $w(\cdot, \cdot)$ defined over neighbouring random variables, it specifies the pairwise potentials as $\theta_{ab;ij}^2 = w(a, b) d(i, j)$.

Note that in order to prove inequality (6.4.33), i.e.

$$e_2^S(\mathbf{x}_T) \leq e_2^L(\mathbf{x}_T), \forall \mathbf{x}_T \in [-1,1]^{|\mathcal{T}|}, \tag{6.4.45}$$

it would be sufficient to show that $e_2^L(\mathbf{x}_T)$ specified in equation (6.4.38) is greater than the RHS of inequality (6.4.44) (since the RHS of inequality (6.4.44) is greater than $e_2^S(\mathbf{x}_T)$). We now simplify the two infimums $e_2^L(\mathbf{x}_T)$ and $e_2^S(\mathbf{x}_T)$ as follows.

LP-S Infimum: Let $z_{a;i} = \sqrt{|x_{a;i}|(1-|x_{a;i}|)}$. From equation (6.4.38), we see that the infimum provided by the LP-S relaxation is given by

$$\begin{aligned}
&\textstyle\sum_{(a;i,b;j)\in\mathcal{T}_{2+}} \theta^2_{ab;ij}(|x_{a;i}+x_{b;j}|) - \sum_{(a;i,b;j)\in\mathcal{T}_{2-}} \theta^2_{ab;ij}(|x_{a;i}-x_{b;j}|) - \sum_{(a;i,b;j)\in\mathcal{T}_2} |\theta^2_{ab;ij}| \\
&= -\textstyle\sum_{(a;i,b;j)\in\mathcal{T}_{2+}} |\theta^2_{ab;ij}|(1-|x_{a;i}+x_{b;j}|+x_{a;i}x_{b;j}) \\
&\quad -\textstyle\sum_{(a;i,b;j)\in\mathcal{T}_{2-}} |\theta^2_{ab;ij}|(1-|x_{a;i}-x_{b;j}|-x_{a;i}x_{b;j}) \\
&\quad +\textstyle\sum_{(a;i,b;j)\in\mathcal{T}_2} \theta^2_{ab;ij}x_{a;i}x_{b;j} \\
&\geq -\textstyle\sum_{(a;i,b;j)\in\mathcal{T}_2} |\theta^2_{ab;ij}|(1-|x_{a;i}|)(1-|x_{b;j}|) - 2\sum_{(a;i,b;j)\in\mathcal{T}_2} |\theta^2_{ab;ij}|z_{a;i}z_{b;j} + \\
&\quad +\textstyle\sum_{(a;i,b;j)\in\mathcal{T}_2} \theta^2_{ab;ij}x_{a;i}x_{b;j}.
\end{aligned} \tag{6.4.46}$$

The last expression is obtained using the fact that

$$(1-|x_{a;i}+x_{b;j}|+x_{a;i}x_{b;j}) \leq (1-|x_{a;i}|)(1-|x_{b;j}|) + 2z_{a;i}z_{b;j}, \tag{6.4.47}$$

$$(1-|x_{a;i}-x_{b;j}|-x_{a;i}x_{b;j}) \leq (1-|x_{a;i}|)(1-|x_{b;j}|) + 2z_{a;i}z_{b;j}. \tag{6.4.48}$$

SOCP Infimum: From inequality (6.4.44), we see that the infimum provided by the SOCP-T relaxation is given by

$$\begin{aligned}
&\textstyle\sum_{a;i\in\mathcal{T}} C_{aa;ii}x^2_{a;i} + \sum_{(a;i,b;j)\in\mathcal{T}_1} \frac{|C_{ab;ij}|}{2}(x^2_{a;i}+x^2_{b;j}) + \sum_{(a;i,b;j)\in\mathcal{T}_2} \theta^2_{ab;ij}x_{a;i}x_{b;j} \\
&\quad -\textstyle\sum_{a;i\in\mathcal{T}} C_{aa;ii} - \sum_{(a;i,b;j)\in\mathcal{T}_1} |C_{ab;ij}| \\
=\ &-\textstyle\sum_{a;i\in\mathcal{T}} C_{aa;ii}(1-x^2_{a;i}) - \sum_{(a;i,b;j)\in\mathcal{T}_1} |C_{ab;ij}|(1-\frac{x^2_{a;i}}{2}-\frac{x^2_{b;j}}{2}) \\
&\quad +\textstyle\sum_{(a;i,b;j)\in\mathcal{T}_2} \theta^2_{ab;ij}x_{a;i}x_{b;j} \\
\leq\ &-\textstyle\sum_{a;i\in\mathcal{T}} C_{aa;ii}(1-|x_{a;i}|)^2 - \sum_{(a;i,b;j)\in\mathcal{T}_1} |C_{ab;ij}|(1-|x_{a;i}|)(1-|x_{b;j}|) \\
&\quad -2\textstyle\sum_{a;i\in\mathcal{T}} C_{aa;ii}z^2_{a;i} - 2\sum_{(a;i,b;j)\in\mathcal{T}_1} |C_{ab;ij}|z_{a;i}z_{b;j} \\
&\quad +\textstyle\sum_{(a;i,b;j)\in\mathcal{T}_2} \theta^2_{ab;ij}x_{a;i}x_{b;j}.
\end{aligned} \tag{6.4.49}$$

The last expression is obtained using

$$1-x^2_{a;i} \geq (1-|x_{a;i}|)^2 + 2z^2_{a;i}, \tag{6.4.50}$$

$$1-\tfrac{x^2_{a;i}}{2}-\tfrac{x^2_{b;j}}{2} \geq (1-|x_{a;i}|)(1-|x_{b;j}|) + 2z_{a;i}z_{b;j}. \tag{6.4.51}$$

In order to prove the theorem, we use the following two lemmas.

Lemma 4.1: The following inequality holds true for any matrix $\mathbf{C} \succeq 0$:

$$\begin{aligned}
&\textstyle\sum_{a;i\in\mathcal{T}} C_{aa;ii}(1-|x_{a;i}|)^2 + \sum_{(a;i,b;j)\in\mathcal{T}_1} |C_{ab;ij}|(1-|x_{a;i}|)(1-|x_{b;j}|) \\
&\quad \geq \textstyle\sum_{(a;i,b;j)\in\mathcal{T}_2} |\theta^2_{ab;ij}|(1-|x_{a;i}|)(1-|x_{b;j}|).
\end{aligned} \tag{6.4.52}$$

(a) (b)

Figure 6.3: **(a)** *An example subgraph G which forms a tree. The weights of the edges and corresponding elements of the vector* $\mathbf{m}$ *are also shown.* **(b)** *An example subgraph G which forms an even cycle where all weights are positive. The elements of* $\mathbf{s}$ *are defined using the* $\{+1, -1\}$ *assignments of the vertices.*

In other words, the first term in the RHS of inequality (6.4.46) exceeds the sum of the first two terms in the RHS of inequality (6.4.49).

Proof: The proof relies on the fact that $\mathbf{C}$ is positive semidefinite. We construct a vector $\mathbf{m} = \{m_a, a = 1, \cdots, n\}$ where n is the number of variables. Let $p(a)$ denote the parent of a non-root vertex a of tree G (where the root vertex can be chosen arbitrarily). The vector $\mathbf{m}$ is defined such that

$$m_a = \begin{cases} 0 & \text{if } a \text{ does not belong to tree } G, \\ 1 & \text{if } a \text{ is the root vertex of } G, \\ -m_{p(a)} & \text{if } w(a, p(a)) > 0, \\ m_{p(a)} & \text{if } w(a, p(a)) < 0. \end{cases}$$

Here $w(\cdot, \cdot)$ are the weights provided for a given CRF. Fig. 6.3(a) shows an example of a graph which forms a tree together with the corresponding elements of $\mathbf{m}$. Using the vector $\mathbf{m}$, we define a vector $\mathbf{s}$ of length nh (where $h = |\mathbf{l}|$) such that $s_{a;i} = 0$ if $a; i \notin \mathcal{T}$ and $s_{a;i} = m_a(1 - |x_{a;i}|)$ otherwise. Since $\mathbf{C}$ is positive semidefinite, we get

$$\mathbf{s}^\top \mathbf{C}\mathbf{s} \geq 0 \tag{6.4.53}$$

$$\Rightarrow \sum_{a;i \in \mathcal{T}} C_{aa;ii}(1 - |x_{a;i}|)^2 + \sum_{(a;i,b;j) \in \mathcal{T}_1} m_a m_b C_{ab;ij}(1 - |x_{a;i}|)(1 - |x_{b;j}) + \sum_{(a;i,b;j) \in \mathcal{T}_2} m_a m_b \theta^2_{ab;ij}(1 - |x_{a;i}|)(1 - |x_{b;j}) \geq 0, \tag{6.4.54}$$

$$\Rightarrow \sum_{a;i \in \mathcal{T}} C_{aa;ii}(1 - |x_{a;i}|)^2 + \sum_{(a;i,b;j) \in \mathcal{T}_1} m_a m_b C_{ab;ij}(1 - |x_{a;i}|)(1 - |x_{b;j}) \geq \sum_{(a;i,b;j) \in \mathcal{T}_2} |\theta^2_{ab;ij}|(1 - |x_{a;i}|)(1 - |x_{b;j}), \tag{6.4.55}$$

$$\Rightarrow \sum_{a;i \in \mathcal{T}} C_{aa;ii}(1 - |x_{a;i}|)^2 + \sum_{(a;i,b;j) \in \mathcal{T}_1} |C_{ab;ij}|(1 - |x_{a;i}|)(1 - |x_{b;j}|) \geq \sum_{(a;i,b;j) \in \mathcal{T}_2} |\theta^2_{ab;ij}|(1 - |x_{a;i}|)(1 - |x_{b;j}|). \quad \blacksquare \tag{6.4.56}$$

Lemma 4.2: The following inequality holds true for any matrix $\mathbf{C} \succeq 0$:

$$\sum_{a;i \in \mathcal{T}} C_{aa;ii} z^2_{a;i} + \sum_{(a;i,b;j) \in \mathcal{T}_1} |C_{ab;ij}| z_{a;i} z_{b;j} \geq \sum_{(a;i,b;j) \in \mathcal{T}_2} |\theta^2_{ab;ij}| z_{a;i} z_{b;j}. \tag{6.4.57}$$

In other words the second term in the RHS of inequality (6.4.46) exceeds the sum of the third and fourth terms in inequality (6.4.49).

Proof: Similar to Lemma 4.1, we construct a vector $\mathbf{s}$ of length nh such that $s_{a;i} = 0$ if $a; i \notin \mathcal{T}$ and $s_{a;i} = m_a z_{a;i}$ otherwise. The proof follows by observing that $\mathbf{s}^\top \mathbf{C}\mathbf{s} \geq 0$. ∎

Using the above two lemmas, we see that the sum of the first two terms of inequality (6.4.46) exceed the sum of the first four terms of inequality (6.4.49). Further, the third and the fifth terms of inequalities (6.4.46) and (6.4.49) are the same. Since inequality (6.4.46) provides the lower limit of $e_2^L(\mathbf{x}_T)$ and inequality (6.4.49) provides the upper limit of $e_2^S(\mathbf{x}_T)$, it follows that $e_2^L(\mathbf{x}_T) \geq e_2^S(\mathbf{x}_T)$ for all $\mathbf{x}_T \in [-1, 1]^{|\mathcal{T}|}$. Using condition (6.4.33), this proves the theorem. ∎

The proofs of Lemmas 4.1 and 4.2 make use of the fact that for any neighbouring random variables v_a and v_b (i.e. $(a, b) \in \mathcal{E}$), the pairwise potentials $\theta^2_{ab;ij}$ have the same sign for all $l_i, l_j \in \mathbf{l}$. This follows from the non-negativity property of the distance function. However, Theorem 4 can be extended to the case where the distance function does not obey the non-negativity property. To this end, we define a parameter $\overline{\boldsymbol{\theta}}$ which is the reparameterization of $\boldsymbol{\theta}$ (i.e. $\overline{\boldsymbol{\theta}} \equiv \boldsymbol{\theta}$). Note that there exist several reparameterizations of any parameter $\boldsymbol{\theta}$. We are interested in a parameter $\overline{\boldsymbol{\theta}}$ which satisfies

$$\sum_{l_i, l_j \in \mathbf{l}} |\overline{\theta}^2_{ab;ij}| = |\sum_{l_i, l_j \in \mathbf{l}} \theta^2_{ab;ij}|, \forall (a, b) \in \mathcal{E}. \tag{6.4.58}$$

It can easily be shown that such a reparameterization always exists. Specifically, consider the general form of reparameterization discussed in § 2.4.2, i.e.

$$\overline{\theta}^1_{a;i} = \theta^1_{a;i} + M_{ba;i}, \tag{6.4.59}$$

$$\overline{\theta}^2_{ab;ij} = \theta^2_{ab;ij} - M_{ba;i} - M_{ab;j}. \tag{6.4.60}$$

Clearly one can set the values of the terms $M_{ba;i}$ and $M_{ab;j}$ such that equation (6.4.58) is satisfied. Further, the optimal value of LP-S for the parameter $\overline{\boldsymbol{\theta}}$ is equal to its optimal value obtained using $\boldsymbol{\theta}$. For details, we refer the reader to [46]. Using this parameter $\overline{\boldsymbol{\theta}}$, we obtain an LP-S infimum which is similar in form to the inequality (6.4.46) for any distance function (i.e. without the positivity constraint $d(i, j) \geq 0$ for all $l_i, l_j \in \mathbf{l}$). This LP-S infimum can then be easily compared to the SOCP-T infimum of inequality (6.4.49) (using slight extensions of Lemmas 4.1 and 4.2), thereby proving the results of Theorem 4 for a general distance function. We omit details.

As an upshot of the above theorem, we see that the feasibility region of LP-S is always a subset of the feasibility region of SOCP-T (for any general set of trees and SOC constraints), i.e. $\mathcal{F}$(LP-S) $\subset \mathcal{F}$(SOCP-T). This implies that $\mathcal{F}$(LP-S) $\subset \mathcal{F}$(QP-T), where QP-T is the equivalent QP relaxation defined by SOCP-T.

We note that the above theorem can also be proved using the results of [102] on *moment constraints* (which imply that LP-S provides the exact solution for the

MAP estimation problems defined over tree-structured random fields). However, the proof presented here allows us to generalize the results of Theorem 4 for certain cycles as follows.

6.4.3 QP and SOCP Relaxations over Cycles

We now prove that the above result also holds true when the graph G forms an *even cycle*, i.e. cycles with even number of vertices, whose weights are all non-negative or all non-positive provided $d(i,j) \geq 0$, for all $l_i, l_j \in \mathbf{l}$.

Theorem 5: When $d(i,j) \geq 0$ for all $l_i, l_j \in \mathbf{l}$, the SOCP relaxations which define constraints only using non-overlapping graphs G which form (arbitrarily large) even cycles with all positive or all negative weights are dominated by the LP-S relaxation.

Proof: It is sufficient to show that Lemmas 4.1 and 4.2 hold for a graph $G = (V, E)$ which forms an even cycle. We first consider the case where $\theta^2_{ab;ij} > 0$. Without loss of generality, we assume that $V = \{1, 2, \ldots, t\}$ (where t is even) such that $(i, i+1) \in E$ for all $i = 1, \cdots, t-1$. Further, $(t, 1) \in E$ thereby forming an even cycle. We construct a vector $\mathbf{m}$ of size n such that $m_a = -1^a$ if $a \in V$ and $m_a = 0$ otherwise. When $\theta^2_{ab;ij} < 0$, we define a vector $\mathbf{m}$ such that $m_a = 1$ if $a \in V$ and $m_a = 0$ otherwise. Fig. 6.3(b) shows an example of a graph G which forms an even cycle together with the corresponding elements of $\mathbf{m}$. Using $\mathbf{m}$, we construct a vector $\mathbf{s}$ of length nh (similar to the proofs of Lemmas 4.1 and 4.2). Lemmas 4.1 and 4.2 follow from the fact that $\mathbf{s}^\top \mathbf{C}\mathbf{s} \geq 0$. We omit details. ∎

The above theorem can be proved for cycles of any length whose weights are all negative by a similar construction. Further, it also holds true for *odd cycles* (i.e. cycles of odd number of variables) which have only one positive or only one negative weight. However, as will be seen in the next section, unlike trees it is not possible to extend these results for any general cycle.

6.5. Some Useful SOC Constraints

We now describe two SOCP relaxations which include all the marginalization constraints specified in LP-S. Note that the marginalization constraints can be incorporated within the SOCP framework but not in the QP framework.

6.5.1 The SOCP-C Relaxation

The SOCP-C relaxation (where C denotes cycles) defines second order cone (SOC) constraints using positive semidefinite matrices $\mathbf{C}$ such that the graph G (defined in § 6.4.1) form cycles. Let the variables corresponding to vertices of one such cycle G of length c be denoted as $\mathbf{v}_C = \{v_b | b \in \{a_1, a_2, \cdots, a_c\}\}$. Further, let

$\mathbf{l}_C = \{l_j | j \in \{i_1, i_2, \cdots, i_c\}\} \in \mathbf{l}^c$ be a set of labels for the variables $\mathbf{v}_C$. The SOCP-C relaxation specifies the following constraints:

- The marginalization constraints, i.e.

$$\sum_{l_j \in \mathbf{l}} X_{ab;ij} = (2-h) x_{a;i}, \forall (a,b) \in \mathcal{E}, l_i \in \mathbf{l}. \tag{6.5.1}$$

- A set of SOC constraints

$$||\mathbf{U}^\top \mathbf{x}|| \leq \mathbf{C} \bullet \mathbf{X}, \tag{6.5.2}$$

  such that the graph G defined by the above constraint forms a cycle. The matrix $\mathbf{C}$ is 0 everywhere except the following elements:

$$C_{a_k,a_l,i_k,i_l} = \begin{cases} \lambda_c & \text{if} \quad k = l, \\ D_c(k,l) & \text{otherwise.} \end{cases}$$

  Here $\mathbf{D}_c$ is a $c \times c$ matrix which is defined as follows:

$$D_c(k,l) = \begin{cases} 1 & \text{if} \quad |k-l| = 1 \\ (-1)^{c-1} & \text{if} \quad |k-l| = c-1 \\ 0 & \text{otherwise,} \end{cases}$$

  and λ_c is the absolute value of the smallest eigenvalue of $\mathbf{D}_c$.

In other words the submatrix of $\mathbf{C}$ defined by $\mathbf{v}_C$ and $\mathbf{l}_C$ has diagonal elements equal to λ_c and off-diagonal elements equal to the elements of $\mathbf{D}_c$. As an example we consider two cases when $c = 3$ and $c = 4$. In these cases the matrix $\mathbf{D}_c$ is given by

$$\mathbf{D}_3 = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \text{ and } \mathbf{D}_4 = \begin{pmatrix} 0 & 1 & 0 & -1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ -1 & 0 & 0 & 1 \end{pmatrix}, \tag{6.5.3}$$

respectively, while $\lambda_3 = 1$ and $\lambda_4 = \sqrt{2}$. Clearly, $\mathbf{C} = \mathbf{U}^\top \mathbf{U} \succeq 0$ since its only non-zero submatrix $\lambda_c I + \mathbf{D}_c$ (where I is a $c \times c$ identity matrix) is positive semidefinite. This allows us to define a valid SOC constraint as shown in inequality (6.5.2). We choose to define the SOC constraint (6.5.2) for only those set of labels $\mathbf{l}_C$ which satisfy the following:

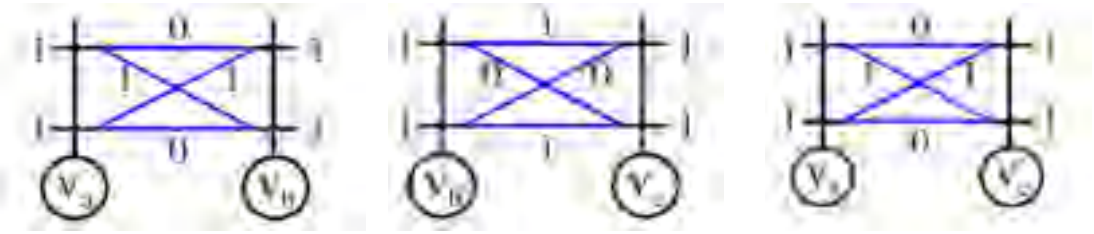
$$\sum_{(a_k,a_l) \in \mathcal{E}} D_c(k,l) \theta^2_{a_k a_l; i_k i_l} \geq \sum_{(a_k,a_l) \in \mathcal{E}} D_c(k,l) \theta^2_{a_k a_l; j_k j_l}, \forall \{j_1, j_2, \cdots, j_c\}. \tag{6.5.4}$$

Note that this choice is motivated by the fact that the variables $X_{a_k a_l; i_k i_l}$ corresponding to these sets $\mathbf{v}_C$ and $\mathbf{l}_C$ are assigned trivial values by the LP-S relaxation in the presence of non-submodular terms (see example below), i.e.

$$X_{a_k a_l; i_k i_l} = \begin{cases} -1 & \text{if} \quad \theta^2_{a_k a_l; i_k i_l} \geq 0, \\ 1 & \text{otherwise.} \end{cases} \tag{6.5.5}$$

In order to avoid this trivial solution, we impose the SOC constraint (6.5.2) on them.

