博士論文

Discrete Inference Approaches to Image Segmentation and Dense Correspondence

（画像領域分割と対応点推定問題への離散最適化アプローチ）

東京大学

大学院情報理工学系研究科

電子情報学専攻

谷合　竜典

指導教員　　佐藤 洋一　教授

The University of Tokyo
Graduate School of Information Science and Technology
Department of Information and Communication Engineering

Discrete Inference Approaches to Image Segmentation and Dense Correspondence

written by

Tatsunori Taniai

A Ph.D. Thesis presented to the Graduate School of Information Science and Technology of The University of Tokyo in partial fulfillment of the requirements for the degree of Philosophy Doctor in Information Science and Technology.

Advisor: Professor Yoichi Sato

December 2016

Committee Members

Professor Yoichi Sato
The University of Tokyo, Japan

Professor Kiyoharu Aizawa
The University of Tokyo, Japan

Professor Shin'ichi Sato
The University of Tokyo, Japan

Professor Takeshi Naemura
The University of Tokyo, Japan

Professor Toshihiko Yamasaki
The University of Tokyo, Japan

Professor Hiroshi Ishikawa
Waseda University, Japan

December 2016

I dedicate this dissertation to myself and all my family, friends and professors who gave me all necessary support to get here.

Abstract

We consider discrete inference approaches to image segmentation and dense correspondence. The two problems cover diverse tasks such as image segmentation, binarization, cosegmentation, motion segmentation, binocular stereo vision, optical flow and general dense correspondence, which are addressed sorely or jointly in this work as energy minimization problems on Markov random fields (MRFs). Discrete inference approaches are employed to effectively optimize inherently discrete functions or highly non-convex continuous functions. The contributions of this work are two folds: proposal of novel joint frameworks of image segmentation and dense correspondence problems, and development of new inference techniques for sole or joint tasks. Specifically, we comprehensively address three challenges of discrete inference, that is, label space size, higher-order energy, and non-submodular energy, which are posed in various forms in the following tasks that we tackle.

First, we study inference problems on non-submodular and higher-order MRFs that have binary variables. Such problems naturally appear in low-level computer vision tasks such as image segmentation and binarization of gray images. They are also imposed as subproblems in estimation of more general multi-valued or continuous-valued variables. For such fundamental inference problems, we develop a new theoretical insight into several existing optimization methods and propose a new method by unifying them. The proposed method has a mechanism to better avoid bad local minimums of non-submodular functions, and is thus more robust to initializations compared to existing methods. The proposed method was evaluated on image segmentation and binarization tasks and was shown to outperform state-of-the-art methods.

Second, we propose an efficient and accurate binocular stereo matching method, whose model and inference both favor piecewise planar surfaces. We formulate the stereo problem by a model of per-pixel local 3D surface planes with piecewise planar smoothness regularization, which forms a pairwise MRF with a continuous 3D label space. In order to efficiently infer this rich model, we propose a new inference technique that extends the well-known expansion move algorithm by incorporating the spatial propagation and randomization search mechanisms of PatchMatch inference. Unlike conventional fusion-based approaches, the proposed method does not require solution proposals and also produces submodular energies that are optimally minimized by graph cuts during the inference. The computations can be easily accelerated by parallelization and using fast cost-map filtering. The proposed method achieved the state-of-the-art performance on the Middlebury stereo benchmark among more than 160 stereo algorithms.

Third, we propose a unified framework of general dense correspondence and cosegmentation for two images, where common "foreground" regions in the two images are

segmented and aligned to each other. Our method is formulated using a hierarchical MRF model with joint labels of segmentation and correspondence. The correspondence field is parameterized using similarity transformations (4-DOF) assigned on superpixels. The hierarchy is used to evaluate correspondence across various coarseness of superpixels, which brings high robustness when aligning objects with different appearances. Unlike prior hierarchical methods which assume that the structure is given, we dynamically recover the structure along with the correspondence and segmentation labeling. This joint inference is performed in an energy minimization framework using iterated graph cuts. The proposed method was quantitatively evaluated on a new dataset and it outperformed state-of-the-art methods designed specifically for either cosegmentation or correspondence estimation.