Since marginalization constraints are included in the SOCP-C relaxation, the value of the objective function obtained by solving this relaxation would at least be equal to the value obtained by the LP-S relaxation (i.e. SOCP-C dominates LP-S, see Case II in § 6.1.1). We can further show that in the case where $|\mathbf{l}| = 2$ and the constraint (6.5.2) is defined over a frustrated cycle[1] SOCP-C strictly dominates LP-S. One such example is given below. Note that if the given CRF contains no frustrated cycle, then it can be solved exactly using the method described in [35].



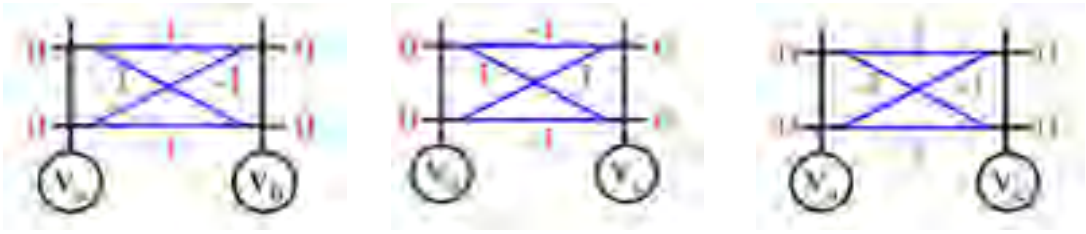
(a) (b) (c)

Figure 6.4: *An example* CRF *defined over three random variables* $\mathbf{v} = \{v_a, v_b, v_c\}$ *shown as unfilled circles. Each of these variables can take one of two labels from the set* $\mathbf{l} = \{l_0, l_1\}$ *which are shown as branches (i.e. the horizontal lines) of trellises (i.e. the vertical lines) on top of the random variables. The unary potentials are shown next to the corresponding branches. The pairwise potentials are shown next to the edges connecting the branches of two neighbouring variables. Note that the pairwise potentials defined for* (a, b) *and* (a, c) *form a submodular Ising model (in* **(a)** *and* **(b)** *respectively). The pairwise potentials defined for* (b, c) *are non-submodular (in* **(c)***).*

Example: We consider a frustrated cycle and show that SOCP-C strictly dominates LP-S. Specifically, we consider a CRF with $\mathbf{v} = \{v_a, v_b, v_c\}$ and $\mathbf{l} = \{l_0, l_1\}$. The neighbourhood of this CRF is defined such that the variables form a cycle of length 3, i.e. $\mathcal{E} = \{(a, b), (b, c), (c, a)\}$. We define a frustrated cycle which consists of all 3 variables of this CRF using the unary and pairwise potentials shown in Fig. 6.4, i.e. the unary potentials are uniform and the pairwise potentials define only one non-submodular term (between the vertices b and c). Clearly, the energy of the optimal labelling for the above problem is 4. The value of the objective function obtained by solving the LP-S relaxation is 3 at an optimal solution shown in Fig. 6.5.

The LP-S optimal solution is no longer feasible when the SOCP-C relaxation is

[1] A cycle is called frustrated if it contains an odd number of non-submodular terms.



(a) (b) (c)

Figure 6.5: *An optimal solution provided by the* LP-S *relaxation for the* CRF *shown in Fig. 6.4. This solution is shown in red to avoid confusing it with the potentials shown in Fig. 6.4. The value of variable $x_{a;i}$ is shown next to the i^{th} branch of the trellis on top of v_a. In this optimal solution, all such variables $x_{a;i}$ are equal to 0. The value of the variable $X_{ab;ij}$ is shown next to the connection joining the i^{th} and the j^{th} branch of the trellises on top of v_a and v_b respectively. Note that $X_{ab;ij} = -1$ when $\theta^2_{ab;ij} > 0$ and $X_{ab;ij} = 1$ otherwise. This provides us with the minimum value of the objective function of* LP-S*, i.e. 3.*

used. Specifically, the constraint

$$(x_{a;0} + x_{b;1} + x_{c;1})^2 \leq 3 + 2(X_{ab;01} + X_{ac;01} + X_{bc;11}) \tag{6.5.6}$$

is violated. In fact, the value of the objective function obtained using the SOCP-C relaxation is 3.75. Fig. 6.6 shows an optimal solution of the SOCP-C relaxation for the CRF in Fig. 6.4. The above example can be generalized to a frustrated cycle of any length. This proves that SOCP-C strictly dominates the LP-S relaxation (and hence, the QP-RL and SOCP-MS relaxations).

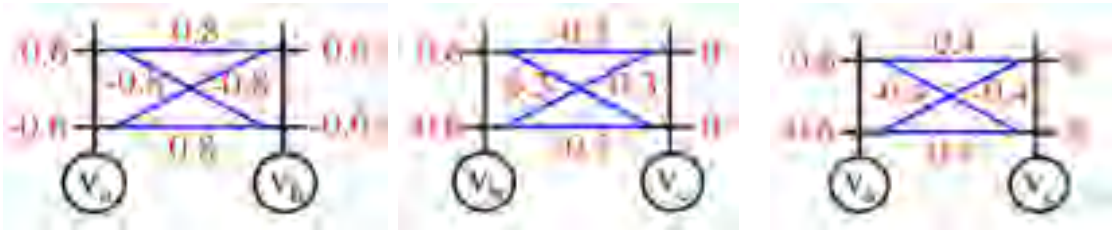
The constraint defined in equation (6.5.2) is similar to the (linear) cycle inequality constraints [4] which are given by

$$\sum_{k,l} D_c(k,l) X_{a_k a_l; i_k i_l} \geq 2 - c. \tag{6.5.7}$$

We believe that the feasibility region defined by cycle inequalities is a strict subset of the feasibility region defined by equation (6.5.2). In other words a relaxation defined by adding cycle inequalities to LP-S would strictly dominate SOCP-C. We are not aware of a formal proof for this. However, it will be seen in the next chapter that the empirical analysis agrees with this conjecture. We now describe the SOCP-Q relaxation.

6.5.2 The SOCP-Q Relaxation

In this previous section we saw that LP-S dominates SOCP relaxations whose constraints are defined on trees. However, the SOCP-C relaxation, which defines its constraints using cycles, strictly dominates LP-S. This raises the question



(a) (b) (c)

Figure 6.6: *An optimal solution provided by the* SOCP-C *relaxation for the* CRF *shown in Fig. 6.4. This optimal solution provides us with the optimal value of* 3.75 *which greater than the* LP-S *optimal value for the solution shown in Fig. 6.5. Note that the optimal solution of* LP-S *does not belong to the feasibility region of* SOCP-C *as it violates constraint (6.5.6). This example proves that* SOCP-C *strictly dominates* LP-S.

whether matrices $\mathbf{C}$, which result in more complicated graphs G, would provide an even better relaxation for the MAP estimation problem. In this section, we answer this question in an affirmative. To this end, we define an SOCP relaxation which specifies constraints such that the resulting graph G from a clique. We denote this relaxation by SOCP-Q (where Q indicates cliques).

The SOCP-Q relaxation contains the marginalization constraint and the cycle inequalities (defined above). In addition, it also defines SOC constraints on graphs G which form a clique. We denote the variables corresponding to the vertices of clique G as $\mathbf{v}_Q = \{v_b | b \in \{a_1, a_2, \cdots, a_q\}\}$. Let $\mathbf{l}_Q = \{l_j | j \in \{i_1, i_2, \cdots, i_q\}\}$ be a set of labels for these variables $\mathbf{v}_Q$. Given this set of variables $\mathbf{v}_Q$ and labels $\mathbf{l}_Q$, we define an SOC constraint using a matrix $\mathbf{C}$ of size $nh \times nh$ which is zero everywhere except for the elements $C_{a_k a_l; i_k i_l} = 1$. Clearly, $\mathbf{C}$ is a rank 1 matrix with eigenvalue 1 and eigenvector $\mathbf{u}$ which is zero everywhere except $u_{a_k; i_k} = 1$ where $v_{a_k} \in \mathbf{v}_Q$ and $l_{i_k} \in \mathbf{l}_Q$. This implies that $\mathbf{C} \succeq 0$, which enables us to obtain the following SOC constraint:

$$\left(\sum_k x_{a_k; i_k} \right)^2 \leq q + \sum_{k,l} X_{a_k a_l; i_k i_l}. \tag{6.5.8}$$

We choose to specify the above constraint only for the set of labels $\mathbf{l}_Q$ which satisfy the following condition:

$$\sum_{(a_k, a_l) \in \mathcal{E}} \theta^2_{a_k a_l; i_k i_l} \geq \sum_{(a_k, a_l) \in \mathcal{E}} \theta^2_{a_k a_l; j_k j_l}, \forall \{j_1, j_2, \cdots, j_q\}. \tag{6.5.9}$$

Again, this choice is motivated by the fact that the variables $X_{a_k a_l; i_k i_l}$ corresponding to these sets $\mathbf{v}_Q$ and $\mathbf{l}_Q$ are assigned trivial values by the LP-S relaxation in the presence of non-submodular pairwise potentials.

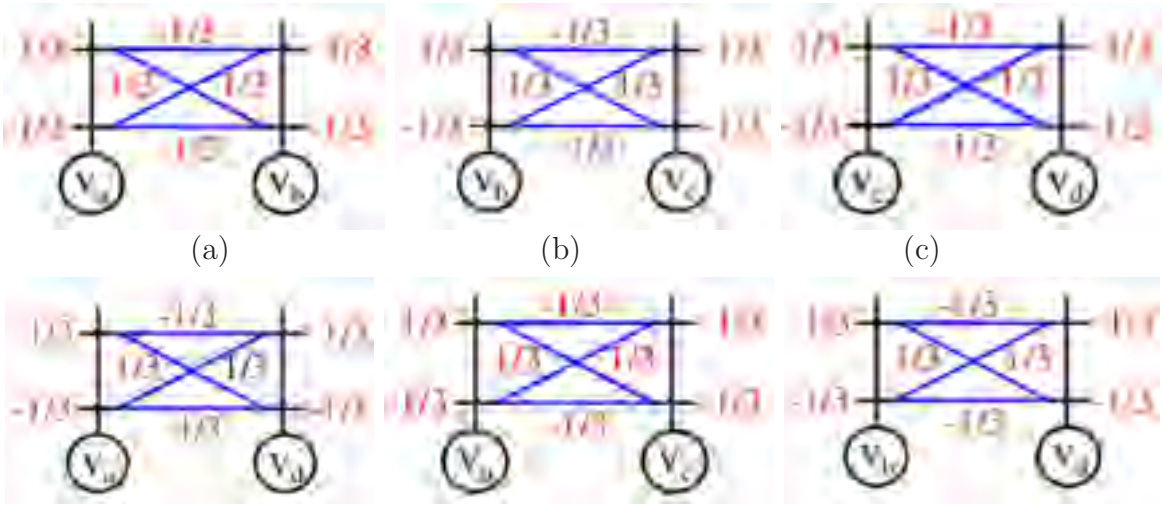


Figure 6.7: *An infeasible solution for* SOCP-Q. *The value of the variable $x_{a;i}$ is shown next to the i^{th} branch of the trellis on top of v_a. The value of $X_{ab;ij}$ is shown next to the connection between the i^{th} and the j^{th} branches of the trellises on top of v_b and v_b respectively. It can easily be verified that these variables satisfy all cycle inequalities. However, they do not belong to the feasibility region of* SOCP-Q *since they violate constraint (6.5.12).*

When the clique contains a frustrated cycle, it can be shown that SOCP-Q dominates the LP-S relaxation (similar to SOCP-C). Further, using a counter-example, it can proved that the feasibility region given by cycle inequalities is not a subset of the feasibility region defined by constraint (6.5.8). One such example is given below.

Example: We present an example to prove that the feasibility region given by cycle inequalities is not a subset of the feasibility region defined by the SOC constraint

$$\left(\sum_{k} x_{a_k;i_k}\right)^2 \leq q + \sum_{k,l} X_{a_k a_l;i_k i_l}, \tag{6.5.10}$$

which is used in SOCP-Q. Note that it would be sufficient to provide a set of variables $(\mathbf{x}, \mathbf{X})$ which satisfy the cycle inequalities but not constraint (6.5.10).

To this end, we consider a CRF defined over the random variables $\mathbf{v} = \{v_a, v_b, v_c, v_d\}$ which form a clique of size 4 with respect to the neighbourhood relationship $\mathcal{E}$, i.e.

$$\mathcal{E} = \{(a,b),(b,c),(c,d),(a,d),(a,c),(b,d)\}. \tag{6.5.11}$$

Each of these variables takes a label from the set $\mathbf{l} = \{l_0, l_1\}$. Consider the set of variables $(\mathbf{x}, \mathbf{X})$ shown in Fig. 6.7 which do not belong to the feasibility region of SOCP-Q. It can be easily shown that these variables satisfy all the cycle

inequalities (together with all the constraints of the LP-S relaxation). However, $(\mathbf{x}, \mathbf{X})$ defined in Fig. 6.7 does not belong to the feasibility region of the SOCP-Q relaxation since it does not satisfy the following SOC constraint:

$$\left(\sum_{v_a \in \mathbf{v}} x_{a;0}\right)^2 \leq 4 + 2\left(\sum_{(a,b)\in\mathcal{E}} X_{ab;00}\right). \tag{6.5.12}$$

6.6. Discussion

We presented an analysis of approximate algorithms for MAP estimation which are based on convex relaxations. The surprising result of our work is that despite the flexibility in the form of the objective function/constraints offered by QP and SOCP, the LP-S relaxation dominates a large class of QP and SOCP relaxations. It appears that the authors who have previously used SOCP relaxations in the Combinatorial Optimization literature [64] and those who have reported QP relaxation in the Machine Learning literature [75] were unaware of this result. We also proposed two new SOCP relaxations (SOCP-C and SOCP-Q) and presented some examples to prove that they provide a better approximation than LP-S. An interesting direction for future research would be to determine the best SOC constraints for a given MAP estimation problem (e.g. with truncated linear/quadratic pairwise potentials).

Chapter 7

Efficiently Solving Convex Relaxations

The problem of obtaining the maximum a posteriori (MAP) estimate of a discrete random field plays a central role in many applications. In the previous chapter, we presented an analysis of convex relaxations for MAP estimation. In this chapter, we build on the tree reweighted message passing (TRW) framework of [46, 101] which iteratively optimizes the Lagrangian dual of the linear programming relaxation called LP-S (described in chapters 3 and 6). Specifically, we show how the dual formulation of TRW can be extended to include cycle inequalities [4]. We then consider the inclusion of the second order cone constraints proposed in the previous chapter (which provide a better approximation for the MAP estimation problem) in the dual formulation.

We propose efficient iterative algorithms for solving the resulting duals. Similar to the method described in [46], these methods are guaranteed to converge (i.e. each iteration of algorithm does not decrease the value of the bounded dual). We test our algorithms on a large set of synthetic data, as well as real data. Our experiments show that the additional constraints (i.e. cycle inequalities and second order cone constraints) provide better results in cases where the TRW framework fails (namely MAP estimation for non-submodular energy functions).

7.1. Introduction

The problem of obtaining the maximum a posteriori (MAP) estimate of a discrete random field is of fundamental importance in many areas of Computer Science. In the previous chapter we presented an analysis of convex relaxations based approaches which attempt to solve the MAP estimation problem, namely LP-S [17, 51, 79, 101], QP-RL [75] and SOCP-MS [55, 64]. Specifically, we showed that a large class of quadratic programming (QP) and second order cone programming (SOCP) relaxations, including QP-RL and SOCP-MS, are dominated by the linear programming (LP) relaxation LP-S. Based on these results, we proposed two new SOCP relaxations which incorporate additional second order cone (SOC) constraints in the LP-S relaxation, i.e.

- SOCP-C, which include all the constraints of the LP-S relaxation in addition to SOC constraints that are defined over random variables which form a cycle in the graphical model of the random field. As mentioned in the previous chapter, these constraints are similar to the linear cycle inequalities [4]. We also conjectured that adding cycle inequalities to LP-S provides a better approximation of the MAP estimation problem than SOCP-C.

- SOCP-Q, which include all the constraints of the LP-S relaxation in addition to cycle inequalities and SOC constraints defined over random variables which form a clique in the graphical model of the random field.

Using some example random fields, we proved that SOCP-C and SOCP-Q dominate previously proposed convex relaxations [17, 51, 55, 64, 79, 101]. In theory, the SOCP-C and SOCP-Q relaxations allow us to obtain better MAP estimates. However, in practice, we are faced with the problem of solving these relaxations for discrete random fields defined over a large number of variables and labels. One method for obtaining the optimal solution of these relaxations is to use Interior Point algorithms which solve a large class of convex optimization problems. However, the state of the art Interior Point algorithms can handle only a few thousand variables and are usually extremely slow (e.g. the typical asymptotic time complexity of such algorithms is a 3^{rd} degree polynomial in the number of variables and constraints). Hence, there is a need to design efficient methods which are specific to the MAP estimation problem. In this work, we will address the issue of efficiently solving the convex relaxations proposed in the previous chapter.

Related Work: The LP-S relaxation has received considerable amount of attention in the literature. Specifically, several closely related algorithms have been proposed which solve the LP-S relaxation approximately [46, 79, 107, 101]. In

this work we build on the approach of Wainwright *et al.* [101] who presented two iterative algorithms to solve a particular formulation of the Lagrangian dual problem (hereby referred to as simply the dual) of the LP-S relaxation. Their dual formulation relies on a set of tree structured random fields (described in detail in the next section). Similar to min-sum belief propagation (min-sum BP), the algorithms in [101] are not guaranteed to converge. Kolmogorov [46] addressed this problem by proposing a convergent sequential tree-reweighted message passing (TRW-S) algorithm for solving the dual[1].

Despite its strong theoretical foundation, it was observed that TRW-S provides labellings with higher energies than min-sum BP when the energy function of the discrete random field contains non-submodular terms [46]. This is not surprising since the LP-S relaxation provides an inaccurate approximation in such cases (e.g. see § 6.5.1, Fig. 6.4 and 6.5). We address this issue by appending the LP-S relaxation with some useful constraints. Specifically, we incorporate an arbitrary number of linear cycle inequalities [4] in the dual of the LP-S relaxation (section 7.3). Furthermore, we show how the SOC constraints proposed in the previous chapter can be included within this framework (section 7.4). We also propose convergent algorithms for solving the resulting duals. Our experiments indicate that incorporating these constraints provides a better approximation for the MAP estimation problem compared to LP-S alone (section 7.5).

7.2. Preliminaries

We begin by recalling some of the notation and concepts introduced in earlier chapters. This would then allow us to concisely describe the MAP estimation approach of [46, 101]. Throughout this chapter we assume, for convenience, that the given discrete random field is a CRF defined over random variables $\mathbf{v} = \{v_0, v_1, \cdots, v_{n-1}\}$, each of which can take a label from the set $\mathbf{l} = \{l_0, l_1, \cdots, l_{h-1}\}$. However, we note that the results presented here are equally applicable to the MRF framework.

Given data $\mathbf{D}$ and a CRF with parameter $\boldsymbol{\theta}$, the energy of the labelling f, where $f : \{0, \cdots, n-1\} \longrightarrow \{0, \cdots, h-1\}$, is specified as

$$Q(f; \mathbf{D}, \boldsymbol{\theta}) = \sum_{v_a \in \mathbf{v}} \theta^1_{a;f(a)} + \sum_{(a,b) \in \mathcal{E}} \theta^2_{ab;f(a)f(b)}. \tag{7.2.1}$$

Recall that $\mathcal{E}$ is the neighbourhood relationship defined over the random variables $\mathbf{v}$ by the CRF. The terms $\theta^1_{a;f(a)}$ and $\theta^2_{ab;f(a)f(b)}$ are the unary and pairwise potentials respectively. The problem of MAP estimation is to find a labelling f^* with the minimum energy (or equivalently the maximum posterior probability), i.e.

$$f^* = \arg\min_f Q(f; \mathbf{D}, \boldsymbol{\theta}). \tag{7.2.2}$$

[1]We note here that TRW-S is closely related to the node-averaging algorithm of [79, 107].

For a given parameter $\boldsymbol{\theta}$, we denote the energy of an optimal (i.e. MAP) labelling as $q(\boldsymbol{\theta})$, i.e.

$$q(\boldsymbol{\theta}) = Q(f^*; \mathbf{D}, \boldsymbol{\theta}). \tag{7.2.3}$$

Min-marginals: We now describe the min-marginals associated with a CRF parameterized by $\boldsymbol{\theta}$. Recall that min-marginals were earlier introduced in § 2.3.2. Consider a labelling f which assigns a particular label l_i to a random variable v_a. The min-marginal of this assignment is defined as the minimum energy over all such labellings f. Formally, the min-marginal $q_{a;i}(\mathbf{D}, \boldsymbol{\theta})$ is given by

$$q_{a;i}(\mathbf{D}, \boldsymbol{\theta}) = \min_{f, f(a)=i} Q(f; \mathbf{D}, \boldsymbol{\theta}). \tag{7.2.4}$$

Similarly, we can define the min-marginal for assigning labels l_i and l_j to two neighbouring random variables v_a and v_b respectively as

$$q_{ab;ij}(\mathbf{D}, \boldsymbol{\theta}) = \min_{f, f(a)=i, f(b)=j} Q(f; \mathbf{D}, \boldsymbol{\theta}). \tag{7.2.5}$$

Note that, for the work described in this chapter, the data $\mathbf{D}$ will always remain fixed (since we are considering the MAP estimation problem). Hence, to make the text less cluttered, we will drop the term $\mathbf{D}$ from the notation of min-marginals. In other words, the min-marginal of assigning l_i to v_a (given parameter $\boldsymbol{\theta}$) will be denoted by $q_{a;i}(\boldsymbol{\theta})$ and the min-marginal for assigning l_i and l_j to v_a and v_b respectively will be denoted by $q_{ab;ij}(\boldsymbol{\theta})$.