Finally, we propose a fast scene flow method for stereo image sequences that simultaneously recovers motion segmentation of moving objects as well as camera ego-motion. This framework unifies four tasks —stereo, optical flow, motion segmentation and visual odometry— providing rich information of disparity, 2D flow and binary segmentation of moving objects at every pixel along with camera motion. The inference is carried out through a multi-staged pipeline where the solution to one task benefits others, leading to overall higher accuracy and efficiency. The proposed method was evaluated on the KITTI 2015 scene flow benchmark and was ranked third. Furthermore, our CPU implementation processed each frame in 2–3 seconds, which was 1–3 orders of magnitude faster than the top six methods that took 1–50 minutes per frame. Our method was also thoroughly evaluated on challenging Sintel sequences having fast camera and object motion, where our method consistently outperformed the method ranked second on the KITTI benchmark.

Keywords: Markov random field, energy minimization, higher-order energy, discrete-continuous optimization, segmentation, stereo, optical flow, dense correspondence

Contents

List of Figures

List of Tables

List of Algorithms

List of Acronyms

AC	auxiliary cuts
BJ	Boykov and Jolly
BP	belief propagation
DSP	deformable spatial pyramid
FTR	fast trust region
GC	graph cut
MAP	maximum a posteriori
MPBP	max-product belief propagation
MPM	maximum posterior marginal
MRF	Markov random field
OSF	object scene flow
LBP	loopy belief propagation
LSA	local submodular approximation
PBP	particle belief propagation
PMBP	PatchMatch belief propagation
pPBC	parametric pseudo bound cuts
PRSM	piecewise rigid scene model
QPBO	quadratic pseudo-boolean optimization
SDC	superdifferential cuts
SGM	semi-global matching
SIFT	scale-invariant feature transform
SSP	submodular-supermodular procedure
TRW	tree-reweighted message passing
TRW-S	sequential tree-reweighted message passing

1 Introduction

1.1 Markov Random Fields in Computer Vision

MARKOV random fields (MRFs) [Geman and Geman, 1984] are considered a fundamental probabilistic model for various low and middle-level computer vision tasks such as image restoration [Boykov *et al.*, 1998], image segmentation [Boykov and Jolly, 2001; Rother *et al.*, 2004], stereo vision [Kolmogorov and Zabih, 2001, 2002; Woodford *et al.*, 2008, 2009], optical flow [Lempitsky *et al.*, 2007, 2008; Xu *et al.*, 2012] and super resolution [Rajan and Chaudhuri, 2002]. In this approach, an unknown solution is formulated as a mapping function $f_p = f(p) : \Omega \rightarrow \mathcal{L}$ that assigns each site $p \in \Omega$ a value from a label space $\mathcal{L}$. The sites p often represent the pixels of a reference image. For example, in image segmentation problems, we estimate a discrete object labeling $f(p) : \Omega \rightarrow \{1, 2, \cdots, K\}$ on the image domain Ω. When $K = 2$, the labeling indicates a binary object mask. In stereo vision, we estimate unknown depth values $z = f(p) \in \mathbb{R}^+$ at individual pixels p on the image domain Ω. Such mapping functions $f(p) : \Omega \rightarrow \mathbb{R}^+$ are often called *depth maps*. In optical flow problems, 2D vector fields $f(p) : \Omega \rightarrow \mathbb{R}^2$ as known as *flow maps* are estimated for image sequences, where each value $\mathbf{u} = f(p)$ indicates a 2D motion vector (or 2D translation) from a pixel p in a reference image to its visually corresponding point in another image. See Figures 1.1–1.3 for visualization of mapping functions f in segmentation, stereo and optical flow.

MRF-based approaches infer desired solutions by minimizing energy functions that are generally formulated as follows.

$$E(f) = \underbrace{\sum_{p \in \Omega} \phi_p(f_p)}_{\text{Unary term}} + \underbrace{\sum_{(p,q) \in \mathcal{N}} \phi_{pq}(f_p, f_q)}_{\text{Pairwise term}} + \underbrace{\sum_{c \in \mathcal{C}} \phi_c(f_{c_1}, f_{c_2}, ..., f_{c_n})}_{\text{Higher-order term}} . \tag{1.1}$$

Here, $\mathcal{N}$ is a set of node pairs that represents a neighborhood system for Ω. $\mathcal{C}$ is a set of *cliques* that represents subsets of nodes having mutual dependencies. The first term defined for individual nodes $p \in \Omega$ is called a *unary term*, and the second term defined for

Figure 1.1 Illustration of image segmentation. Defined on a reference image (left), a solution $f : \Omega \to \{1, 2, \cdots, K\}$ represents a discrete object labeling (right).

Figure 1.2 Illustration of stereo vision. Defined on a reference image (left), a solution $f : \Omega \to \mathbb{R}^+$ represents a depth map (right).

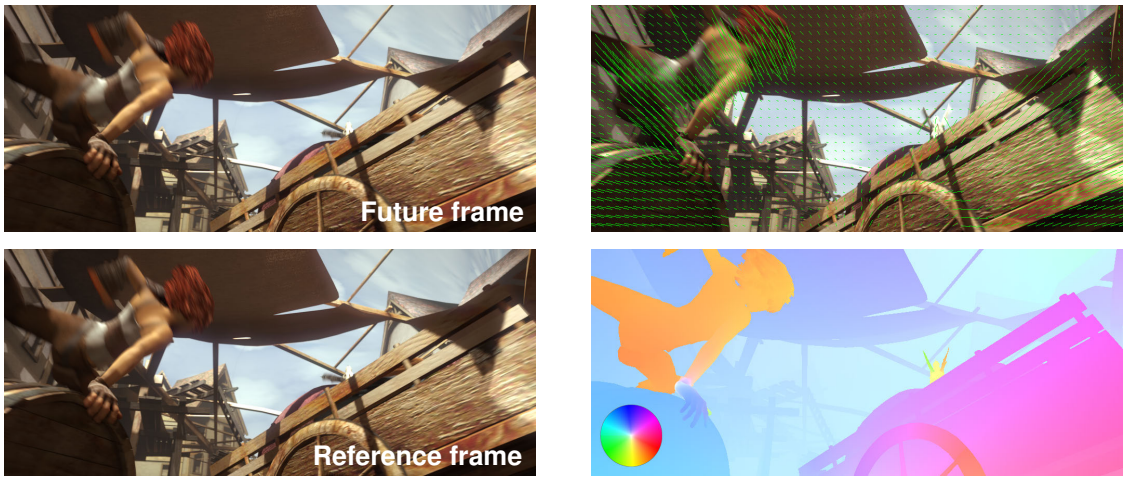


Figure 1.3 Illustration of optical flow. Given an image sequence (left), a solution $f : \Omega \to \mathbb{R}^2$ represents a 2D-motion vector field or flow map (right). Motion vectors from the reference to future frame are visualized by green lines in top-right and by color-code in bottom-right.

neighbor pairs $(p, q) \in \mathcal{N}$ is called a *pairwise term*. The last term involving more than two nodes in cliques $c \in \mathcal{C}$, is called a *higher-oder term*. MRFs with only unary and pairwise terms are called *pairwise MRFs* or *first-order MRFs*, while MRFs containing higher-order terms are called *higher-order MRFs*. These terms are often used for specific purposes as explained below.

Unary Term: The unary term is often called a *data term*, because it is typically used for evaluating (dis)likelihood that each node p takes some value f_p given observed data. More specifically, when it is expected from observed data that f_p is likely to take some value α than β, the cost $\phi_p(\alpha)$ for f_p taking α will be lower than the cost $\phi_p(\beta)$ for taking β, so that f_p more favors α. The unary data terms provide direct cues for desired solutions, but they are usually noisy. This leads to introducing additional terms below.

Pairwise Term: The pairwise term is typically defined for pixel neighbor pairs $(p, q) \in \mathcal{N}$ on the image pixel grid. This form of term is often called a *smoothness term* or *regularization term*, because $\phi_{pq}(f_p, f_q)$ is usually used for enforcing two values f_p and f_q of neighboring nodes to be spatially smooth. Typically, the smoothness terms is defined as $\phi_{pq}(f_p, f_q) = 0$ if $f_p = f_q$ and $\phi_{pq}(f_p, f_q) > 0$ if $f_p \neq f_q$. For example, stereo methods often use a smoothness term that enforces f_p and f_q to take close depth values, and segmentation methods enforce the two values to take the same object label.