Reparameterization: The concept of reparameterization plays an important role in many MAP estimation approaches (e.g. min-sum BP, st-MINCUT and TRW-S). Recall that a parameter $\overline{\boldsymbol{\theta}}$ is called a reparameterization of the parameter $\boldsymbol{\theta}$ (denoted by $\overline{\boldsymbol{\theta}} \equiv \boldsymbol{\theta}$) if, and only if,

$$Q(f; \mathbf{D}, \overline{\boldsymbol{\theta}}) = Q(f; \mathbf{D}, \boldsymbol{\theta}), \forall f. \tag{7.2.6}$$

The concept of reparameterization, as defined above, is meaningful only when considering the MAP estimation IP (since the definition only deals with non-fractional, i.e. integer, labellings), and not any of its convex relaxations.

Kolmogorov [46] proved that, for the MAP estimation IP, all reparameterizations can be expressed in the following form:

$$\overline{\theta}_{a;i} = \theta_{a;i} + \sum_{b,(a,b)\in\mathcal{E}} M_{ba;i}, \quad \overline{\theta}_{ab;ij} = \theta_{ab;ij} - M_{ba;i} - M_{ab;j}. \tag{7.2.7}$$

It is worth remembering this result as it will be used at several places throughout this chapter.

LP-S **Relaxation:** Since we build on the work of [46, 101], we would find it convenient to use their notation to describe the LP-S relaxation. To this end, we specify an over-complete representation of a labelling f using binary variables $y_{a;i}$ such that

$$y_{a;i} = \begin{cases} 1 & \text{if} \quad f(a) = i, \\ 0 & \text{otherwise.} \end{cases} \tag{7.2.8}$$

We also specify binary variables $y_{ab;ij}$ for all $(a,b) \in \mathcal{E}$ and $l_i, l_j \in \mathbf{l}$ such that $y_{ab;ij} = y_{a;i} y_{b;j}$. The vector $\mathbf{y}$ is then defined as consisting of variables $y_{a;i}$ and $y_{ab;ij}$ (in the order which matches the arrangement of potentials $\theta^1_{a;i}$ and $\theta^2_{ab;ij}$ in the parameter vector $\boldsymbol{\theta}$). Note that the variable $\mathbf{y}$ is closely related to the binary variables $(\mathbf{x}, \mathbf{X})$ used to describe the Integer Programming (IP) formulation of the MAP estimation problem in § 3.3.1. Specifically,

$$x_{a;i} = 2y_{a;i} - 1, \quad X_{ab;ij} = 4y_{ab;ij} - 2y_{a;i} - 2y_{b;j} + 1. \tag{7.2.9}$$

We will sometimes specify the additional constraints (i.e. cycle inequalities and the SOC constraints described in the previous chapter) using variables $(\mathbf{x}, \mathbf{X})$, since they will allow us to write these constraints concisely. However, it is worth noting that any constraint on $(\mathbf{x}, \mathbf{X})$ can be converted to a constraint on $\mathbf{y}$ using the relationship in equation (7.2.9).

Using the vector $\mathbf{y}$, the problem of MAP estimation can be formulated as an IP as follows:

$$\mathbf{y}^* = \arg\min_{\mathbf{y}} \mathbf{y}^\top \boldsymbol{\theta},$$
$$MARG(\mathbf{v}, \mathcal{E}) = \begin{cases} y_{a;i} \in \{0,1\}, y_{ab;ij} \in \{0,1\}, & \forall v_a \in \mathbf{v}, (a,b) \in \mathcal{E}, l_i, l_j \in \mathbf{l}, \\ \sum_{l_i \in \mathbf{l}} y_{a;i} = 1, & \forall v_a \in \mathbf{v}, \\ y_{ab;ij} = y_{a;i} y_{b;j}, & \forall (a,b) \in \mathcal{E}, l_i, l_j \in \mathbf{l}. \end{cases} \tag{7.2.10}$$

The term $MARG(\mathbf{v}, \mathcal{E})$ stands for *marginal polytope* which is defined as the feasibility region of the MAP estimation IP. This notation has been borrowed from [101]. Note that the above IP can easily be shown to be equivalent to the IP described in § 3.3.1 (by converting variables $\mathbf{y}$ to $(\mathbf{x},\mathbf{X})$). The LP-S relaxation is formulated as follows:

$$\mathbf{y}^* = \arg\min_{\mathbf{y} \in LOCAL(\mathbf{v}, \mathcal{E})} \mathbf{y}^\top \boldsymbol{\theta},$$
$$LOCAL(\mathbf{v}, \mathcal{E}) = \begin{cases} y_{a;i} \in [0,1], y_{ab;ij} \in [0,1], & \forall v_a \in \mathbf{v}, (a,b) \in \mathcal{E}, l_i, l_j \in \mathbf{l}, \\ \sum_{l_i \in \mathbf{l}} y_{a;i} = 1, & \forall v_a \in \mathbf{v}, \\ \sum_{l_j \in \mathbf{l}} y_{ab;ij} = y_{a;i}, & \forall v_a \in \mathbf{v}, l_i \in \mathbf{l}. \end{cases} \tag{7.2.11}$$

The term $LOCAL(\mathbf{v}, \mathcal{E})$ stands for *local polytope* which is defined as the feasibility region of the LP-S relaxation [101] (i.e. the set of all vectors $\mathbf{y}$ with non-negative elements that satisfy the uniqueness and the marginalization constraints). Note that by definition of relaxation $MARG(\mathbf{v}, \mathcal{E}) \subset LOCAL(\mathbf{v}, \mathcal{E})$.

Dual of the LP-S Relaxation: Before providing the dual formulation of [46, 101] for the LP-S relaxation, we need the following definitions. Let $\mathcal{T}$ represent a set of tree-structured CRFs, each of which is defined over a subset of variables $\mathbf{v}$. In other words, if the CRF $T \in \mathcal{T}$ is defined over variables $\mathbf{v}_T$, then $\mathbf{v}_T \subseteq \mathbf{v}$. Further, if $\mathcal{E}_T$ is the neighbourhood relationship of the CRF T, then $\mathcal{E}_T \subseteq \mathcal{E}$ such that the graphical model of T is connected and has no loops (i.e. it forms a tree). We represent the parameter of this CRF T as $\boldsymbol{\theta}^T$, which is the same size as the parameter vector $\boldsymbol{\theta}$ of the original CRF. In other words, $\theta^{T1}_{a;i} = 0$ if $v_a \notin \mathbf{v}_T$ and $\theta^{T2}_{ab;ij} = 0$ if $(a,b) \notin \mathcal{E}_T$, where $\theta^{T1}_{a;i}$ and $\theta^{T2}_{ab;ij}$ are the unary and pairwise potentials respectively.

Let $\boldsymbol{\rho} = \{\rho(T), T \in \mathcal{T}\}$ be a set of non-negative real numbers which sum to one. For such a set $\boldsymbol{\rho}$ we define the variable and edge appearance term[1] as

$$\rho_a = \sum_{T, v_a \in \mathbf{v}_T} \rho(T), \tag{7.2.12}$$

$$\rho_{ab} = \sum_{T, (a,b) \in \mathcal{E}_T} \rho(T), \tag{7.2.13}$$

respectively for all $v_a \in \mathbf{v}$ and $(a,b) \in \mathcal{E}$. In other words, the variable and edge appearance terms are the sum of terms $\rho(T)$ over all tree structured CRFs T which contain that variable or edge respectively. Typically, the terms $\rho(T)$ are set as $\frac{1}{|\mathcal{T}|}$ for all tree structured CRFs $T \in \mathcal{T}$. This assignment is purely for convenience and it is not yet known whether it significantly effects the performance of the algorithms described in [46, 101]. As an example, consider three tree structured CRFs which together define a cycle of size 3 (as shown in Fig. 7.1). If we assign $\rho(T) = \frac{1}{3}$ for all three trees, then we obtain the following variable and edge appearance terms:

$$\begin{aligned} &\rho_a = \tfrac{2}{3}, \rho_b = \tfrac{2}{3}, \rho_c = \tfrac{2}{3}, \\ &\rho_{ab} = \tfrac{1}{3}, \rho_{bc} = \tfrac{1}{3}, \rho_{ac} = \tfrac{1}{3}. \end{aligned} \tag{7.2.14}$$

Using the above notation, the dual of the LP-S relaxation can be written as follows [46, 101]:

$$\max_{\sum_{T \in \mathcal{T}} \rho(T) \boldsymbol{\theta}^T \equiv \boldsymbol{\theta}} \sum_T \rho(T) q(\boldsymbol{\theta}^T). \tag{7.2.15}$$

Recall that $q(\boldsymbol{\theta}^T)$ indicates the energy of the MAP labelling of a CRF parameterized by $\boldsymbol{\theta}^T$. The above dual is valid for any choice of sets $\mathcal{T}$ and $\boldsymbol{\rho}$ as long as $\rho_a > 0$ and $\rho_{ab} > 0$ for all $v_a \in \mathbf{v}$ and $(a,b) \in \mathcal{E}$ [46, 101]. For the sake of completeness, we provide the proof that problem (7.2.15) is the dual of the LP-S relaxation. This proof is based on [101] (along with its correction in [46]) and will be used in subsequent sections.

[1] Note that in [46, 101], these terms were referred to as node and edge appearance probabilities respectively. However, we prefer to use the word *term* instead of *probabilities* as their sum over all variables or all edges is not necessarily 1 (i.e. they are not actually probabilities).

Theorem of [46, 101]: The problem (7.2.15) is the dual of the LP-S relaxation.

Proof: We begin by writing the dual of the following problem:

$$\begin{aligned} \max \quad & \textstyle\sum_T \rho(T) q(\boldsymbol{\theta}^T), \\ \text{s.t.} \quad & \textstyle\sum_T \rho(T) \boldsymbol{\theta}^T = \boldsymbol{\theta}, \end{aligned} \tag{7.2.16}$$

This was the problem considered in [101]. The equivalence of problems (7.2.16) and (7.2.15) (i.e. using the equality constraint $\sum_T \rho(T)\boldsymbol{\theta}^T = \boldsymbol{\theta}$ and the reparameterization constraint $\sum_T \rho(T)\boldsymbol{\theta}^T \equiv \boldsymbol{\theta}$) was shown in [46]. After rearranging the terms, the dual of problem (7.2.16) can be written as

$$\min_{\boldsymbol{\tau}} \max_{\boldsymbol{\theta}^T} \sum_T \rho(T)(q(\boldsymbol{\theta}^T) - \boldsymbol{\tau}^\top \boldsymbol{\theta}^T), \tag{7.2.17}$$

where $\boldsymbol{\tau}$ is a vector of (non-negative) Lagrangian multipliers. Using Lemma 2 of [101] we see that

$$\max_{\boldsymbol{\theta}^T} \sum_T \rho(T)(q(\boldsymbol{\theta}^T) - \boldsymbol{\tau}^\top \boldsymbol{\theta}^T) = \begin{cases} 0 & \text{if} \quad \boldsymbol{\tau} \in LOCAL(\mathbf{v}, \mathcal{E}), \\ \infty & \text{otherwise.} \end{cases} \tag{7.2.18}$$

Thus problem (7.2.17) can be reformulated as

$$\begin{aligned} & \min\nolimits_{\boldsymbol{\tau}} \boldsymbol{\tau}^\top \boldsymbol{\theta}, \\ \text{s.t.} \quad & \boldsymbol{\tau} \in LOCAL(\mathbf{v}, \mathcal{E}), \end{aligned} \tag{7.2.19}$$

which is the same as the LP-S relaxation. Hence, the dual of the LP-S relaxation is given by problem (7.2.16). Using the properties of the constraint set of LP-S (i.e. the constraints which specify $LOCAL(\mathbf{v}, \mathcal{E})$), we see that problem (7.2.16) is equivalent to problem (7.2.15). This follows from the result that all reparameterizations are of the form shown in equation (7.2.7), which implies that LP-S provides the same solution for all reparameterizations (due to the presence of uniqueness and marginalization constraint). This proves the theorem. ∎

The TRW-S Algorithm: Table 7.1 describes the TRW-S algorithm [46] which attempts to solve the dual of the LP-S relaxation. In other words, it solves for the set of parameters $\boldsymbol{\theta}^T$, $T \in \mathcal{T}$, which maximize the dual (7.2.15). There are two main steps: (i) reparameterization, which involves running one pass of min-sum BP on the tree structured CRFs T[2]; and (ii) averaging operation. TRW-S is guaranteed to not to decrease the value of the dual (7.2.15) at each iteration. Further, it can be shown that it converges to a solution which satisfies the weak tree agreement described below.

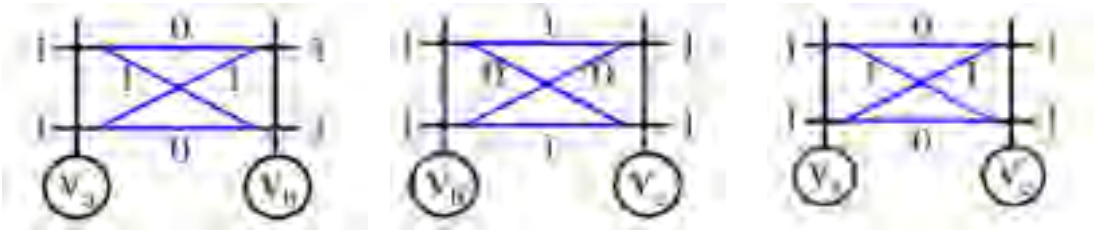
[2]Recall that min-sum BP provides the exact min-marginals of the (arbitrarily chosen) root variable of a tree-structured random field in one iteration called forward pass (see § 2.4.1).

Weak Tree Agreement: Let $OPT(\boldsymbol{\theta}^T)$ be the set of all optimal (i.e. MAP) labellings of a CRF with parameter $\boldsymbol{\theta}^T$. Note that a set $OPT(\boldsymbol{\theta}^T)$ may contain more than one labelling (i.e. the optimum labelling of a CRF is not necessarily unique). Further, for all $\omega \in \mathbf{v} \bigcup \mathcal{E}$, let $\mathcal{T}_\omega \subseteq \mathcal{T}$ be the set of all tree-structured CRF which contain ω. The parameters $\boldsymbol{\theta}^T$, $T \in \mathcal{T}$, are said to satisfy the weak tree agreement (WTA) condition if, and only if, for all ω and for all pairs of parameters $\boldsymbol{\theta}^{T_1}$ and $\boldsymbol{\theta}^{T_2}$, where $T_1, T_2 \in \mathcal{T}_\omega$, there exist labellings $f_1^* \in OPT(\boldsymbol{\theta}^{T_1})$ and $f_2^* \in OPT(\boldsymbol{\theta}^{T_2})$ which agree on the labelling of ω. In other words, if $\omega = v_a \in \mathbf{v}$ then $f_1^*(a) = f_2^*(a)$, and if $\omega = (a, b) \in \mathcal{E}$ then $f_1^*(a) = f_2^*(a)$ and $f_1^*(b) = f_2^*(b)$.

For example, Fig. 7.1 shows the parameters of three tree structured CRFs (denoted by $\boldsymbol{\theta}^{T_1}$, $\boldsymbol{\theta}^{T_2}$ and $\boldsymbol{\theta}^{T_3}$ respectively) which satisfy the WTA condition. This can be verified by observing that

$$\begin{aligned} OPT(\boldsymbol{\theta}^{T_1}) &= \{\{0,0\},\{1,1\}\}, \\ OPT(\boldsymbol{\theta}^{T_2}) &= \{\{0,1\},\{1,0\}\}, \\ OPT(\boldsymbol{\theta}^{T_3}) &= \{\{1,1\},\{1,1\}\}. \end{aligned} \tag{7.2.20}$$

Thus if we consider $\omega = v_b$, then for parameters $\boldsymbol{\theta}^{T_1}$ and $\boldsymbol{\theta}^{T_2}$ there exist labellings $f_1^* = \{0,0\} \in OPT(\boldsymbol{\theta}^{T_1})$ and $f_2^* = \{0,1\} \in OPT(\boldsymbol{\theta}^{T_2})$ which agree on the label for v_b (i.e. the label l_0).



(a) (b) (c)

Figure 7.1: *Example of the* WTA *condition. Three tree structured* CRF*s defined over two variables, each of which can take one of two labels* $\mathbf{l} = \{l_0, l_1\}$*. All the unary potentials, shown next to the branches of the trellis, are* 1*. The pairwise potentials (shown next to the connections between branches of neighbouring trellises) in the* **(a)** *and* **(c)** *form an Ising model. The pairwise potentials between* v_b *and* v_c*, shown in* **(b)***, are non-submodular. The* TRW-S *algorithm converges when it reaches this solution. However, it is not obvious how to obtain the primal solution (i.e. the labelling of the variables* v_a*,* v_b *and* v_c*) using these parameters* $\boldsymbol{\theta}^T$*, since for all parameters both* l_0 *and* l_1 *are the optimal solutions for all the variables.*

Note that TRW-S only provides us with the set of parameters $\boldsymbol{\theta}^T$ which satisfy the WTA condition. In other words, TRW-S solves only the dual (7.2.15) and not the primal problem (i.e. the LP-S relaxation) itself. If the energy $Q(\cdot; \mathbf{D}, \boldsymbol{\theta})$ of

the CRF is a binary submodular function, the method of [47] provides an optimal labelling once the WTA condition is satisfied. However, a binary submodular energy function can be minimized more efficiently using the st-MINCUT approach (see § 2.4.3). A more interesting property is that if the set $OPT(\boldsymbol{\theta}^T)$ contained a unique labelling for all $\boldsymbol{\theta}^T$ which satisfy the WTA condition, then the union of all sets $OPT(\boldsymbol{\theta}^T)$, $T \in \mathcal{T}$, would provide us with the exact MAP estimate of the original CRF $\boldsymbol{\theta}$ [101]. However, in practice, this is often not the case (e.g. see Fig. 7.1). Hence, the final labelling of the random variables is found by a sequential procedure used in [63, 46] (described in § 7.3.1). We refer the reader to [46] for details.

7.3. Adding Linear Constraints

We now show how the results of [46, 101] can be extended to include an arbitrary number of linear cycle inequalities [4] which are defined as follows. Consider a cycle of length c in the graphical model of the given CRF, which is specified over a set of random variables $\mathbf{v}_C = \{v_b, b = a_1, a_2, \cdots, a_c\}$ such that $\mathcal{E}_C = \{(a_1, a_2), (a_2, a_3), \cdots, (a_n, a_1)\} \subseteq \mathcal{E}$. Further, let $\mathcal{E}_F \subseteq \mathcal{E}_C$ such that $|\mathcal{E}_F|$ (i.e. the cardinality of $\mathcal{E}_F$) is odd. Using these sets of edges, a cycle inequality can be specified as

$$\sum_{(a_k,a_l)\in\mathcal{E}_F} X_{a_k a_l;i_k i_l} - \sum_{(a_k,a_l)\in\mathcal{E}_C-\mathcal{E}_F} X_{a_k a_l;i_k i_l} \geq 2 - c, \tag{7.3.1}$$

where $\mathbf{l}_C = \{l_j, j = i_1, i_2, \cdots, i_c\} \in \mathbf{l}^c$. The variables $X_{a_k a_l;i_k i_l}$ are defined in equation (7.2.9). For example, consider a case where $\mathbf{v}_C = \{v_a, v_b, v_c\}$ (i.e. $c = 3$) and $\mathbf{l}_C = \{l_0, l_0, l_0\}$. Note that, in this case, $\mathcal{E}_C = \{(a, b), (b, c), (a, c)\}$. By choosing $\mathcal{E}_F = \{(a, b), (b, c), (a, c)\}$, we can obtain the following cycle inequality:

$$X_{ab;00} + X_{bc;00} + X_{ac;00} \geq -1. \tag{7.3.2}$$

Similarly, when $\mathbf{v}_C = \{v_a, v_b, v_c, v_d\}$ (i.e. $c = 4$), $\mathbf{l}_C = \{l_0, l_0, l_0, l_0\}$ and $\mathcal{E}_F = \{(a, b), (b, c), (c, d)\}$, we get

$$X_{ab;00} + X_{bc;00} + X_{cd;00} - X_{ad;00} \geq -2. \tag{7.3.3}$$

Note that cycle inequalities are valid constraints, i.e. they can be included in a relaxation of the MAP estimation IP. Further, it can be shown that adding cycle inequalities to LP-S, i.e. problem (7.2.11), provides a better relaxation (i.e. the resulting relaxation dominates LP-S) [4]. In other words, the feasibility region obtained by including cycle inequalities is a strict subset of the feasibility region of problem (7.2.11).