Higher-order Term: Although pairwise MRFs are widely used for their well-balanced trade-off between efficiency and accuracy, their expressive power is limited. When data terms are noisy and provide only weak cues for desired solutions, the minimizers of such simple energy functions are strongly biased toward undesired solutions. Such a bias is well known as a *front-parallel bias* in stereo [Woodford *et al.*, 2009] and a *short-boundary bias* in segmentation [Jegelka and Bilmes, 2011]. Higher-order terms are used to overcome this limitation. Exploiting higher-order information becomes especially very crucial for medical image processing [Kitamura *et al.*, 2016; Gorelick *et al.*, 2013, 2012] each pixel in observed images can provide only limited information of a noisy gray-scale intensity. Also, some special forms of higher-order terms can be efficiently converted to pairwise forms by adding auxiliary variables [Kohli *et al.*, 2007, 2009; Delong *et al.*, 2012].

Generally, more sophisticated models have higher expressive power that can capture more realistic behaviors of desired solutions and data structures. This is in-turn traded-off by lower efficiency due to difficulty of inference on complicated models. In other words, existence of powerful and efficient inference methods enables us to use sophisticated models and achieve higher overall performances in various applications.

1.2 Inference on Markov Random Fields

Depending on what type of optimizers we use to minimize the energy function in Eq. (1.1), there are mainly two groups of approaches for inference on MRFs: continuous and discrete inference approaches.

Continuous inference approaches [Lucas and Kanade, 1981; Shi and Malik, 2000; Baker and Matthews, 2004; Pock *et al.*, 2008; Levin *et al.*, 2008; Mollenhoff *et al.*, 2016; Laude *et al.*, 2016; Lv *et al.*, 2016] use continuous optimizers that are based on the principle of gradient descent. While continuous optimizers are very powerful tools for optimizing convex functions, the energy functions of our interests are mostly highly non-convex. Because of the non-convexity, continuous inference methods tend to converge into bad local minimum solutions. To avoid this, continuous inference methods require good initializations obtained by discrete inference methods or require relaxation to the functions to reduce or remove the non-convexity. However, because the relaxation approaches change the form of functions, the global minimums of the relaxed functions may not correspond to those of the original functions. Also, these methods often require the energy functions be differentiable.

On the other hand, discrete inference approaches use discrete optimizers such as graph cuts (GC) [Kolmogorov and Zabin, 2004] and message passing algorithms [Felzenszwalb and Huttenlocher, 2006; Kolmogorov, 2006] that are based on combinatorial optimization. While it is straightforward to apply discrete optimizers for the inference of discrete-valued solutions (*e.g.*, segmentation labeling), they are also used to infer solutions that reside in continuous spaces. This type of approach is called *discrete-continuous optimization*. In contrast to pure continuous optimization, discrete-continuous optimization is very insensitive to initializations and has better ability to avoid bad local minimums of non-convex functions [Chen and Koltun, 2016; Lempitsky *et al.*, 2010]. In fact, on a well-known stereo benchmark [Scharstein and Szeliski, 2002] where more than 200 stereo algorithms have been registered, most of the methods fully or partly employ discrete inference approaches, because of the non-convexity inherent in the stereo problem formulations.

In this dissertation, we also study discrete inference approaches in various applications regarding image segmentation and dense correspondence. Generally, discrete inference approaches have the following three challenges, which are comprehensively addressed in this study.

Label Space Size: When we naively use a discrete optimizer for a labeling problem, we take an exhaustive search approach where all possible labels in the label space $\mathcal{L}$ are enumerated as candidates for combinatorial optimization. Hence, when the label space $\mathcal{L}$ is enormous, such exhaustive approaches become computationally intractable.

Discrete-continuous optimization is considered a typical instance of this case where $\mathcal{L}$ is a continuous space and the number of candidate labels is essentially infinite. Similarly, when the label space $\mathcal{L}$ becomes multi-dimensional (*e.g.*, $|\mathcal{L}| = 2$ in optical flow), the label number increases exponentially with the dimension size.