In general, a set of N_C cycle inequalities defined on a cycle $C = (\mathbf{v}_C, \mathcal{E}_C)$ (using different sets $\mathbf{l}_C$) can be written as

$$\mathbf{A}^C \mathbf{y} \geq \mathbf{b}^C. \tag{7.3.4}$$

Initialization

1. For every $\omega \in \mathbf{v} \bigcup \mathcal{E}$, find all trees $\mathcal{T}_\omega \subseteq \mathcal{T}$ which contains ω.
2. Initialize $\boldsymbol{\theta}^T$ such that $\sum_T \rho(T)\boldsymbol{\theta}^T \equiv \boldsymbol{\theta}$.
   Typically, we set $\rho(T) = 1/|\mathcal{T}|$ for all $T \in \mathcal{T}$.
   Then we can initialize $\theta^{T1}_{a;i} = \theta^1_{a;i}\frac{|\mathcal{T}|}{|\mathcal{T}_{v_a}|}$ for all $T \in \mathcal{T}_{v_a}$.
   Similarly, $\theta^{T2}_{ab;ij} = \theta^2_{ab;ij}\frac{|\mathcal{T}|}{|\mathcal{T}_{(a,b)}|}$ for all $T \in \mathcal{T}_{(a,b)}$.
   The above values are chosen for convenience.
   TRW-S is equally applicable for other values of the above terms.

Iterative Steps

3. Pick an element $\omega \in \mathbf{v} \bigcup \mathcal{E}$.
4. For all $T \in \mathcal{T}_\omega$, reparameterize $\boldsymbol{\theta}^T$ to $\overline{\boldsymbol{\theta}}^T$ such that
   (i) $\overline{\theta}^{T1}_{a;i} = q_{a;i}(\boldsymbol{\theta}^T)$, if $\omega = v_a \in \mathbf{v}$,
   (ii) $\overline{\theta}^{T1}_{a;i} + \overline{\theta}^{T1}_{b;j} + \overline{\theta}^{T2}_{ab;ij} = q_{ab;ij}(\boldsymbol{\theta}^T)$, if $\omega = (a,b) \in \mathcal{E}$.
   This step involves running one iteration (i.e. the forward pass) of min-sum BP for all trees $T \in \mathcal{T}_\omega$ (see § 2.4.1).
5. Averaging operation:
   (i) If $\omega = v_a \in \mathbf{v}$,
      (a) Compute $\nu_{a;i} = \frac{1}{\rho_a}\sum_{T \in \mathcal{T}_a} \rho(T)\overline{\theta}^{T1}_{a;i}$.
      (b) Set $\overline{\theta}^{T1}_{a;i} = \nu_{a;i}$, for all $T \in \mathcal{T}_{v_a}$.
   (ii) If $\omega = (a,b) \in \mathcal{T}$,
      (a) Compute $\nu_{ab;ij} = \frac{1}{\rho_{ab}}\sum_{T \in \mathcal{T}_{(a,b)}} \rho(T)(\overline{\theta}^{T1}_{a;i} + \overline{\theta}^{T1}_{b;j} + \overline{\theta}^{T2}_{ab;ij})$.
      (b) Set $\overline{\theta}^{T1}_{a;i} + \overline{\theta}^{T1}_{b;j} + \overline{\theta}^{T2}_{ab;ij} = \nu_{ab;ij}$, for all $T \in \mathcal{T}_{(a,b)}$.
6. Repeat steps 3, 4 and 5 till convergence.

Table 7.1: *The* TRW-S *algorithm. Recall that* $\theta^{T1}_{a;i}$ *and* $\theta^{T2}_{ab;ij}$ *are the unary and pairwise potentials for the parameter* $\boldsymbol{\theta}^T$*. Similarly,* $\overline{\theta}^{T1}_{a;i}$ *and* $\overline{\theta}^{T2}_{ab;ij}$ *are the unary and pairwise potentials defined by the parameter* $\overline{\boldsymbol{\theta}}$*. The terms* ρ_a *and* ρ_{ab} *are the variable and edge appearance terms for* $v_a \in \mathbf{v}$ *and* $(a,b) \in \mathcal{E}$ *respectively. In step 3, the value of the dual (7.2.15) remains unchanged. Step 4, i.e. the averaging operation, ensures that the value of the dual does not decrease.* TRW-S *converges to a solution which satisfies the* WTA *condition.*

For every cycle we can define upto $|\mathbf{l}|^c$ cycle inequalities using some or all sets $\mathbf{l}_C \in \mathbf{l}^c$ (i.e. $N_C \in \{0, 1, \cdots, |\mathbf{l}|^c\}$). Note that the variables $(\mathbf{x}, \mathbf{X})$ can be converted to variable $\mathbf{y}$ using equation (7.2.9), which allows us to represent cycle inequalities in the form (7.3.4). Let $\mathcal{C}$ be a set of cycles in the given CRF. Theorem 1 (given below) provides us with the dual of the LP relaxation obtained by appending problem (7.2.11) with cycle inequalities (defined over cycles in the set $\mathcal{C}$). We refer to the resulting relaxation as LP-C (where C denotes cycles).

Theorem 1: The following problem is the dual of problem (7.2.11) appended

with a set of cycle inequalities $\mathbf{A}^C\mathbf{y} \geq \mathbf{b}^C$, for all $C \in \mathcal{C}$ (hereby referred to as the LP-C relaxation):

$$\begin{aligned} \max \quad & \textstyle\sum_T \rho(T) q(\boldsymbol{\theta}^T) + \sum_C \rho'(C)(\mathbf{b}^C)^\top \mathbf{u}^C, \\ \text{s.t.} \quad & \textstyle\sum_T \rho(T)\boldsymbol{\theta}^T + \sum_C \rho'(C)(\mathbf{A}^C)^\top \mathbf{u}^C \equiv \boldsymbol{\theta}, \\ & u_k^C \geq 0, \forall k \in \{1, 2, \cdots, N_C\}, C \in \mathcal{C}. \end{aligned} \tag{7.3.5}$$

Here $\boldsymbol{\rho}' = \{\rho'(C), C \in \mathcal{C}\}$ is some (fixed) set of non-negative real numbers which sum to one, and $\mathbf{u}^C = \{u_k^C, k = 1, \cdots, N_C\}$ are some non-negative slack variables.

Proof: We begin by writing the dual of the following problem, which replaces the reparameterization constraint in problem (7.3.5) with the equality constraint, i.e.

$$\begin{aligned} \max \quad & \textstyle\sum_T \rho(T) q(\boldsymbol{\theta}^T) + \sum_C \rho'(C)(\mathbf{b}^C)^\top \mathbf{u}^C, \\ \text{s.t.} \quad & \textstyle\sum_T \rho(T)\boldsymbol{\theta}^T + \sum_C \rho'(C)(\mathbf{A}^C)^\top \mathbf{u}^C = \boldsymbol{\theta}, \\ & u_k^C \geq 0, \forall k \in \{1, 2, \cdots, N_C\}, C \in \mathcal{C}. \end{aligned} \tag{7.3.6}$$

The dual of problem (7.3.6) can be written as

$$\min_{\boldsymbol{\tau}, \boldsymbol{\mu}} \left(\max_{\boldsymbol{\theta}^T, \mathbf{u}^C} \sum_T \rho(T)(q(\boldsymbol{\theta}^T) - \boldsymbol{\tau}^\top \boldsymbol{\theta}^T) + \sum_C \rho'(C)(\mathbf{b}^C + \boldsymbol{\mu} - \mathbf{A}^C\boldsymbol{\tau}))^\top \mathbf{u}^C + \boldsymbol{\tau}^T \boldsymbol{\theta} \right), \tag{7.3.7}$$

where $\boldsymbol{\tau}$ and $\boldsymbol{\mu}$ are some (non-negative) Lagrangian multipliers (see section 3.2). Using equation (7.2.18) the above problem can be simplified as

$$\begin{aligned} \min_{\boldsymbol{\tau}, \boldsymbol{\mu}} \max_{\mathbf{u}^C} \quad & \textstyle\sum_C \rho'(C)(\mathbf{b}^C + \boldsymbol{\mu} - \mathbf{A}^C\boldsymbol{\tau}))^\top \mathbf{u}^C + \boldsymbol{\tau}^T \boldsymbol{\theta}, \\ \text{s.t.} \quad & \boldsymbol{\tau} \in LOCAL(\mathbf{v}, \mathcal{E}). \end{aligned} \tag{7.3.8}$$

Furthermore,

$$\min_{\boldsymbol{\tau}, \boldsymbol{\mu}} \max_{\mathbf{u}^C} \rho'(C)(\mathbf{b}^C + \boldsymbol{\mu} - \mathbf{A}^C\boldsymbol{\tau}))^\top \mathbf{u}^C = \begin{cases} 0 & \text{if} \quad \mathbf{A}^C\boldsymbol{\tau} = \mathbf{b}^C + \boldsymbol{\mu}, \\ \infty & \text{otherwise.} \end{cases} \tag{7.3.9}$$

This can easily be verified by assuming $\mathbf{A}^C\boldsymbol{\tau} \neq \mathbf{b}^C + \boldsymbol{\mu}$, in which case each slack variable u_k^C which maximizes the LHS of equation (7.3.9) will take one of the following values:

$$u_k^C = \begin{cases} +\infty & \text{if} \quad b_k^C + \mu_k > (\mathbf{A}^C\boldsymbol{\tau})_k \\ -\infty & \text{otherwise,} \end{cases} \tag{7.3.10}$$

where $(\mathbf{A}^C\boldsymbol{\tau})_k$ indicates the k^{th} element of the vector $\mathbf{A}^C\boldsymbol{\tau}$. Either of these values of u_k^C would result in a value of $+\infty$ for the expression in the LHS of

equation (7.3.9). In other words, for the LHS of equation (7.3.9) to be bounded, the following condition must hold true:

$$\mathbf{A}^C\boldsymbol{\tau} = \mathbf{b}^C + \boldsymbol{\mu}, \tag{7.3.11}$$

$$\Rightarrow \mathbf{A}^C\boldsymbol{\tau} \geq \mathbf{b}^C. \tag{7.3.12}$$

The last expression is obtained using the fact that $\boldsymbol{\mu} \geq 0$. Thus problem (7.3.8) can be reformulated as

$$\begin{aligned} &\min_{\boldsymbol{\tau}} \boldsymbol{\tau}^\top \boldsymbol{\theta}, \\ \text{s.t.} \quad &\boldsymbol{\tau} \in LOCAL(\mathbf{v}, \mathcal{E}), \mathbf{A}^C\boldsymbol{\tau} \geq \mathbf{b}^C, \forall C \in \mathcal{C}, \end{aligned} \tag{7.3.13}$$

which is the same as the LP-C relaxation. Hence, the dual of the LP-C relaxation is given by problem (7.3.6). Using the fact that the constraint set of LP-C includes all the constraints of LP-S (i.e. the uniqueness and marginalization constraints) it follows that problem (7.3.6) is equivalent to problem (7.3.5) Recall that this is a direct consequence of the form of reparameterizations (given in equation (7.2.7)) which implies that LP-S provides the same solution for all reparameterizations. Hence, LP-C also provides the same solution for all reparameterizations (see [46] for details). This proves Theorem 1. ∎

Note that the above theorem provides the dual for an arbitrary number of constraints N_C per cycle (i.e. N_C can be any number between 0 and $|\mathbf{l}|^c$. In our experiments, we found it sufficient to use $N_C = 1$ (see section 7.5 for details). Similar to the dual (7.2.15) used in TRW-S, the above problem can be solved using standard software for only a small number of variables $\mathbf{v}$. In order to overcome this deficiency we propose a convergent algorithm (similar to TRW-S) to approximately solve problem (7.3.5). We call our approach the TRW-S(LP-C) algorithm. In order to describe TRW-S(LP-C), we need the following definitions.

We say that a tree structured random field $T = (\mathbf{v}_T, \mathcal{E}_T) \in \mathcal{T}$ *belongs to* a cycle $C = (\mathbf{v}_C, \mathcal{E}_C) \in \mathcal{C}$ (denoted by $T \in C$) if, and only if, there exists an edge $(a, b) \in \mathcal{E}_T$ such that $(a, b) \in \mathcal{E}_C$. In other words, $T \in C$ if they share a common pair of neighbouring random variables $(a, b) \in \mathcal{E}$. We also define the following problem:

$$\begin{aligned} \max \quad &\textstyle\sum_{T \in C} \rho(T) q(\boldsymbol{\theta}^T) + \rho'(C)(\mathbf{b}^C)^\top \mathbf{u}^C, \\ \text{s.t.} \quad &\textstyle\sum_{T \in C} \rho(T) \boldsymbol{\theta}^T + \rho'(C)(\mathbf{A}^C)^\top \mathbf{u}^C = \boldsymbol{\theta}^C, \\ &u^C_k \geq 0, \forall k \in \{1, 2, \cdots, N_C\}, \end{aligned} \tag{7.3.14}$$

for some parameter $\boldsymbol{\theta}^C$. The above problem is the dual for the LP-C relaxation over one cycle C with parameter $\boldsymbol{\theta}^C$ (with any number of cycle inequalities defined over it). Note that problem (7.3.14) has fewer variables and constraints than dual (7.3.5) and can be solved easily using standard Interior Point algorithms for small cycles C. As will be seen in section 7.5, even using cycles of size 3

or 4 results in much better approximations of the MAP estimation problem for non-submodular energy functions.

Table 7.2 describes the convergent TRW-S(LP-C) algorithm for approximately solving the dual (7.3.5). The algorithm consists of two main steps : (i) solving problem (7.3.14) for a cycle; and (ii) running steps 4 and 5 of the TRW-S algorithm. The properties of the algorithm are summarized below.

Initialization
1. Choose a set of tree structured random fields $\mathcal{T}$ (similar to TRW-S). Choose a set of cycles $\mathcal{C}$. For example, if the 4-neighbourhood is employed, $\mathcal{C}$ can be the set of all cycles of size 4. For 8-neighbourhood, $\mathcal{C}$ can be all cycles of size 3.
2. Initialize $\boldsymbol{\theta}^T$ such that $\sum_T \rho(T)\boldsymbol{\theta}^T \equiv \boldsymbol{\theta}$. Initialize $u_k^C = 0$ for all C and k.
Iterative Steps
3. Pick an element $\omega \in \mathbf{v} \bigcup \mathcal{C}$. Find all cycles $\mathcal{C}_\omega \subseteq \mathcal{C}$ which contains ω.
4. For a cycle $C \in \mathcal{C}_\omega$, compute $\boldsymbol{\theta}^C = \sum_{T \in C} \rho(T)\boldsymbol{\theta}^T + \rho'(C)(\mathbf{A}^C)^\top \mathbf{u}^C$ using the values of $\boldsymbol{\theta}^T$ and $\mathbf{u}^C$ obtained in the previous iteration. Solve problem (7.3.14) using an Interior Point method. Update the values of $\boldsymbol{\theta}^T$ and $\mathbf{u}_C$.
5. For all trees $T \in \mathcal{T}$ which contain ω, run steps 4 and 5 of the TRW-S algorithm.
6. Repeat steps 3 and 4 for all cycles $C \in \mathcal{C}_\omega$.
7. Repeat steps 3 to 5 for all elements ω till convergence.

Table 7.2: *The* TRW-S*(*LP-C*) algorithm.*

7.3.1 Properties of the TRW-S(LP-C) Algorithm.

Property 1: At each step of the algorithm, the reparameterization constraint is satisfied, i.e.

$$\sum_T \rho(T)\boldsymbol{\theta}^T + \sum_C \rho'(C)(\mathbf{A}^C)^\top \mathbf{u}^C \equiv \boldsymbol{\theta}. \tag{7.3.15}$$

The constraint in problem (7.3.14) ensures that parameter vector $\boldsymbol{\theta}^C$ of cycle C remains unchanged. Hence, after step 3 of the TRW-S(LP-C) algorithm, the reparameterization constraint is satisfied. It was also shown that step 4 (i.e. running TRW-S) provides a reparameterization of $\boldsymbol{\theta}$ (see Lemma 3.3 of [46] for details). This proves Property 1.

Property 2: At each step of the algorithm, the value of the dual (7.3.5) never decreases.

Clearly, step 3 of the TRW-S(LP-C) algorithm does not decrease the value of the dual (7.3.5) (since the objective function of problem (7.3.14) is part of the objective function of dual (7.3.5)). Kolmogorov [46] showed that step 4 (i.e. TRW-S) also does not decrease this value. Note that the LP-C relaxation is guaranteed to be bounded since it dominates the LP-S relaxation (which itself is bounded [46]). Therefore, by the Bolzano-Weierstrass theorem [27], it follows that TRW-S(LP-C) is guaranteed to converge.

Property 3: Like TRW-S, the necessary condition for convergence of TRW-S(LP-C) is that the parameter vectors $\boldsymbol{\theta}^T$ of the trees $T \in \mathcal{T}$ satisfy WTA.

This follows from the fact that TRW-S increases the value of the dual in a finite number of steps as long as the set of parameters $\boldsymbol{\theta}^T$, $T \in \mathcal{T}$, do not satisfy WTA (see [46] for details).

Property 4: Unlike TRW-S, WTA is not the sufficient condition for convergence.

One of the main drawbacks of the TRW-S algorithm is that it converges as soon as the WTA condition is satisfied. Experiments in [46] indicate that this results in high energy solutions for the MAP estimation problem when the energy function is non-submodular. Using a counter-example, it can be shown that WTA is not the sufficient condition for the convergence of the TRW-S(LP-C) algorithm. One such example is given below.

Example: Fig. 7.1 shows an example of a set of three tree structured CRFs. Together, these CRFs form a frustrated cycle C (i.e. a cycle with odd number of non-submodular pairwise potentials) where $\mathbf{v}_C = \{v_a, v_b, v_c\}$ and $\mathcal{E}_C = \{(a,b),(b,c),(c,a)\}$. As noted earlier, the CRFs in Fig. 7.1 satisfy the WTA condition. The TRW-S algorithm converges at this point. Assuming $\rho(T) = 1/3$ for all $T \in \mathcal{T}$, the value of the dual (7.2.15) is 3. In contrast, the TRW-S(LP-C) algorithm increases the value of the dual (7.3.5) to 4 when the problem (7.3.14) is solved for the cycle C (i.e. WTA is not the sufficient condition for convergence).

Obtaining the Labelling: Similar to the TRW-S algorithm, TRW-S(LP-C) solves the dual (7.3.5) and not the primal problem. In other words, it does not directly provide a labelling of the random variables. In order to obtain a labelling, we use the same scheme as the one suggested in [46] for the TRW-S algorithm. Briefly, we assign labels to the variables $\mathbf{v} = \{v_0, v_1, \cdots, v_{n-1}\}$ in increasing order (i.e. we label variable v_0, followed by variable v_1 and so on). Let $\boldsymbol{\theta}^{\mathcal{T}} = \sum_T \rho(T)\boldsymbol{\theta}^T$. At each stage, a variable v_a is assign the label $l_{f(a)}$ such that

$$f(a) = \arg\min_{i, l_i \in \mathbf{l}} \left(\theta^{\mathcal{T}1}_{a;i} + \sum_{b<a,(a,b)\in\mathcal{E}} \theta^{\mathcal{T}2}_{ab;i,f(b))} \right), \tag{7.3.16}$$

where $\theta^{\mathcal{T}1}_{a;i}$ and $\theta^{\mathcal{T}2}_{ab;i,f(b)}$ are the unary and pairwise potentials corresponding to the parameter $\boldsymbol{\theta}^{\mathcal{T}}$ respectively. It can be shown that under certain conditions the

above procedure provides the optimal labelling [63][1].

7.4. Adding Second Order Cone Constraints

We now describe how second order cone (SOC) constraints can be added to the dual (7.2.15). Specifically, we consider the two SOC constraints proposed in the previous chapter which result in the SOCP-C and SOCP-Q relaxations (see § 6.5.1 and § 6.5.2 respectively). While SOCP-C adds constraints using random variables which form a cycle, SOCP-Q introduces SOC constraints using random variables connected in a clique.

In general, a set of N_C SOC constraints on a cycle/clique can be defined as

$$||\mathbf{A}_k^C\mathbf{y} + \mathbf{b}_k^C|| \leq \mathbf{y}^\top\mathbf{c}_k^C + d_k^C, k \in \{1, 2, \cdots, N_C\}. \tag{7.4.1}$$

Again, this is achieved by converting variables $(\mathbf{x}, \mathbf{X})$ to $\mathbf{y}$ using equation (7.2.9). Note that, similar to cycle inequalities, we can define upto $|\mathbf{l}|^c$ SOC constraints for a cycle/clique, where c is the size of the cycle/clique (i.e. $N_C \in \{0, 1, \cdots, |\mathbf{l}|^c\}$). Let $\mathcal{C}$ be a set of cycles/cliques in the graphical model of the given random field. The following theorem provides us with the dual of the SOCP relaxation obtained by appending problem (7.2.11) with SOC constraints defined over the set $\mathcal{C}$.

Theorem 2: The following problem is the dual of problem (7.2.11) appended with a set of SOC constraints $||\mathbf{A}_k^C\mathbf{y} + \mathbf{b}_k^C|| \leq \mathbf{y}^\top\mathbf{c}_k^C + d_k^C$ for $k \in \{1, 2, \cdots, N_C\}$ and $C \in \mathcal{C}$.

$$\begin{aligned}
\max \quad & \textstyle\sum_T \rho(T) q(\boldsymbol{\theta}^T) - \sum_C \rho'(C) \sum_k ((\mathbf{b}_k^C)^\top \mathbf{u}_k^C + d_k^C v_k^C), \\
\text{s.t.} \quad & \textstyle\sum_T \rho(T)\boldsymbol{\theta}^T + \sum_C \rho'(C) \sum_k ((\mathbf{A}_k^C)^\top \mathbf{u}_k^C + \mathbf{c}_k^C v_k^C) \equiv \boldsymbol{\theta}, \\
& ||\mathbf{u}_k^C|| \leq v_k^C, \forall k \in \{1, 2, \cdots, N_C\}, C \in \mathcal{C}.
\end{aligned} \tag{7.4.2}$$

Here $\mathbf{u}_k^C$ and v_k^C are some slack variables. Recall that $\boldsymbol{\rho}' = \{\rho'(C), C \in \mathcal{C}\}$ is a (fixed) set of non-negative real numbers which sum to one.

Proof: We begin by formulating the dual of the following problem:

$$\begin{aligned}
\max \quad & \textstyle\sum_T \rho(T) q(\boldsymbol{\theta}^T) - \sum_C \rho'(C) \sum_k ((\mathbf{b}_k^C)^\top \mathbf{u}_k^C + d_k^C v_k^C), \\
\text{s.t.} \quad & \textstyle\sum_T \rho(T)\boldsymbol{\theta}^T + \sum_C \rho'(C) \sum_k ((\mathbf{A}_k^C)^\top \mathbf{u}_k^C + \mathbf{c}_k^C v_k^C) = \boldsymbol{\theta}, \\
& ||u_k^C|| \leq v_k^C, \forall k \in \{1, 2, \cdots, N_C\}, C \in \mathcal{C}.
\end{aligned} \tag{7.4.3}$$

Using the definition of dual (see section 3.2), this can be written as

$$\min_{\boldsymbol{\tau}, \boldsymbol{\mu}} \left(\max_{\boldsymbol{\theta}^\top, \mathbf{u}_k^C, v_k^C} \sum_T \rho(T)(q(\boldsymbol{\theta}^T) - \boldsymbol{\tau}^\top\boldsymbol{\theta}^T) - \sum_C \rho'(C) \sum_k \mathcal{D}_k^C + \boldsymbol{\tau}^\top\boldsymbol{\theta} \right), \tag{7.4.4}$$

[1] If the set of variables that have more than one optimum labelling in any of the tree structured random fields $\boldsymbol{\theta}^T$, $T \in \mathcal{T}$, (satisfying WTA) form disjoint, monotonic chains with respect to the ordering of the variables (i.e. $v_0 < v_1 < \cdots, v_{n-1}$), then this procedure is guaranteed to provide the exact MAP estimate.

where

$$\mathcal{D}_k^C = v_k^C(\boldsymbol{\tau}^\top \mathbf{c}_k^C + d_k^C - \mu_k^C) + \boldsymbol{\tau}^\top (\mathbf{A}_k^C)^\top \mathbf{u}_k^C + (\mathbf{b}_k^C)^\top \mathbf{u}_k^C + \mu_k^C ||\mathbf{u}_k^C||. \tag{7.4.5}$$

Using equation (7.2.18), problem (7.4.4) can be simplified as

$$\min_{\boldsymbol{\tau} \in LOCAL(\mathbf{v},\mathcal{E}), \boldsymbol{\mu}} \max_{\boldsymbol{\theta}^\top, \mathbf{u}_k^C, v_k^C} -\sum_C \rho'(C) \sum_k \mathcal{D}_k^C + \boldsymbol{\tau}^\top \boldsymbol{\theta}, \tag{7.4.6}$$

From equation (7.4.5) we observe that $\mathcal{D}_k^C$ is bounded if, and only if, $\mu_k^C = \boldsymbol{\tau}^\top \mathbf{c}_k^C + d_k^C$. This can be easily verified by assuming $\mu_k^C \neq \boldsymbol{\tau}^\top \mathbf{c}_k^C + d_k^C$, in which case v_k^C will take one of the following values,

$$v_k^C = \begin{cases} -\infty & \text{if} \quad \mu_k^C < \boldsymbol{\tau}^\top \mathbf{c}_k^C + d_k^C, \\ \infty & \text{otherwise,} \end{cases} \tag{7.4.7}$$

thereby making $\mathcal{D}_k^C$ unbounded. Substituting the value of μ_k^C in equation (7.4.5), we get

$$\mathcal{D}_k^C = \boldsymbol{\tau}^\top (\mathbf{A}_k^C)^\top \mathbf{u}_k^C + (\mathbf{b}_k^C)^\top \mathbf{u}_k^C + (\boldsymbol{\tau}^\top \mathbf{c}_k^C + d_k^C) ||\mathbf{u}_k^C||. \tag{7.4.8}$$

Before proceeding further, we require a lemma.

Lemma 4.1: The following expression holds true:

$$\min_{\mathbf{a},d} \max_{\mathbf{z}} \mathbf{a}^T \mathbf{z} - d||\mathbf{z}|| = \begin{cases} 0 & \text{if} \quad ||\mathbf{a}|| \leq d, \\ \infty & \text{otherwise.} \end{cases} \tag{7.4.9}$$

Proof : Case (i): Let $||\mathbf{a}|| \leq d$. Thus,

$$\begin{aligned} d^2||\mathbf{z}||^2 &\geq ||\mathbf{a}||^2||\mathbf{z}||^2, & (7.4.10)\\ &= (\mathbf{a}^\top \mathbf{z})^2 + |\mathbf{a} \times \mathbf{z}|^2, & (7.4.11)\\ &\geq (\mathbf{a}^\top \mathbf{z})^2, & (7.4.12)\\ \Rightarrow d||\mathbf{z}|| &\geq \mathbf{a}^\top \mathbf{z}. & (7.4.13) \end{aligned}$$

Here $|\mathbf{a} \times \mathbf{z}|$ represents the cross-product of vectors $\mathbf{a}$ and $\mathbf{z}$. By making $\mathbf{a}$ and $\mathbf{z}$ parallel (i.e. $|\mathbf{a} \times \mathbf{z}| = 0$) , we see that the optimal value of the problem (7.4.9) is 0.

Case(ii): Let $||\mathbf{a}|| > d$. Thus,

$$\begin{aligned} (\mathbf{a}^\top \mathbf{z})^2 &= ||\mathbf{a}||^2||\mathbf{z}||^2 - |\mathbf{a} \times \mathbf{z}|^2, & (7.4.14)\\ &= ||\mathbf{a}||^2||\mathbf{z}||^2, \quad \text{when } \mathbf{a} \text{ is parallel to } \mathbf{z}, & (7.4.15)\\ &> d^2||\mathbf{z}||^2, & (7.4.16)\\ \Rightarrow (\mathbf{a}^\top \mathbf{z}) &> d||\mathbf{z}||. & (7.4.17) \end{aligned}$$

Thus the optimal value of problem (7.4.9) is ∞ (when $\mathbf{a}$ is parallel to $\mathbf{z}$). ∎

Applying the above lemma, we obtain the following equation:

$$\min_{\boldsymbol{\tau}} \max_{\mathbf{u}_k^C} -\mathcal{D}_k^C = \begin{cases} 0 & \text{if} \quad ||\mathbf{A}_k^C \boldsymbol{\tau} + \mathbf{b}_k^C|| \leq \boldsymbol{\tau}^\top \mathbf{c}_k^C + d_k^C, \\ \infty & \text{otherwise.} \end{cases} \tag{7.4.18}$$

Substituting the above equation in problem (7.4.6), we get a reformulation of the dual (7.4.3) as

$$\begin{aligned} & \min_{\boldsymbol{\tau}} \boldsymbol{\tau}^{\top}\boldsymbol{\theta} \\ \text{s.t.} \quad & \boldsymbol{\tau} \in LOCAL(\mathbf{v},\mathcal{E}), ||\mathbf{A}_k^C\boldsymbol{\tau} + \mathbf{b}_k^C|| \leq \boldsymbol{\tau}^{\top}\mathbf{c}_k^C + d_k^C, \forall C, k, \end{aligned} \tag{7.4.19}$$

which is the same as the SOCP-C or SOCP-Q relaxation (depending upon the choice of the SOC constraints). Thus problem (7.4.3) is the dual of the SOCP-C/SOCP-Q relaxation. Again, using the fact that the constraint set includes all the constraints of the LP-S relaxation, problem (7.4.3) can be shown to be equivalent to problem (7.4.2) [46]. This proves Theorem 2. ∎

Initialization
1. Choose a set of tree structured random fields $\mathcal{T}$ (similar to TRW-S). Choose a set of cycles/cliques $\mathcal{C}$. For example, if the 8-neighbourhood is employed, we can choose all cliques of size 4 or all cycles of size 3.
2. Initialize $\boldsymbol{\theta}^T$ such that $\sum_T \rho(T)\boldsymbol{\theta}^T \equiv \boldsymbol{\theta}$. Initialize $\mathbf{u}_k^C = v_k^C = 0$.
Iterative Steps
3. Pick an element $\omega \in \mathbf{v} \bigcup \mathcal{C}$. Find all cycles/cliques $\mathcal{C}_\omega \subseteq \mathcal{C}$ which contains ω.
4. For a cycle/clique $C \in \mathcal{C}_\omega$, compute $\boldsymbol{\theta}^C = \sum_{T\in C} \rho(T)\boldsymbol{\theta}^T + \rho'(C)\sum_k((\mathbf{A}_k^C)^{\top}\mathbf{u}_k^C + \mathbf{c}_k^C v_k^C)$ using the values of $\boldsymbol{\theta}^T$, $\mathbf{u}_k^C$ and v_k^C obtained in the previous iteration. Solve problem (7.4.20) using an Interior Point method. Update the values of $\boldsymbol{\theta}^T$, $\mathbf{u}_k^C$ and v_k^C.
5. For all trees $T \in \mathcal{T}$ which contain ω, run steps 4 and 5 of the TRW-S algorithm.
6. Repeat steps 3 and 4 for all cycles /cliques $C \in \mathcal{C}_\omega$.
7. Repeat steps 3 to 5 for all elements ω till convergence.

Table 7.3: *The* TRW-S*(*SOCP-C*)/*TRW-S*(*SOCP-Q*) algorithm.*

Recall that for SOCP-C and SOCP-Q relaxations, we specify one SOC constraint per cycle/clique (i.e. $N_C = 1$, see § 6.5.1 and § 6.5.2). This SOC constraint is defined using variables which take a trivial value in the LP-S relaxation when the energy function is non-submodular. Before proceeding further, we also define the following problem:

$$\begin{aligned} \max \quad & \textstyle\sum_{T\in C} \rho(T) q(\boldsymbol{\theta}^T) - \rho'(C)\sum_k((\mathbf{b}_k^C)^{\top}\mathbf{u}_k^C + d_k^C v_k^C), \\ \text{s.t.} \quad & \textstyle\sum_{T\in C} \rho(T)\boldsymbol{\theta}^T + \rho'(C)\sum_k((\mathbf{A}_k^C)^{\top}\mathbf{u}_k^C + \mathbf{c}_k^C v_k^C) = \boldsymbol{\theta}^C, \\ & ||\mathbf{u}_k^C|| \leq v_k^C, \forall k \in \{1, 2, \cdots, N_C\}, \end{aligned} \tag{7.4.20}$$

where $\boldsymbol{\theta}^C$ is some parameter vector. The above problem is the dual for the SOCP-C/SOCP-Q relaxation over one cycle/clique C with parameter $\boldsymbol{\theta}^C$ (with any

number of SOC constraints defined over it). Like problem (7.3.14), we can solve problem (7.4.20) using standard Interior Point algorithms for small cycles/cliques C.

Similar to TRW-S(LP-C), a convergent algorithm can now be described for solving the dual (7.4.2). This algorithm, outlined in table 7.3, differs from TRW-S(LP-C) in only step 4, where it solves problem (7.4.20) for a cycle/clique C instead of problem (7.3.14). Recall that the notation $T \in C$ in table 7.3 implies that, if $T = (\mathbf{v}_T, \mathcal{E}_T)$ and $C = (\mathbf{v}_C, \mathbf{E}_C)$, then there exists a pair of neighbouring random variables v_a and v_b such that $(a, b) \in \mathcal{E}_T$ and $(a, b) \in \mathcal{E}_C$.

We refer to this algorithm as either TRW-S(SOCP-C) or TRW-S(SOCP-Q) depending upon the SOCP relaxation that we are solving. When using the TRW-S(SOCP-Q) algorithm, we include all slack variables corresponding to the cycle inequalities defined over the cycles in clique C. It can easily be shown that both TRW-S(SOCP-C) and TRW-S(SOCP-Q) satisfy all the properties given in § 7.3.1. Note that, like TRW-S and TRW-S(LP-C), these algorithms do not directly provide a labelling for the random variables of the CRF. Instead we use the procedure described in § 7.3.1 to obtain the final solution.

7.5. Experiments

We tested the approaches described in this chapter using both synthetic and real data. As will be seen, the results conform with the analysis of the previous chapter. Specifically, we show that (i) SOCP-C outperforms the LP-S relaxation; (ii) the LP-C relaxation provide a better approximation of the MAP estimation problem than SOCP-C; and (iii) SOCP-Q (when applicable) obtains better labellings than LP-C.

7.5.1 Synthetic Data

Datasets: We conducted two sets of experiments using binary grid CRFs (i.e. $h = |\mathbf{l}| = 2$) of size 30×30. In the first experiment the edges of the graphical model, i.e. $\mathcal{E}$, were defined using a 4-neighbourhood system while the second experiment used an 8-neighbourhood system. Note that both these neighbourhood systems are commonly used in many Computer Vision applications [89]. The unary potentials $\theta^1_{a;0}$ and $\theta^1_{a;1}$ were generated using the normal distribution $\mathcal{N}(0, 1)$. The pairwise potentials $\theta^2_{ab;00}$ and $\theta^2_{ab;11}$ were set to 0 while $\theta^2_{ab;01}$ and $\theta^2_{ab;10}$ were generated using $\mathcal{N}(0, \sigma^2)$. We used three values for σ: $\sigma \in \{\frac{2}{\sqrt{d}}, \frac{5}{\sqrt{d}}, \frac{10}{\sqrt{d}}\}$ where d is the degree of the variables in the graphical model (i.e. $d = 4$ in the first experiment and $d = 8$ in the second experiment). Note that when $\theta^2_{ab;01} + \theta^2_{ab;10} < 0$ then the energy function is non-submodular. For each value of σ, 50 CRFs were generated using the method described above (i.e. each experiment was conducted

using 150 CRFs).

Implementation Details: We tested the LP-C and the SOCP-C relaxations in the first experiment. Constraints were defined on all cycles of size 4. The LP-C and SOCP-Q relaxation were tested in the second experiment. Cycles inequalities were defined on all cycles of size 3. In addition for SOCP-Q, SOC constraints were defined on all cliques of size 4. In both the experiments, the trees were chosen as the individual edges of the graphical model. The terms $\rho(T)$ and $\rho'(C)$ were set to $1/|\mathcal{T}|$ and $1/|\mathcal{C}|$ respectively for all $T \in \mathcal{T}$ and $C \in \mathcal{C}$. We found it sufficient to define one cycle inequality per cycle C using a set $\{l_{i_1}, l_{i_2}, \cdots, l_{i_c}\} \in \mathbf{l}^c$ which satisfies

$$\sum_{(a_k,a_l)\in\mathcal{E}_F} \theta_{a_k a_l;i_k i_l} - \sum_{(a_k,a_l)\in\mathcal{E}_C-\mathcal{E}_F} \theta^2_{a_k a_l;i_k i_l} \geq \sum_{(a_k,a_l)\in\mathcal{E}_F} \theta_{a_k a_l;j_k j_l} - \sum_{(a_k,a_l)\in\mathcal{E}_C-\mathcal{E}_F} \theta^2_{a_k a_l;j_k j_l}, \tag{7.5.1}$$

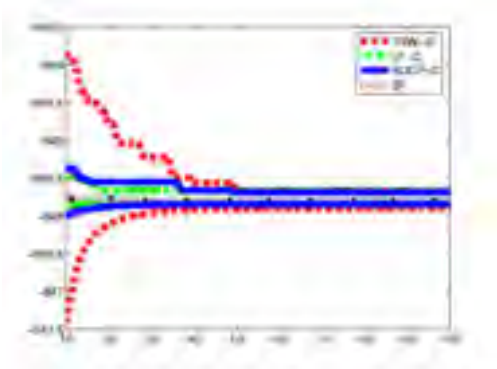
for all $\{l_{j_1}, \cdots, l_{j_c}\} \in \mathbf{l}^c$. Here $\mathcal{E}_C = \{(a_1, a_2), \cdots, (a_n, a_1)\}$ and $\mathcal{E}_F \subseteq \mathcal{E}_C$ such that $|\mathcal{E}_F| = 3$. As mentioned earlier, we define one SOC constraint per cycle/clique when considering the SOCP-C and the SOCP-Q relaxations (see § 6.5.1 and § 6.5.2). At each iteration, problems (7.3.14) and (7.4.20) were solved using the MOSEK software (available at http://www.mosek.com).

Results: Figure 7.2 shows the results obtained for the first experiment using the three values of σ. Note that since the energy functions are non-submodular, TRW-S provides labellings with higher energies than min-sum BP. However, the additional constraints in the LP-C and SOCP-C algorithm enable us to obtain labelling with lower energies than min-sum BP. Further, unlike min-sum BP, they also provide us with the value of the dual at each iteration. This value allows us to find out how close we are to the global optimum (since the energy of the optimal labelling cannot be less than the value of the dual). Also note that the value of the LP-C dual is greater than the value of the SOCP-C dual. This indicates that LP-C provides a better approximation for the MAP estimation problem.

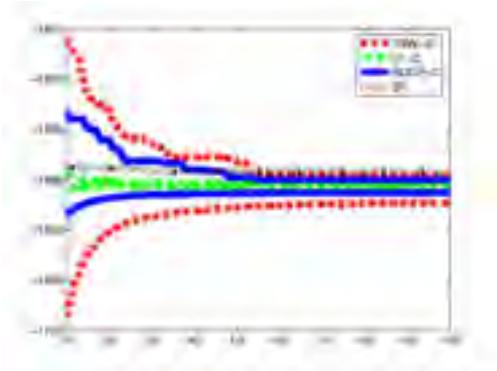
The results of the second experiment are shown in Figure 7.3. Again, min-sum BP outperforms TRW-S, while LP-C and SOCP-Q provide better approximations. The SOC constraints defined over cliques in the SOCP-Q relaxation provide a greater value of the dual compared to the LP-C relaxation. The complexity and timings for all the algorithms is given in tables 7.4 and 7.5.

7.5.2 Real Data - Video Segmentation

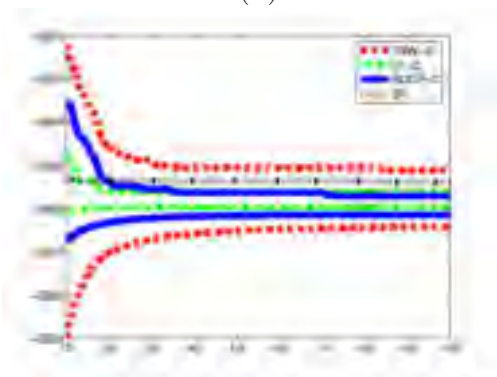
We now present the results of our method on the problem of video segmentation which can be described as follows. Given some seed pixels for all the segments present in a video, we would like to obtain the segmentation of all the frames.