Higher-order Energy: In principle, the order of an energy function also exponentially influences the computation amount of discrete optimization methods [Ishikawa, 2011], except for some special forms of higher-order terms [Delong *et al.*, 2012; Kohli *et al.*, 2009]. While optimization of higher-order terms with a clique size of three [Woodford *et al.*, 2009] or four [Ishikawa, 2011] may be computationally tractable, more higher-order energies will require proper approximations for efficient inference.

Non-submodular Energy: For discrete functions (especially, functions with the binary label space $S = \{0, 1\}$), there is a special property named *submodularity*, which is a similar concept to convexity of continuous functions. It is well known that submodular functions can be minimized in polynomial times [Schrijver, 2000; Iwata *et al.*, 2001]. However, when the energy functions contain non-submodular terms, minimization of those functions generally becomes NP-hard problems.

1.3 Contributions

In this thesis, we study discrete inference approaches to image segmentation and dense correspondence estimation. The two problems cover a variety of tasks such as image segmentation, binarization, cosegmentation, motion segmentation, binocular stereo vision, optical flow and general dense correspondence, which are addressed sorely or jointly in this work as energy minimization problems on MRFs.

The contributions of this work are two folds: proposal of novel joint frameworks of multiple tasks, and development of new inference techniques for sole or joint tasks. As contributions on frameworks, we propose unified methods for new joint problems of image segmentation and dense correspondence, after thoroughly studying each problem. As contributions on inference algorithms, we comprehensively address the aforementioned three difficulties of discrete inference that are posed by individual tasks in different way. Here, instead of directly applying existing core optimizers, we carefully tailor them to individual tasks in order to account for the properties of optimizers as well as the properties specific to the tasks. We summarize the contributions of this work in Table 1.1 and explain more details below. This also provides an overview of the main four chapters of this thesis.

Table 1.1 Summary of tasks and inference challenges addressed in this work. In the left side, we show which tasks in computer vision are addressed in this work. We mainly focus on three tasks: image segmentation, binocular stereo vision, and optical flow. In the right side, we also show which of the three challenges – label space size, higher-order energy, non-submodular energy – in discrete inference approaches (described in Section 1.2) are posed by individual problems and addressed in this work. Note that when multiple tasks are simultaneously addressed, label spaces of individual tasks are concatenated to form the overall label space, which is expressed using a symbol of "+" in the table. Also notice that submodularity or non-submodularity is a property specific to energies with the binary label space, which is thus not applicable to other non-binary inference problems.

	Tasks			Inference challenges		
	Seg.	Stereo	Flow	Label space	MRF form	Non-submodular
Chap. 3	✓	–	–	Binary	**Higher-order**	**Yes**
Chap. 4	–	✓	–	**Continuous 3D**	Pairwise	(Non-convex)
Chap. 5	✓	–	✓	**Bin. + Cont. 4D**	**Hierarchical**	(Non-convex)
Chap. 6	✓	✓	✓	**Bin. + 1D + 2D**	Pairwise	(Non-convex)

Chapter 3: Binary MRF Inference for Segmentation and Low Level Vision

First, we study inference on binary MRFs that have the binary label space $\mathcal{L} = \{0, 1\}$. Particularly, inference on higher-order and non-submodular MRFs is addressed. Such problems naturally appear in image segmentation, binarization and deconvolution [Tang *et al.*, 2014] and become crucial in medical image processing [Kitamura *et al.*, 2016]. Furthermore, optimization of binary MRFs is a central problem in more general multi-valued MRF inference. Thus, binary MRF optimization can be considered the most fundamental problem in MRF inference.

In this study, we show theoretical links between several existing optimization techniques [Narasimhan and Bilmes, 2005; Ayed *et al.*, 2015, 2013; Tang *et al.*, 2014; Gorelick *et al.*, 2014] and propose a new method by unifying them. The proposed method has a mechanism to better avoid bad local minimums of non-submodular discrete functions and is more robust to initializations compared to other methods. The proposed method was evaluated on image segmentation and binarization tasks and was shown to outperform state-of-the-art optimization methods.

Chapter 4: Continuous MRF Inference for Binocular Stereo Vision

Second, we study discrete-continuous optimization for a binocular stereo vision problem. Employing an accurate stereo matching data term of [Bleyer *et al.*, 2011] and a pairwise smoothness term of [Olsson *et al.*, 2013], stereo matching is formulated as recovery of per-pixel local 3D surface planes. Hence, a pairwise MRF used here has a continuous and three-dimensional label space $\mathcal{L}$ that represents the parameter space of surface planes.