(a)



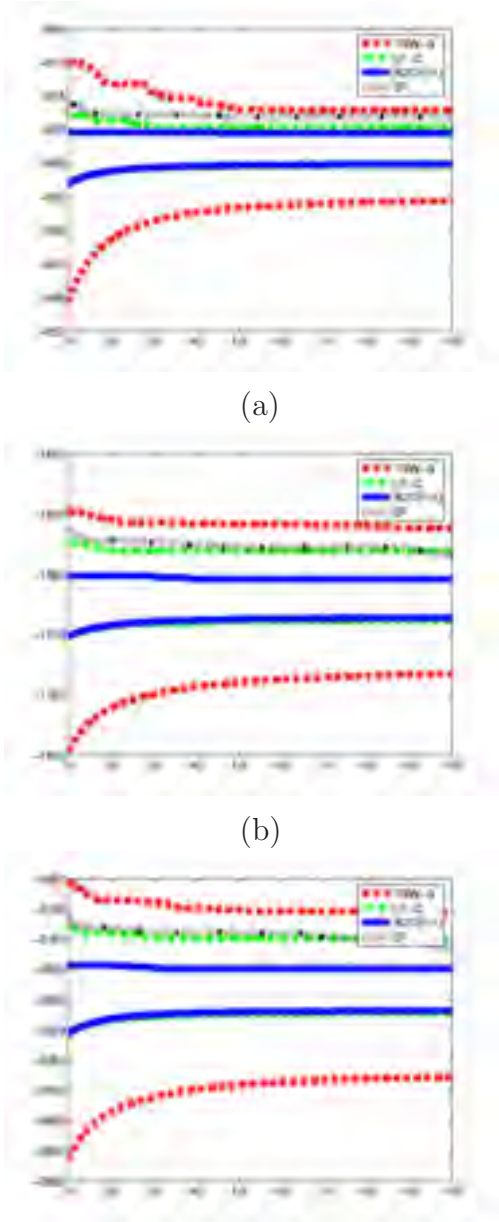
(b)



(c)

Figure 7.2: *Results of the first experiment.* **(a)** $\sigma = 2$. **(b)** $\sigma = 5$. **(c)** $\sigma = 10$. *The x-axis shows the iteration number. The lower curves show the average value of the dual at each iteration over 50 random* CRF*s while the upper curves show the average energy of the best labelling found till that iteration. The additional constraints in the* LP-C *and* SOCP-C *relaxations enable us to obtain labellings with lower energy (or equivalently higher posterior probability) compared to* TRW-S *and min-sum* BP*. Cycle inequalities provide a better approximation than the* SOC *constraint of the* SOCP-C *relaxation.*

For this work, we will describe a method which segments each frame individually *without making any use of the motion information.*



(a)

(b)

(c)

Figure 7.3: *Results of the second experiment.* **(a)** $\sigma = 2$. **(b)** $\sigma = 5$. **(c)** $\sigma = 10$. *Note that the value of the dual obtained using* SOCP-Q *is greater than the value of the dual of the* LP-C *relaxation.*

Problem Formulation: The problem of obtaining the segmentation of a frame can be cast within the CRF framework (similar to the interactive binary image segmentation CRF described in § 2.2.2). Specifically, we define a CRF over random variables $\mathbf{v} = \{v_0, \cdots, v_{n-1}\}$, where each variable corresponds to a pixel of the frame. Each label in the set $\mathbf{l} = \{l_0, \cdots, l_{h-1}\}$ corresponds to a segment (where h is the total number of segments). The unary potential of assigning a variable v_a to segment l_i is specified by the negative log-likelihood of the RGB value of pixel a, i.e. $\mathbf{D}_a$, given the seed pixels of the segment l_i. In our experiments, the unary

Algorithm	No. of Var.	No. of Cons.	Time(sec)
min-sum BP	-	-	0.0018
TRW-S	$nh + \|\mathcal{E}\|h^2$	$n + 2\|\mathcal{E}\|h$	0.0018
LP-C	$nh + \|\mathcal{E}\|h^2$	$2n + 2\|\mathcal{E}\|h$	7.5222
SOCP-C	$nh + \|\mathcal{E}\|h^2$	$2n + 2\|\mathcal{E}\|h$	8.9091

Table 7.4: *Complexity and timings of the algorithms for the first synthetic data experiment with a 4-neighbourhood relationship. Recall that $n = |\mathbf{v}|$ is the number of random variables, $h = |\mathbf{l}|$ is the size of the label set and $\mathcal{E}$ is the neighbourhood relationship defined by the* CRF*. The second and third columns show the number of variables and constraints in the primal problem respectively. The fourth column shows the average time of the each algorithm for one iteration (in seconds). All timings are reported for a Pentium IV 3.3 GHz processor with 2GB RAM.*

Algorithm	No. of Var.	No. of Cons.	Time(sec)
min-sum BP	-	-	0.0027
TRW-S	$nh + \|\mathcal{E}\|h^2$	$n + 2\|\mathcal{E}\|h$	0.0027
LP-C	$nh + \|\mathcal{E}\|h^2$	$5n + 2\|\mathcal{E}\|h$	7.7778
SOCP-Q	$nh + \|\mathcal{E}\|h^2$	$6n + 2\|\mathcal{E}\|h$	9.1170

Table 7.5: *Complexity and timings of the algorithms for the second synthetic data experiment with an 8-neighbourhood relationship.*

potential takes the form

$$\theta^1_{a;i} = -\log \Pr(\mathbf{D}_a|\mathcal{H}_i), \tag{7.5.2}$$

where $\mathcal{H}_i$ is the appearance model of l_i which is represented as a histogram of RGB values of the seed pixels of l_i (with 15 bins each for R, G and B). The pairwise potentials encourage continuous segments whose boundaries lie on image edges. Specifically, they take a form similar to equation (2.2.16) (for interactive binary image segmentation), i.e.

$$\theta^2_{ab;ij} = \begin{cases} \kappa_1 & \text{if} \quad i = j, \\ \kappa_2 - \gamma(a,b) & \text{otherwise,} \end{cases} \tag{7.5.3}$$

where

$$\gamma(a,b) = \lambda \left(1 - \exp\left(\frac{-\Delta^2(a,b)}{2\sigma^2}\right) \frac{1}{dist(a,b)}\right). \tag{7.5.4}$$

Recall that the term $\Delta(a,b)$ measures the difference in the RGB values $\mathbf{D}_a$ and $\mathbf{D}_b$ and $dist(a,b)$ is the Euclidean distance between pixels a and b. We use the following weight values for all our experiments: $\kappa_1 = 1$, $\kappa_2 = 2.2$, $\lambda = 1$ and $\sigma = 5$.

The problem of obtaining the segmentation of a frame then boils down to that of finding the MAP estimate of the CRF. Note that the energy function for the

above CRF, i.e.

$$Q(f;\mathbf{D},\boldsymbol{\theta}) = \sum_{v_a \in \mathbf{v}} \theta^1_{a;f(a)} + \sum_{(a,b)\in\mathcal{E}} \theta^2_{ab;f(a)f(b)}, \tag{7.5.5}$$

is non-submodular (by the definition provided in § 2.4.3). Here, $\mathcal{E}$ is the neighbourhood relationship of the CRF. Typically, a 4 or 8-neighbourhood is used for similar applications in Computer Vision [13, 89].

Datasets and Implementation Details: We used two well-known sequences to conduct our experiments, namely the 'Garden' sequence (with frame size 120×175) and the 'Dayton' sequence (with frame size 128×192). For both sequences, the seed pixels were provided using the ground truth segmentation of a keyframe as shown in Fig. 7.4 and 7.5 respectively.

Similar to the synthetic data experiment, we defined the trees as individual edges of the graphical model of the CRF. In other words, a tree $T = (\mathbf{v}_T, \mathcal{E}_T) \in \mathcal{T}$ such that $\mathbf{v}_T = \{v_a, v_b\}$ and $\mathcal{E}_T = \{(a,b)\} \subseteq \mathcal{E}$. We specified one cycle inequality and one SOC constraint for each cycle/clique (as described in the previous section). The terms $\rho(T)$ and $\rho'(C)$ were set to $1/|\mathcal{T}|$ and $1/|\mathcal{C}|$ respectively for all $T \in \mathcal{T}$ and $C \in \mathcal{C}$. Once again, problems (7.3.14) and (7.4.20) were solved using MOSEK.

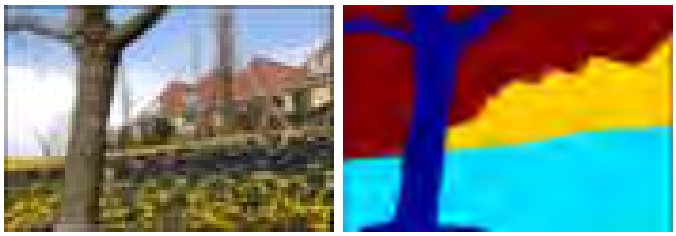


Figure 7.4: *Segmented keyframe of the 'Garden' sequence. The left image shows the keyframe while the right image shows the corresponding segmentation provided by the user. The four different colours indicate pixels belonging to the four segments namely sky, house, garden and tree.*

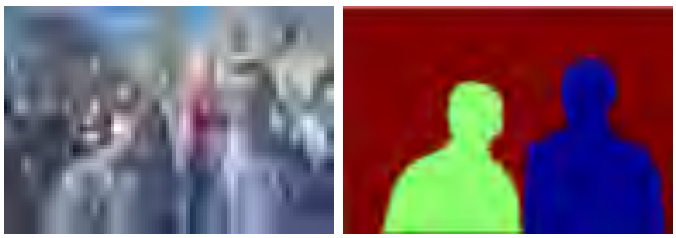


Figure 7.5: *The keyframe of the 'Dayton' video sequence partitioned into three segments.*

Algorithm	Sequence	Avg. Time(sec)
min-sum BP	Garden	0.1400
min-sum BP	Dayton	0.1540
TRW-S	Garden	0.1400
TRW-S	Dayton	0.1540
$\alpha\beta$-swap	Garden	0.1052
$\alpha\beta$-swap	Dayton	0.1154
α-expansion	Garden	0.1100
α-expansion	Dayton	0.1200
LP-C	Garden	140.3320
LP-C	Dayton	150.4440
SOCP-C	Garden	143.6365
SOCP-C	Dayton	158.1820

Table 7.6: *Average timings of the algorithms (per iteration) for the first experiment on video segmentation with a 4-neighbourhood relationship. Again, all timings are reported for a Pentium IV 3.3 GHz processor with 2GB RAM.*

Results: For the first set of experiments, we used a 4-neighbourhood system and tested the following algorithms: TRW-S, LP-C, SOCP-C, $\alpha\beta$-swap, α-expansion and min-sum BP. Fig. 7.6 and 7.7 show the segmentations and the values of the energy function obtained for all algorithms. Note that, by incorporating additional constraints using all cycles of length 4, LP-C and SOCP-C outperform other methods. Further, the cycle inequalities in LP-C provide better results than the SOC constraints of SOCP-C. Table 7.6 provides the average time taken per iteration by all the algorithms for both the video sequences.

The second set of experiments used an 8-neighbourhood system and tested the following algorithms: TRW-S, LP-C, SOCP-Q, $\alpha\beta$-swap, α-expansion and min-sum BP. For the LP-C algorithm, cycle inequalities were specified for all cycles of size 3. In addition, the SOCP-Q algorithm specifies SOC constraints on all cliques of size 4. Fig. 7.8 and 7.9 show the segmentations and energies obtained for all the algorithms. The average timings per iteration are shown in table 7.7. Note that, similar to the synthetic data examples, SOCP-Q outperforms LP-C by incorporating additional SOC constraints.

7.6. Discussion

We have extended the LP-S relaxation based approach of [46, 101] for the MAP estimation problem. Specifically, we showed how cycle inequalities and the SOC constraints of the previous chapter can be incorporated within the dual of the LP relaxation. We also proposed convergent algorithms for solving the resulting

Algorithm	Sequence	Avg. Time(sec)
min-sum BP	Garden	0.1740
min-sum BP	Dayton	0.1810
TRW-S	Garden	0.1740
TRW-S	Dayton	0.1810
$\alpha\beta$-swap	Garden	0.1201
$\alpha\beta$-swap	Dayton	0.1564
α-expansion	Garden	0.1240
α-expansion	Dayton	0.1600
LP-C	Garden	142.2226
LP-C	Dayton	155.5560
SOCP-Q	Garden	144.9890
SOCP-Q	Dayton	162.3400

Table 7.7: *Average timings of the algorithms (per iteration) for the second experiment on video segmentation with an 8-neighbourhood relationship.*

duals. Our experiments indicate that these additional constraints provide a more accurate approximation for MAP estimation when the energy function is non-submodular. Although our algorithm is much faster than Interior Point methods, it is slower than TRW-S and min-sum BP. An interesting direction for future research would be to develop specialized algorithms for solving problems (7.3.14) and (7.4.20) (which are required by the TRW-S(LP-C), TRW-S(SOCP-C) and TRW-S(SOCP-Q) algorithms). Further research is also required to find out how many cycle inequalities and SOC constraints need to be imposed per cycle/clique for MAP estimation problems with large number of labels (e.g. stereo and image denoising).

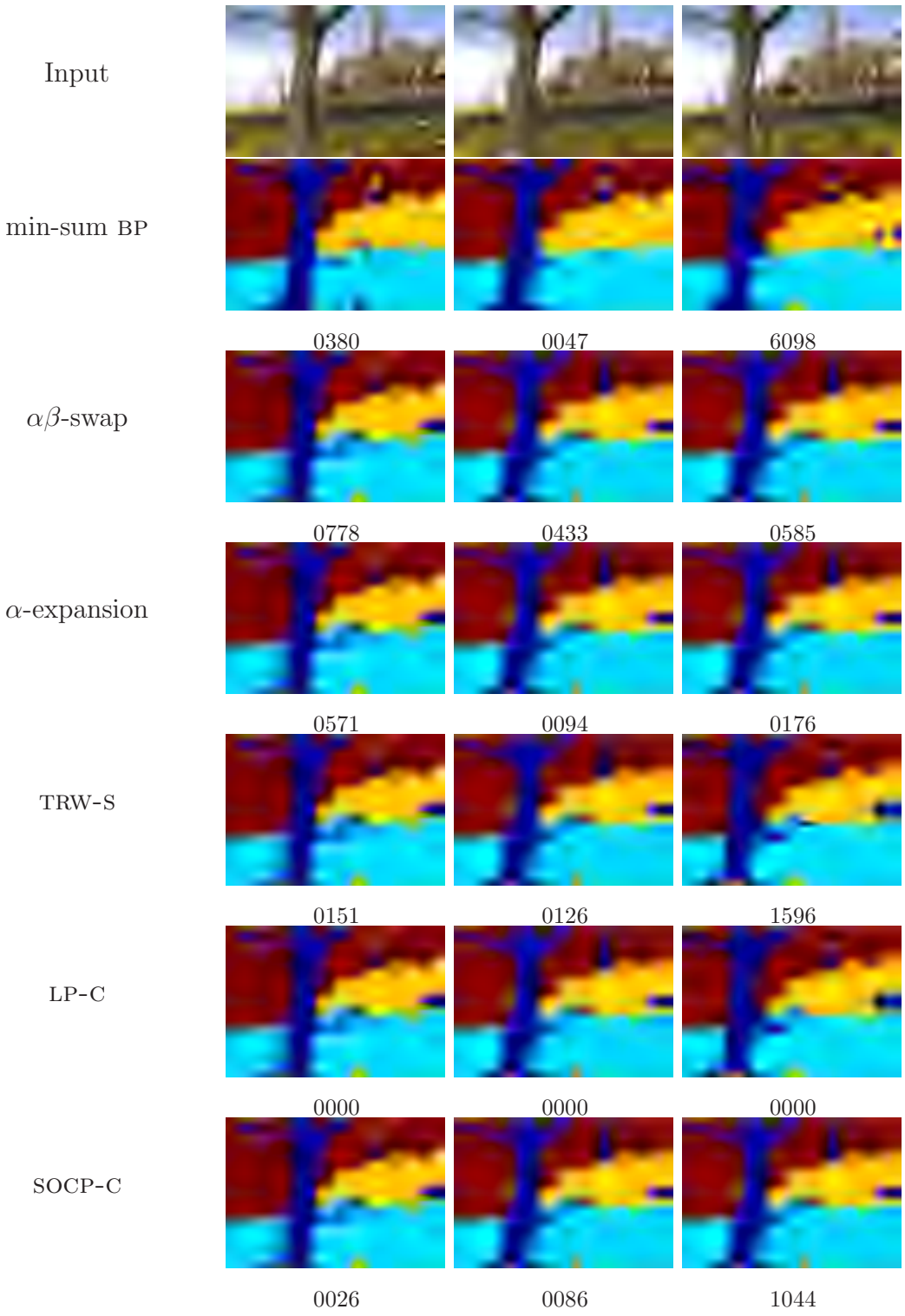


Figure 7.6: *Segmentations obtained for the 'Garden' video sequence using 4-neighbourhood. The corresponding energy values (scaled up to integers for using $\alpha\beta$-swap and α-expansion) of all the algorithms are shown below the segmentation. The following constant terms are subtracted from the energy values of all algorithms for the three frames respectively: 5139499, 5145234 and 5126941. These constant terms ensure that the minimum energy among all algorithms is 0.*

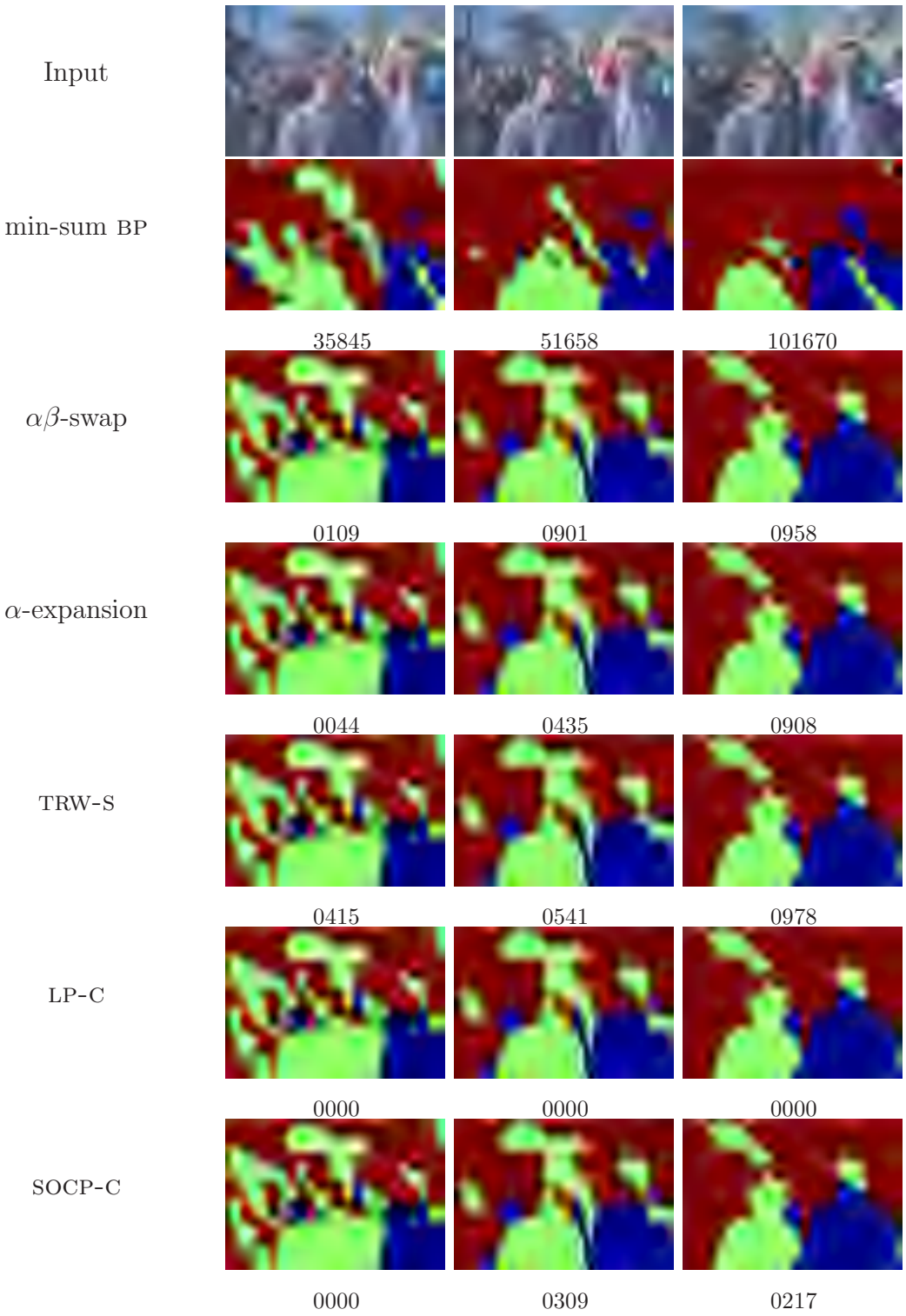


Figure 7.7: *Segmentations obtained for the 'Dayton' video sequence using 4-neighbourhood. The corresponding energy values (again scaled up to integers) of all the algorithms are shown below the segmentation. The following constant terms are subtracted from the energy value of all algorithms for the three frames respectively: 6920341, 6780685 and 6495282.*

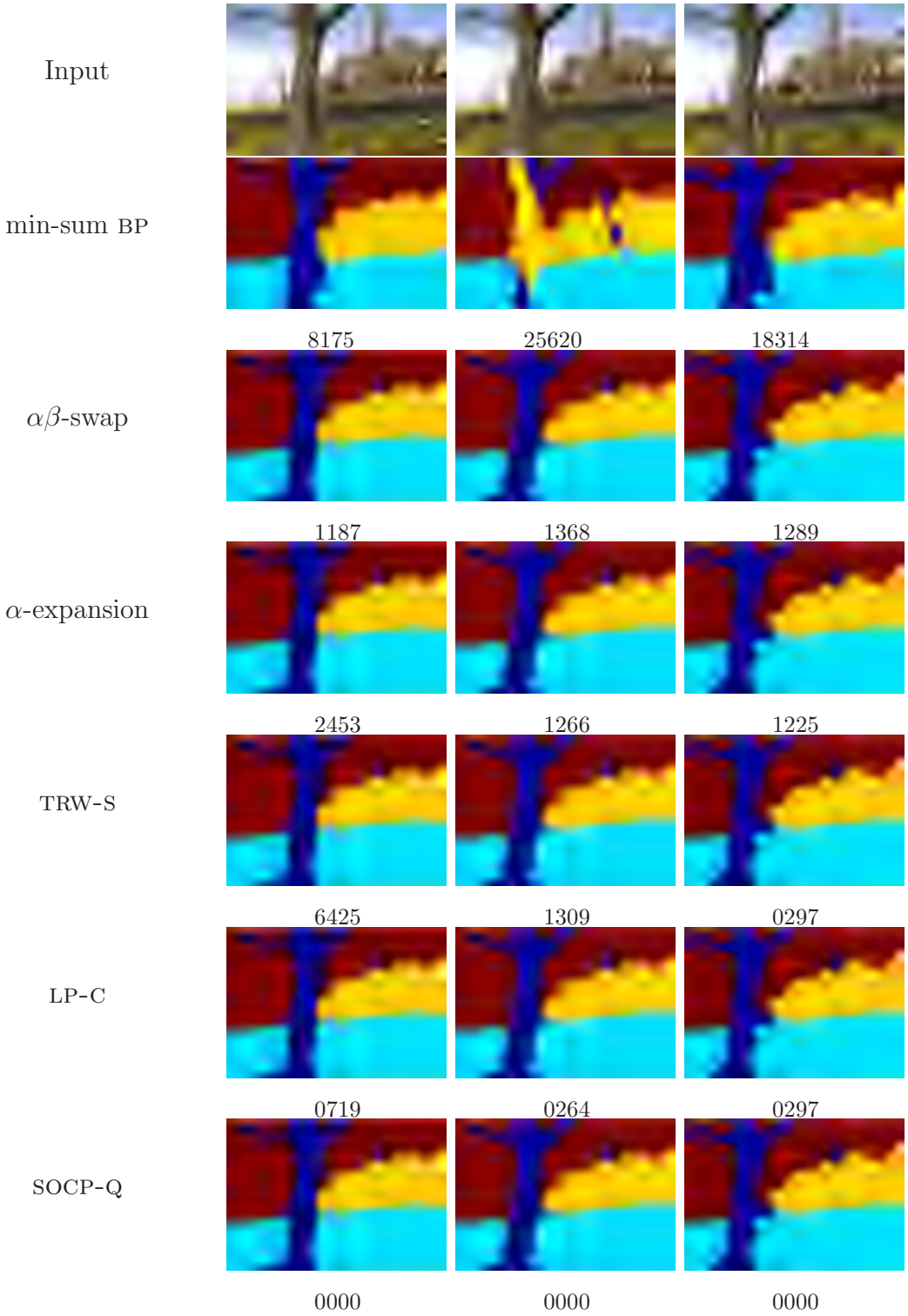


Figure 7.8: *Segmentations obtained for the 'Garden' video sequence using 8-neighbourhood. The corresponding energy values (reduced by 5304466, 5299756 and 5292224 for the three frames respectively) of all the algorithms are also shown.*

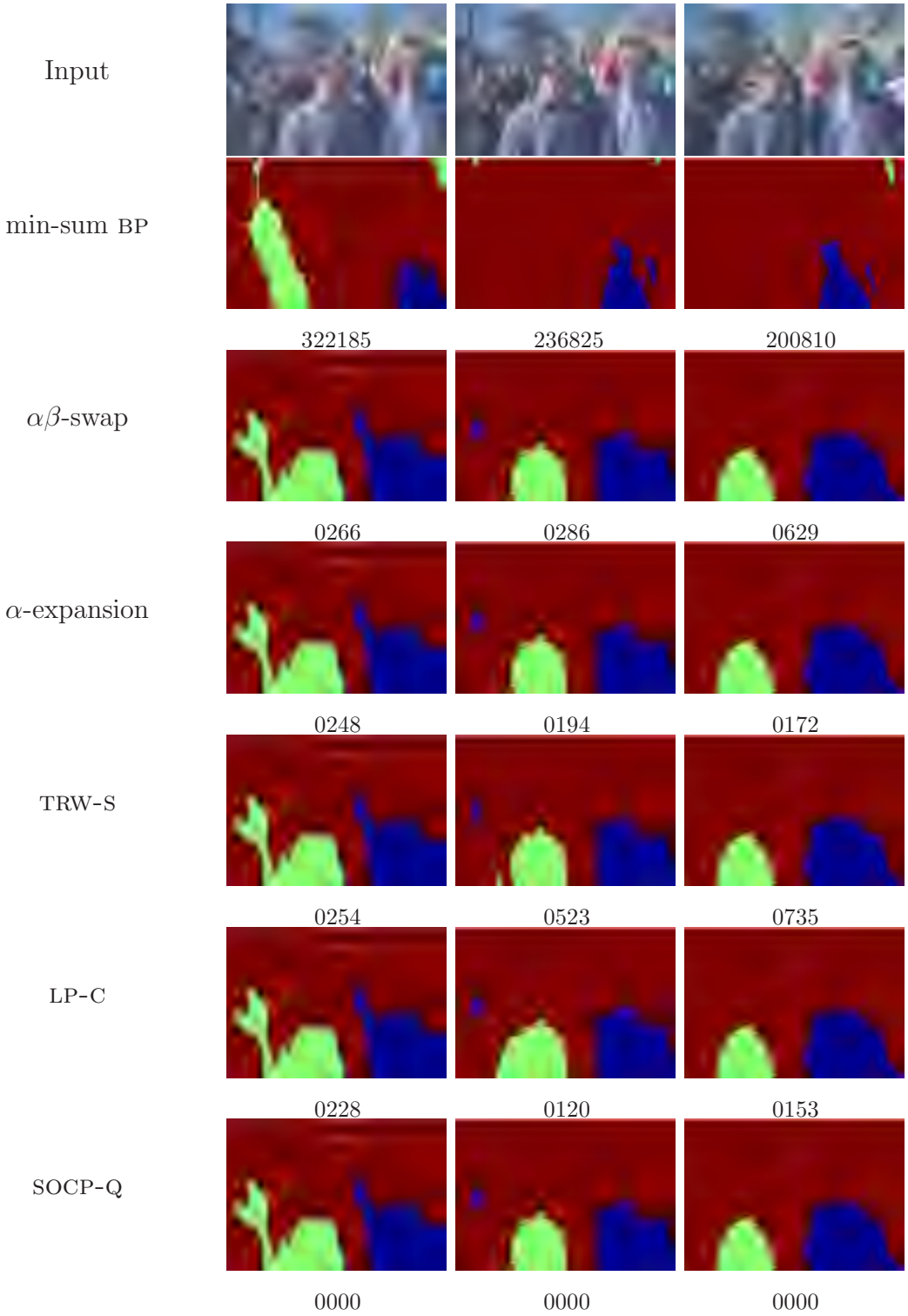


Figure 7.9: *Segmentations obtained for the 'Dayton' video sequence using 8-neighbourhood. The corresponding energy values of all the algorithms (reduced by 7127359, 7013665 and 6803425 for the three frames respectively) are also shown.*

Chapter 8

Discussion

8.1. Contributions of the Thesis

In this thesis, we examined and added to the growing role of optimization in Computer Vision and Machine Learning. The major contributions are:

- In chapter 4, we presented a novel layered representation model which is suitable for motion segmentation. Included in the model are the effects of occlusion, and changes in appearance due to lighting conditions and motion blur. We developed an approach for estimating the layered representation for a given video sequence. The initial estimate of the layered representation was obtained by developing an efficient coarse-to-fine sum-product BP algorithm. This algorithm is applicable for performing inference on any random field whose pairwise potentials describe a non-regular Potts model. A refinement of the initial estimate was obtained using the observation that the energy of the model can be minimized by a combination of $\alpha\beta$-swap and α-expansion algorithms.

- In chapter 5 we introduced a probabilistic model, called the Object Category Specific CDRF, which goes beyond the previously used pairwise CRFs for image segmentation. Specifically, our model includes the clique potentials of the pairwise CRFs (which represent bottom-up information), as well as strong shape potentials provided by an object category model (which represent top-down information). We also proposed a novel object category model for articulated objects (i.e. the LPS) which is suited for applications such as object category specific image segmentation. By employing the Object Category Specific CDRF, the segmentations of previously unseen images were obtained using two algorithmic contributions. Firstly, we showed how samples of our choice of object category models (i.e. SOE and LPS) can be found efficiently by matching them to the image. Secondly, we made the observation that the object category model can be quickly marginalized (using their samples) within the EM algorithm. Our observation is applicable for any latent variable (i.e. not just object category models) within any random field model (i.e. MRF or CRF), and can also be used for efficient generalized EM [30].

- In chapter 6, we presented a theoretical analysis of previously proposed convex relaxations for the MAP estimation problem [17, 51, 55, 64, 75, 79, 101]. Specifically, we showed that the LP-S relaxation [17, 51, 79, 101] strictly dominates the QP-RL [75] and SOCP-MS relaxations [55, 64]. We generalized this result by defining a large class of QP and SOCP relaxations which are dominated by LP-S. Based on this analysis, we proposed two new SOCP relaxations, namely SOCP-C and SOCP-Q. With the help of some examples,

we showed that SOCP-C dominates the previous relaxations considered in the analysis. We conjectured that a linear programming relaxation, which we call LP-C, dominates SOCP-C. However, using a numerical example, we showed that LP-C itself is dominated by the SOCP-Q relaxation. It is worth noting that, although our analysis was concerned with MAP estimation, it is equally applicable to the related theoretical Computer Science problems such as MAXCUT, multi-way cut, metric labelling, and 0-extension.

- In chapter 7, we extend the TRW framework of [46, 101] to efficiently solve a large class of convex relaxations for MAP estimation (including LP-C, SOCP-C and SOCP-Q). The resulting algorithm attempts to iteratively maximize the bounded dual of the relaxations. At each iteration, this algorithm does not decrease the value of the dual. In other words, it is guaranteed to converge. Our approach allows us to approximately solve the convex relaxations on very large problems where Interior Point methods would not be computationally feasible. Using both synthetic and real data, we showed that the empirical results conform with our theoretical analysis. In particular, SOCP-C dominates LP-S, LP-C dominates SOCP-C and SOCP-Q dominates LP-C.

8.2. Future Work

We now discuss some possible future directions which can extend the practical and theoretical implications of our work.

Motion Segmentation

- It is sometimes possible to obtain *a priori* information about which object categories are present in a given video sequence. For example, a video of a football match would mostly contain humans while a video of an urban traffic scene would contain vehicles (e.g. cars, bikes). The layered representation model developed in chapter 4 does not incorporate such information (which may greatly improve its performance). An interesting direction for future research would be to combine the strengths of the Object Category Specific CDRF (which can represent top-down information about the object categories) and the layered representation (which can handle occlusion, lighting changes and motion blur) to obtain a better motion segmentation algorithm for cases where *a priori* information is available.

- Given a video, we presented an approach which automatically learns the parameters of the layered representation. Our method is currently restricted to one visual aspect of the object. However, it is quite common for the visual aspect to change in the video sequence (e.g. a person may turn towards the

camera while walking). One possible solution to handle such scenarios is to determine where the visual aspect changes in the video and learn a new layered representation from that frame onwards. However, a more elegant approach would be to learn a 3D representation of the object. The work of Bray *et al.* [16] is a significant step towards achieving this goal. However, their method relies on multiple views of the same scene which may not always be available. Further, they consider the energy associated with only one frame at a time (in contrast to the layered representation which sums up the energy over the entire video). Sigal *et al.* [87] also propose a method to learn the probability distributions of the relative positions of the limbs corresponding to a 3D representation of a human. However, their method does not learn the shape and appearance of the limbs automatically. Further research is required to overcome these deficiencies.

- We proposed a method for combining the rigidly moving components, computed for every pair of consecutive frames, in order to obtain the initial estimate of the number of segments (together with their shape, appearance and transformations). However, our method is susceptible to error in the presence of a large amount of noise in a few frames of the video. Consider one such noisy pair of frames which may result in erroneous components (e.g. the component corresponding to the head of a person might be split into two components). Each erroneous component can result in estimating a wrong segment. Although we can hope to prune out the wrong segments during the refinement stages (using $\alpha\beta$-swap and α-expansion algorithms), a better solution would be to robustly estimate the number of segments in an iterative fashion (similar to the work described in [73]).

Object Category Specific Image Segmentation

- We presented a method, ObjCut, in order to estimate the Object Category Specific CDRF for image segmentation. Although our method is applicable for any object category model, its efficiency depends heavily on quickly obtaining samples of the object category model. For this reason, we restricted ourselves to segmenting only sideways or near-sideways views of objects such as cows and horses. An interesting direction for future research would be to develop algorithms for efficiently matching SOE and LPS models which represent multiple visual aspects of an object category.

- The ObjCut algorithm goes beyond conventional image segmentation approaches by incorporating top-down information about the object category (in the form of strong shape potentials). However, it uses several general low-level Vision approaches, e.g. edge detection in order to compute Chamfer distance for matching object category models to images. Recent developments in Computer Vision show that the matching can be improved by

considering only object category specific edges [71, 83]. Results in [71] indicate that this leads to substantial improvements in the segmentation itself. The efficiency of ObjCut can also be considerably increased by pruning away those areas of the image which are not likely to contain an instance of an object category (e.g. using low level cues such as colour, similar to [88]). Such approaches need further investigation.

- The various parameters which define the energy of the Object Category Specific CDRF (e.g. κ_1, κ_2, λ and σ) have been set up manually. In our experiments, we showed that our choice of parameters provides accurate segmentation for images of several object categories. However, the recent trend in image segmentation is to learn these parameter values from a set of segmented training images [3, 59, 76]. Parallel work in object category recognition has made available large number of pre-segmented images such as MSRC, Sowerby and Pascal VOC 2007 datasets. Using these datasets, it would be interesting to develop parameter learning methods for the Object Category Specific CDRF model and study their effects on the segmentation accuracy.

Analysis of Convex Relaxations

- A direct consequence of our analysis was two new SOCP relaxations, namely SOCP-C and SOCP-Q. We showed that these relaxations provide better solutions to the MAP estimation problem compared to a number of previously proposed approaches. An interesting direction for future research would be to devise strategies for obtaining other useful constraints, e.g. linear constraints on random variables connected in a clique, which define a smaller feasibility region than SOCP-Q.

- The class of relaxations considered in our analysis relaxed the variables $X_{ab;ij}$, where $(a, b) \notin \mathcal{E}$ (i.e. for non-neighbouring random variables v_a and v_b), to simply lie in the interval $[-1, 1]$. It is well known that the SDP relaxation which constrains all the elements of the matrix $\mathbf{X}$ (by specifying $\mathbf{X} \succeq 0$), provides an accurate approximation for MAP estimation. However, the increase in the number of constrained variables in the SDP relaxation makes it computationally infeasible. A compromise between accuracy and efficiency could be reached by constraining only a subset of variables $X_{ab;ij}$, where $(a, b) \notin \mathcal{E}$. Further research is required to determine which of the above variables should be constrained and which can be left unconstrained without severely effecting the accuracy of the resulting relaxation.

- We conjectured that the LP-C relaxation (which includes all the constraints of the LP-S relaxation together with linear cycle inequalities) dominates the SOCP-C relaxation. Our experiments in chapter 7 support this conjecture. These empirical results provide enough encouragement to attempt a

more theoretical approach towards comparing LP-C and SOCP-C. Similar to the analysis in chapter 6, this may lead to further insights about convex relaxations based methods for the MAP estimation problem.

Efficient Algorithms for Convex Relaxations

- We extended the TRW methods of [46, 101], which maximize the dual of the LP-S, to efficiently solve a larger class of relaxations. Recently, convergent optimization methods based on subgradients, which also solve the dual of the LP-S, have received considerable attention in the literature [49, 80, 81]. Unlike TRW-S (and similar to TRW-S(LP-C), TRW-S(SOCP-C) and TRW-S(SOCP-Q)), WTA is a necessary but not sufficient condition for their convergence. This results in better solutions of some MAP estimation problems encountered in Computer Vision [49]. Given their improvement in performance, the extension of subgradient based algorithms to LP-C, SOCP-C and SOCP-Q requires further investigation.

- Olsson *et al.* [68] have also recently proposed a subgradient based algorithm which attempts to solve the dual of the SDP relaxation. The merit of using this algorithm for Computer Vision (compared to more efficient algorithms such as TRW-S) needs to be explored. It would also be interesting to develop methods which take advantage of the strengths of the SDP relaxation solver of [68] within the TRW framework (e.g. for solving smaller subproblems such as (7.3.14) and (7.4.20) using SDP and further increasing the dual by reparameterization and averaging operation).

- All the methods discussed above (including our algorithms in chapter 7) provide a dual solution of a convex relaxation (since the convex relaxation itself is generally hard to solve efficiently). A labelling is then obtained using the dual solution, e.g. by employing the techniques of [46, 63] described in chapter 7. However, as mentioned earlier, the labellings obtained by the primal problems (i.e. the convex relaxation of the MAP estimation problem) have many interesting theoretical properties. For example, if the pairwise potentials of the random field define a Potts model, then the primal solution of the LP-S relaxation can provide us with a labelling which has a multiplicative bound of 2. Further work is required to extend these theoretical properties to the dual solutions of the convex relaxations.

Chapter 9

Appendix

Appendix A

Efficient Coarse to Fine Belief Propagation: We now modify the sum-product BP algorithm to overcome its two main drawbacks: (i) computational inefficiency; and (ii) large memory requirement. Recall that in sum-product BP, the message that a variable v_a sends to its neighbour v_b is given by

$$m_{ab;k} \leftarrow \eta_1 \sum_{l_i \in \mathbf{l}} \left(\exp(-\psi^1_{a;i}) \exp(-\psi^2_{ab;ij}) \prod_{v_c \in \mathbf{N}_a \setminus b} m_{ca;i} \right). \tag{1}$$

All messages are initialized to 1. The belief (marginal) of a variable v_a taking label l_i is given by

$$P'_{a;i} \leftarrow \eta_2 \exp(-\psi^1_{a;i}) \prod_{b \in \mathbf{N}_a} m_{ba;i}. \tag{2}$$

The time complexity of sum-product BP is $O(nh^2)$, where n is the number of random variables in the CRF and h is the number of labels. This makes sum-product BP computationally infeasible for large h. However, since the pairwise terms of the CRF are defined by a non-regular Potts model (see equation (4.3.2)), we show that the runtime of sum-product BP can be reduced to $O(nh')$, where $h' \ll h^2$ (using an extension of the method described in [24]). In particular, we can speed-up the computation of the message $\mathbf{m}_{ab}$ by precomputing the terms which are common in $m_{ab;k}$, for all labels l_k as follows:

$$T = \sum_{l_i} \left(\exp(-\psi^1_{a;i}) \prod_{c \in \mathbf{N}_a \setminus b} m_{ca;i} \right). \tag{3}$$

To compute the message $m_{ab;k}$ for a particular label l_k, we now consider only those labels l_i which define a rigid motion with l_k (see equation (4.3.2)). These labels are denoted by the set $\mathcal{R}_a(l_k)$. Specifically, let

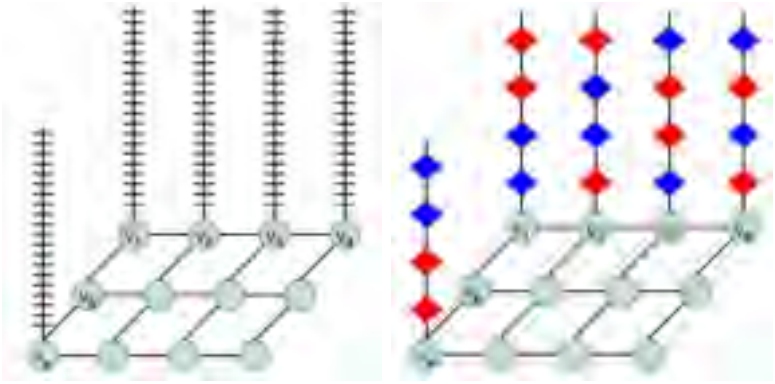
$$T_c = \sum_{l_i \in \mathcal{R}_a(l_k)} \left(\exp(-\psi^1_{a;i}) \prod_{c \in \mathbf{N}_a \setminus b} m_{ca;i} \right), \tag{4}$$

which can be computed efficiently by summing $h' = |\mathcal{R}_a(l_k)| \ll h$ terms. Clearly, the message $m_{ab;k}$ defined in equation (1) is equivalent to

$$m_{ab;k} \leftarrow \eta_1 \left(T_c \exp(1) + (T - T_c) \exp(\zeta \nabla(\mathbf{r}_a, \mathbf{r}_b)) \right). \tag{5}$$

Thus the messages can be computed efficiently in $O(nh')$ time where $h' \ll h^2$. Note that the above speed-up is valid for any random field with non-regular Potts model pairwise potentials. *For example, the same efficiency trick is used to match the layered pictorial structures model to an image in § 5.5.2.*

Another limitation of sum-product BP is that it has memory requirements of $O(nh)$. To overcome this problem, we use a variation of the coarse to fine strategy suggested in [99]. This allows us to solve $O(\log(h)/\log(H))$ problems of H labels instead of one problem of h labels, where $H \ll h$. Thus, the memory requirements are reduced to $O(nH)$. The time complexity is reduced further from $O(nh)$ to $O(\log(h)nH/\log(H))$.



(a) (b)

Figure 9.1: *Coarse to fine sum-product* BP. **(a)** *An example* CRF *with 12 variables and* $h = 20$ *labels (shown as trellises on top of the hidden nodes).* **(b)** *Each set of 5 consecutive labels is grouped into one representative label (shown as a diamond) thereby resulting in a coarser* CRF *with* $H = 4$ *labels. Inference is performed on this* CRF *using efficient sum-product* BP *[24]. In this case, the best* $R = 2$ *representative labels (shown as red diamonds) are chosen for each variable and expanded. This results in an* CRF *with 10 labels. The process of grouping the labels is continued until we obtain the* MMSE *estimate for all variables.*

The basic idea of the coarse to fine strategy is to group together similar labels in the label set $\mathbf{l}$ (differing slightly only in translation) to obtain H *representative* labels L_i (see Fig. 9.1). Let $\mathbf{L}$ denote the set of all representative labels L_i. We now define an CRF where each variable v_a can take one of H labels from the set $\mathbf{L}$. The unary potentials of this CRF are given by $\hat{\psi}^1_{a;i} = \max_{l_p \in L_i} \psi^1_{a;p}$ and the pairwise potentials are defined as $\hat{\psi}^2_{ab;ik} = \max_{l_p \in L_i, l_q \in L_k} \psi^2_{ab;pq}$. Using sum-product BP on this CRF, we obtain the approximate marginal probability (i.e. the beliefs) for each variable taking one of the representative labels. We choose the best R representative transformations (unlike [99], which chooses only the best) with the highest beliefs for each variable. These transformations are again divided into H representative transformations. Note that these H transformations are less coarse than the ones used previously. We repeat this process until we obtain the best transformation (in terms of the MMSE estimate) for each variable v_a. In our experiments described in chapter 4, we use $H = 165$ and $R = 20$. Sum-product BP was found to converge within 20 iterations at each stage.

Appendix B

Refining the shape of the segments by minimizing the energy of the layered representation, defined in equation (4.2.4), requires a series of graph cuts (as described in § 4.3.3). Below, we provide the graph constructions required for both the $\alpha\beta$-swap and the α-expansion algorithms.

Graph Construction for $\alpha\beta$-swap: The $\alpha\beta$-swap algorithm swaps the assignments of certain points belonging to segments s_α or s_β. We now present the graph construction required for performing $\alpha\beta$-swap such that it minimizes the energy of the layered representation. For clarity, we only consider the case when there are two neighbouring points a and b, i.e. the a^{th} and b^{th} points in the mattes $\mathbf{\Omega}_{M\alpha}$ and $\mathbf{\Omega}_{M\beta}$. The complete graph can be obtained by concatenating the graphs for all pairs of neighbouring points [48].

Each of the two points a and b are represented by one vertex in the graph (i.e. vertices V_a and V_b respectively). In addition, there are two special vertices called the source and the sink which represent the segments s_α and s_β. Recall that the unary potential for assigning point a to segment s_α is $\theta^1_{a_\alpha;1}$. Similarly, the unary potential for assigning a to segment s_β is $\theta^1_{a_\beta;1}$.

The pairwise potentials, given by equation (4.2.11), for all four possible assignments of two points a and b are summarized in Fig. 9.2. Here, $\gamma'_{ik}(a_i, b_k) = \kappa_2 - \gamma_{ik}(a, b)$ is the pairwise potential (i.e. prior term plus contrast term) for assigning points a and b to segments s_i and s_k. The corresponding graph construction, also shown in Fig. 9.2, is obtained using the method described in [48]. After the st-MINCUT of this graph is obtained, the points whose corresponding vertices are connected to the source belong to the segment s_α. Similarly, the points whose corresponding vertices are connected to the sink belong to the segment s_β.

	$b \in s_\alpha$	$b \in s_\beta$
$a \in s_\alpha$	0	$\gamma'_{\alpha\beta}(a, b)$
$a \in s_\beta$	$\gamma'_{\beta\alpha}(a, b)$	0

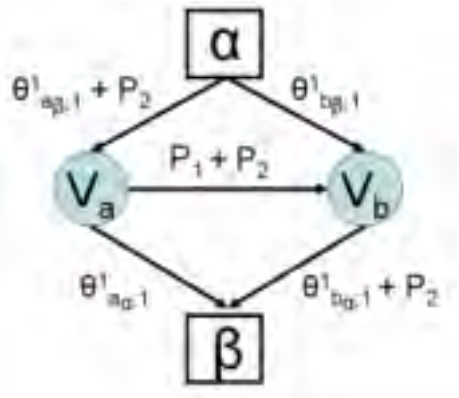


Figure 9.2: *Graph construction for $\alpha\beta$-swap. The table shown in the left summarizes the pairwise potentials for two points a and b. The figure on the right shows the corresponding graph construction. Here P_1 and P_2 are the $(1,2)^{th}$ and $(2,1)^{th}$ (i.e. the non-zero) elements of the table respectively.*

Graph Construction for α-expansion: The α-expansion algorithm relabels some points a_α to belong to the segment s_α. In other words, it attempts to assign the points which were missed by the initial estimate to the segment s_α. Again, we show the graph construction for only two neighbouring points, i.e. a_α and b_α in the matte $\mathbf{\Omega}_{M\alpha}$, for clarity. These points are represented using vertices V_{a_α} and V_{b_α} respectively.

Similar to the $\alpha\beta$-swap case, the unary potential of assigning a_α to s_α is $\theta^1_{a_\alpha;1}$. Recall that the unary potential of not assigning a point a_α to a segment s_α is given by the constant c_1 (see equation (4.2.5)).

The pairwise potentials for all four possible assignments of two points a_α and b_α are summarized in Fig. 9.3. Note that in accordance with the energy of the model, the pairwise potentials are summed over all segments which include the a^{th} or the b^{th} point on the corresponding mattes. Using the source and sink vertices to represent labels α and not-α (denoted by $\sim\alpha$) respectively the corresponding graph construction, shown in Fig. 9.3, can be obtained by the method described in [48]. After the st-MINCUT of this graph is obtained, the points whose corresponding vertices are connected to the source belong to the segment s_α. Similarly, the points whose corresponding vertices are connected to the sink do not belong to the segment s_α.

	$b_\alpha \in s_\alpha$	$b_\alpha \notin s_\alpha$
$a_\alpha \in s_\alpha$	0	$\sum_{i,b_i \in s_i} \gamma'_{\alpha i}(a_\alpha, b_i)$
$a_\alpha \notin s_\alpha$	$\sum_{i,a_i \in s_i} \gamma'_{i\alpha}(a_i, b_\alpha)$	0

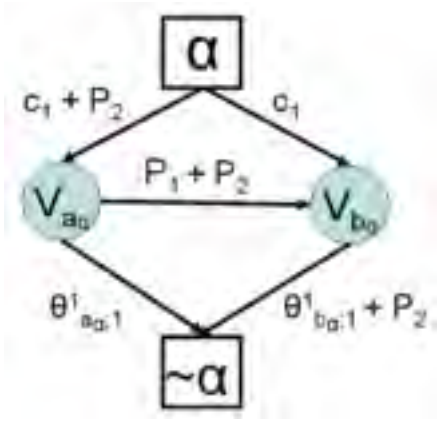


Figure 9.3: *Graph construction for α-expansion. The table shown in the left summarizes the pairwise potentials for two points a_α and b_α. The figure on the right shows the corresponding graph construction. Again, P_1 and P_2 are the $(1,2)^{th}$ and $(2,1)^{th}$ elements of the table respectively.*

Bibliography

[1] A. Agarwal and B. Triggs. Tracking articulated motion using a mixture of autoregressive models. In *ECCV*, pages III:54–65, 2004.

[2] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *ECCV*, pages IV: 113–127, 2002.

[3] D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, and A. Ng. Discriminative learning of markov random fields for segmentation of 3D range data. In *CVPR*, 2005.

[4] F. Barahona and A. Mahjoub. On the cut polytope. *Mathematical Programming*, 36:157–173, 1986.

[5] A. Ben-Tal and M. Bendsoe. A new method for optimal truss topology design. *SIAM Journal of Optimization*, 3:322–358, 1993.

[6] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of Royal Statistical Society*, 36, 1974.

[7] C. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[8] M Black and D. Fleet. Probabilistic detection and tracking of motion discontinuities. *IJCV*, 38:231–245, 2000.

[9] A. Blake, C. Rother, M. Brown, P. Perez, and P.H.S. Torr. Interactive image segmentation using an adaptive GMMRF model. In *ECCV*, pages I: 428–441, 2004.

[10] E. Borenstein and S. Ullman. Class-specific, top-down segmentation. In *ECCV*, pages II: 109–122, 2002.

[11] E. Boros and P. Hammer. Pseudo-boolean optimization. *Discrete Applied Mathematics*, 123:155–225, 2002.

[12] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[13] Y. Boykov and M.P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *ICCV*, pages I: 105–112, 2001.

[14] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 26(9):1124–1137, 2004.

[15] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 23(11):1222–1239, 2001.

[16] M. Bray, P. Kohli, and P. H. S. Torr. PoseCut: Simultaneous segmentation and 3D pose estimation of humans using dynamic graph cuts. In *ECCV*, pages II: 642–655, 2006.

[17] C. Chekuri, S. Khanna, J. Naor, and L. Zosin. Approximation algorithms for the metric labelling problem via a new linear programming formulation. In *SODA*, 2001.

[18] F. Cohen. *Modeling and Applications of Stochastic Processes*. Kluwer, 1986.

[19] D. Cremers and S. Soatto. Variational space-time motion segmentation. In *ICCV*, pages II:886–892, 2003.

[20] D. Cremers, N. Sochen, and C. Schnoerr. Mutliphase dynamic labelling for variational recognition-driven image segmentation. *IJCV*, 66:67–81, 2006.

[21] E. Dalhaus, D. Johnson, C. Papadimitriou, P. Seymour, and M. Yannakakis. The complexity of multi-terminal cuts. *SICOMP*, 23(4):864–894, 1994.

[22] E. de Klerk, D. Pasechnik, and J. Warners. Approximate graph colouring and max-k-cut algorithms based on the theta function. *Journal of Combinatorial Optimization*, 8(3):267–294, 2004.

[23] P.F. Felzenszwalb and D.P. Huttenlocher. Efficient matching of pictorial structures. In *CVPR*, pages II: 66–73, 2000.

[24] P.F. Felzenszwalb and D.P. Huttenlocher. Fast algorithms for large state space HMMs with applications to web usage analysis. In *NIPS*, 2003.

[25] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, pages II: 264–271, 2003.

[26] M.A. Fischler and R.A. Elschlager. The representation and matching of pictorial structures. *TC*, 22:67–92, January 1973.

[27] P. Fitzpatrick. *Advanced Calculus*. Thompson Brooks/Cole, 2006.

[28] D. Freedman and T. Zhang. Interactive graph cut based segmentation with shape priors. In *CVPR*, pages I: 755–762, 2005.

[29] D.M. Gavrilla. Pedestrian detection from a moving vehicle. In *ECCV*, pages II: 37–49, 2000.

[30] A. Gelman, J. Carlin, H. Stern, and D. Rubin. *Bayesian Data Analysis*. Chapman and Hall, 1995.

[31] M. Goemans and D. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of ACM*, 42:1115–1145, 1995.

[32] J. Goldstein, J. Platt, and C. Burges. Redundant bit vectors for quickly searching high-dimensional regions. In *Deterministic and Statistical Methods in Machine Learning*, pages 137–158, 2005.

[33] D. Grieg, B. Proteous, and A. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of Royal Statistical Society*, B(48), 1989.

[34] P. Hammer. Some network flow problems solved with pseudo-boolean programming. *Operations Research*, 13:388–399, 1965.

[35] P. Hammer, P. Hansen, and B. Simeone. Roof duality, complementation and persistency in quadratic 0-1 optimization. *Mathematical Programming*, 28:121–155, 1984.

[36] R. Huang, V. Pavlovic, and D.N. Metaxas. A graphical model framework for coupling mrfs and deformable models. In *CVPR*, pages II: 739–746, 2004.

[37] H. Ishikawa. Exact optimization for Markov random fields with convex priors. *PAMI*, 25(10):1333–1336, October 2003.

[38] N. Jojic and B. Frey. Learning flexible sprites in video layers. In *CVPR*, pages I: 199–206, 2001.

[39] A. Karzanov. Minimum 0-extension of graph metrics. *European Journal of Combinatorics*, 19:71–101, 1998.

[40] S. Kim and M. Kojima. Second-order cone programming relaxation of nonconvex quadratic optimization problems. Technical report, Tokyo Institute of Technology, 2000.

[41] R. Kindermann and J. L. Snell. *Markov random fields and their applications*. AMS, 1980.

[42] J. Kleinberg and E. Tardos. Approximation algorithms for classification problems with pairwise relationships: Metric labeling and Markov random fields. In *STOC*, pages 14–23, 1999.

[43] P. Kohli, M. P. Kumar, and P. H. S. Torr. P3 & beyond: Solving energies with higher order cliques. In *CVPR*, 2007.

[44] P. Kohli and P. H. S. Torr. Efficiently solving dynamic Markov random fields using graph cuts. In *ICCV*, pages II: 922–929, 2005.

[45] P. Kohli and P. H. S. Torr. Measuring uncertainty in graph cut solutions: Efficiently computing min-marginal energies using dynamic graph cuts. In *ECCV*, pages I: 30–43, 2006.

[46] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *PAMI*, 28(10):1568–1583, 2006.

[47] V. Kolmogorov and M. Wainwright. On the optimality of tree-reweighted max-product message passing. In *UAI*, 2005.

[48] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts. *IEEE PAMI*, 26(2):147–159, 2004.

[49] N. Komodakis, N. Paragios, and G. Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *ICCV*, 2007.

[50] N. Komodakis and G. Tziritas. A new framework for approximate labeling via graph cuts. In *ICCV*, pages II: 1018–1025, 2005.

[51] A. Koster, C. van Hoesel, and A. Kolen. The partial constraint satisfaction problem: Facets and lifting theorems. *Operations Research Letters*, 23(3-5):89–97, 1998.

[52] J. Kruskal. On the shortest spanning subtree of a graph and the travelling salesman problem. *American Mathematical Society*, 7(1):48–50, 1956.

[53] M. P. Kumar, P. H. S. Torr, and A. Zisserman. Extending pictorial structures for object recognition. In *BMVC*, pages II: 789–798, 2004.

[54] M. P. Kumar, P. H. S. Torr, and A. Zisserman. Learning layered pictorial structures from video. In *ICVGIP*, pages 148–153, 2004.

[55] M. P. Kumar, P. H. S. Torr, and A. Zisserman. Solving Markov random fields using second order cone programming relaxations. In *CVPR*, volume I, pages 1045–1052, 2006.

[56] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labelling sequence data. In *ICML*, 2001.

[57] J. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal of Optimization*, 11:796–817, 2001.

[58] B. Leibe and B. Schiele. Interleaved object categorization and segmentation. In *BMVC*, pages II: 264–271, 2003.

[59] A. Levin and Y. Weiss. Learning to combine bottom-up and top-down segmentation. In *ECCV*, pages IV: 581–594, 2006.

[60] D. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge Unviersity Press, 2003.

[61] D.R. Magee and R.D. Boyle. Detecting lameness using re-sampling condensation and multi-stream cyclic hidden markov models. *IVC*, 20(8):581–594, 2002.

[62] P. Meer and B. Georgescu. Edge detection with embedded confidence. *PAMI*, 23:1351–1365, December 2001.

[63] T. Meltzer, C. Yanover, and Y. Weiss. Globally optimal solutions for energy minimization in stereo vision using reweighted belief propagation. In *ICCV*, 2005.

[64] M. Muramatsu and T. Suzuki. A new second-order cone programming relaxation for max-cut problems. *Journal of Operations Research of Japan*, 43:164–177, 2003.

[65] K. Murphy. An introduction to graphical models. Technical report, UBC, 2001.

[66] K. Murphy, Y. Weiss, and M. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *UAI*, pages 467–476, 1999.

[67] G. Nemhauser and L. Wolsey. *Integer and combinatorial optimization*. Wiley-Interscience, 1999.

[68] C. Olsson, A.P. Eriksson, and F. Kahl. Solving large scale binary quadratic problems: Spectral methods vs. semidefinite programming. In *CVPR*, pages I: 1–8, 2007.

[69] A. Opelt, A. Pinz, and A. Zisserman. Incremental learning of object detectors using a visual shape alphabet. In *CVPR*, pages I: 3–10, 2006.

[70] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kauffman, 1998.

[71] M. Prasad, A. Zisserman, A. W. Fitzgibbon, M. P. Kumar, and P. H. S. Torr. Learning class-specific edges for object detection and segmentation. In *ICVGIP*, 2006.

[72] R. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36:1389–1401, 1957.

[73] S. Pundlik and S. Birchfield. Motion segmentation at any speed. In *BMVC*, pages 427–436, 2006.

[74] D. Ramanan and D.A. Forsyth. Using temporal coherence to build models of animals. In *ICCV*, pages 338–345, 2003.

[75] P. Ravikumar and J. Lafferty. Quadratic programming relaxations for metric labelling and Markov random field MAP estimation. In *ICML*, 2006.

[76] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: interactive foreground extraction using iterated graph cuts. In *SIGGRAPH*, pages 309–314, 2004.

[77] C. Schellewald and C. Schnorr. Subgraph matching with semidefinite programming. In *IWCIA*, 2003.

[78] D. Schlesinger and B. Flach. Transforming an arbitrary minsum problem into a binary one. Technical Report TUD-F106-01, Dresden University of Technology, 2006.

[79] M. Schlesinger. Sintaksicheskiy analiz dvumernykh zritelnikh singnalov v usloviyakh pomekh (syntactic analysis of two-dimensional visual signals in noisy conditions). *Kibernetika*, 4:113–130, 1976.

[80] M. Schlesinger and V. Giginyak. Solution to structural recognition (MAX,+)-problems by their equivalent transformations. Part 1. *Control Systems and Computers*, 1:3–15, 2007.

[81] M. Schlesinger and V. Giginyak. Solution to structural recognition (MAX,+)-problems by their equivalent transformations. Part 2. *Control Systems and Computers*, 2:3–18, 2007.

[82] M. I. Schlesinger and B. Flach. Some solvable subclass of structural recognition problems. In *Czech Pattern Recognition Workshop*, 2000.

[83] A. Shahrokni, F. Fleuret, and P. Fua. Classifier-based contour tracking for rigid and deformable objects. In *BMVC*, 2005.

[84] J. Shotton, A. Blake, and R. Cipolla. Contour-based learning for object detection. In *ICCV*, pages I: 503–510, 2005.

[85] J. Shotton, J. Winn, C. Rother, and A. Criminisi. TextonBoost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, pages I: 1–15, 2006.

[86] H. Sidenbladh and M.J. Black. Learning the statistics of people in images and video. *IJCV*, 54(1):181–207, September 2003.

[87] L. Sigal, S. Bhatia, S. Roth, M.J. Black, and M. Isard. Tracking loose-limbed people. In *CVPR*, pages 421–428, 2004.

[88] B. Stenger, A. Thayananthan, P.H.S. Torr, and R. Cipolla. Hand pose estimation using heirarchical detection. In *Intl. Workshop on Human-Computer Interaction*, pages 105–116, 2004.

[89] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for markov random fields. In *ECCV*, pages II: 16–29, 2006.

[90] H. Taha. *Operations research: An introduction*. Prentice Hall, 2006.

[91] M. Tappen and W. Freeman. Comparison of graph cuts with belief propagation for stereo. In *ICCV*, pages 900–907, 2003.

[92] A. Thayananthan, B. Stenger, P.H.S. Torr, and R. Cipolla. Shape context and chamfer matching in cluttered scenes. In *CVPR*, pages I: 127–133, 2003.

[93] P. H. S. Torr. Solving Markov random fields using semidefinite programming. In *AISTATS*, 2003.

[94] P. H. S. Torr, R. Szeliski, and P. Anandan. An integrated bayesian approach to layer extraction from image sequences. *IEEE PAMI*, 23(3):297–304, 2001.

[95] P. H. S. Torr and A Zisserman. Feature based methods for structure and motion estimation. In W. Triggs, A. Zisserman, and R. Szeliski, editors, *International Workshop on Vision Algorithms*, pages 278–295, 1999.

[96] A. Torralba, K.P. Murphy, and W.T. Freeman. Sharing visual features for multiclass and multiview object detection. *PAMI*, 29(5):854–869, 2007.

[97] K. Toyama and A. Blake. Probabilistic tracking in a metric space. In *ICCV*, pages II: 50–57, 2001.

[98] M. Varma and A. Zisserman. Texture classification: Are filter banks necessary? In *CVPR*, pages II:691–698, 2003.

[99] G. Vogiatzis, P. H. S. Torr, S. Seitz, and R. Cipolla. Reconstructing relief surfaces. In *BMVC*, pages 117–126, 2004.

[100] M. Wainwright, T. Jaakola, and A. Willsky. Tree-reweighted belief propagation and approximate ML estimation by pseudo-moment matching. In *AISTATS*, 2003.

[101] M. Wainwright, T. Jaakola, and A. Willsky. MAP estimation via agreement on trees: Message passing and linear programming. *IEEE Trans. on Information Theory*, 51(11):3697–3717, 2005.

[102] M. Wainwright and M. Jordan. Graphical models, exponential families, and variational inference. Technical Report 649, University of California, Berkeley, 2003.

[103] M. Wainwright and M. Jordan. Treewidth-based conditions for exactness of the Sherali-Adams and Lasserre relaxations. Technical Report 671, University of California, Berkeley, 2004.

[104] J. Wang and E. Adelson. Representing moving images with layers. *IEEE Trans. on IP*, 3(5):625–638, 1994.

[105] Y. Weiss and E. Adelson. A unified mixture framework for motion segmentation. In *CVPR*, pages 321–326, 1996.

[106] Y. Weiss and W. Freeman. On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs. *IEEE Trans. on Information Theory*, 47(2):723–735, 2001.

[107] T. Werner. A linear programming approach to max-sum problem: A review. *PAMI*, 2007.

[108] C. Williams and M. Titsias. Greedy learning of multiple objects in images using robust statistics and factorial learning. *Neural Computation*, 16(5):1039–1062, 2004.

[109] J. Wills, S. Agarwal, and S. Belongie. What went where. In *CVPR*, pages I:37–44, 2003.

[110] J. Winn and A. Blake. Generative affine localisation and tracking. In *NIPS*, pages 1505–1512, 2004.

[111] J. Winn and N. Jojic. LOCUS: Learning Object Classes with Unsupervised Segmentation. In *ICCV*, pages I: 756–763, 2005.

[112] J. Winn and J. Shotton. The layout consistent random field for recognizing and segmenting partially occluded objects. In *CVPR*, pages I: 37–44, 2006.

[113] C. Yanover and Y. Weiss. Finding the M most probable configurations in arbitrary graphical models. In *NIPS*, 2004.

[114] J. Yedidia, W. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. Technical Report TR2001-22, MERL, 2001.

[115] A. Yuille. CCCP algorithms to minimize the Bethe and Kikuchi free energies: Convergent alternatives to belief propagation. *Neural Computation*, 14(7):1691–1722, 2002.