For this inference problem, we propose a new optimization technique that extends the well-known expansion move algorithm [Boykov *et al.*, 2001] by incorporating fast

inference mechanisms of PatchMatch [Barnes *et al.*, 2009, 2010]. Compared to existing approaches to continuous MRF inference, the proposed technique has several advantages. It produces submodular energies during the inference, which are optimally minimized by graph cuts. Both the proposed model and inference are designed to favor locally planar surfaces that are realistic in many scenes. It can be easily accelerated by parallelization and fast cost-map filtering. The proposed method was evaluated on the Middlebury benchmark (version 2) [Scharstein and Szeliski, 2002] and ranked first among more than 160 stereo methods.

Chapter 5: Joint Hierarchical MRF Inference for General Dense Correspondence and Cosegmentation

Third, a novel joint problem of general dense correspondence and cosegmentation is addressed. General dense correspondence [Liu *et al.*, 2011] is a problem of aligning the regions of same or similar objects in two images showing different scenes. Compared to conventional correspondence problems such as stereo and optical flow, it additionally imposes a fundamental difficulty of robustly aligning objects whose appearances may significantly differ. We solve this problem in a more proper setting where we jointly estimate valid correspondence regions or cosegmentation [Rother *et al.*, 2006] in the two images. The proposed method is useful, *e.g.*, for data augmentation [Smith *et al.*, 2013; Liu *et al.*, 2011] by transferring labelings on annotated images to unlabeled images, or for estimating 3D shapes from images that are obtained through the Internet search [Vicente *et al.*, 2014].

For this task, we propose a novel hierarchical MRF with joint labels of segmentation and correspondence. The correspondence field is parameterized using similarity transformations (4-DOF) assigned on superpixels. The superpixels have a nested structure forming a hierarchy in each image, which brings high robustness in correspondence estimation. Unlike prior hierarchical methods which assume that the structure is given, we dynamically recover the structure along with the correspondence and segmentation labeling. This joint inference involves higher-order energies and is performed in an energy minimization framework using iterated graph cuts. The proposed method was quantitatively evaluated on a new dataset and it outperformed state-of-the-art methods designed specifically for either cosegmentation or correspondence estimation.

Chapter 6: Joint MRF Inference for Stereo Scene Flow and Motion Segmentation

Finally, we propose a fast stereo scene flow method that unifies four tasks of stereo, optical flow, motion segmentation and visual odometry. Hence, the proposed method provides at every pixel its depth value, 2D flow vector, and binary segmentation label

indicating moving objects in a rigid scene. It also estimates the 6-DOF camera-ego motion from visual odometry. This rich information is useful in various applications such as video analysis and editing, 3D mapping, autonomous driving [Menze and Geiger, 2015] and mobile robotics.

Despite the high dimensionality of the overall label space, the inference is efficiently carried out through a proposed multi-staged process. Here, the four tasks are solved collaboratively through a pipeline where the solution to one task benefits others. This leads to higher accuracy and also increases the overall computational efficiency. The proposed method was evaluated on KITTI 2015 scene flow benchmark [Menze and Geiger, 2015] using real images from driving scenes, and was ranked third by overall accuracy. Furthermore, a CPU implementation of our method efficiently processed each frame in 2–3 seconds, which was 1–3 orders of magnitude faster than top six methods on the KITTI benchmark that took 1–50 minutes per frame.

1.4 Thesis Overview

The rest of this thesis is organized as follows.

In Chapter 2, we first review theoretical background of MRFs as well as basic discrete inference techniques using graph cuts and message passing. We then briefly review basic problem settings of image segmentation and dense correspondence, and also introduce related works in these areas.

In Chapter 3, we present our first study on binary MRF inference for segmentation and low level vision. This chapter primarily covers the contents of a published work [Taniai *et al.*, 2015] with some additional descriptions, details, and proofs.

In Chapter 4, we present our second study on continuous MRF inference for binocular stereo vision. This chapter includes the contents of an unpublished work [Taniai *et al.*, 2016a], whose main ideas have been presented in a published work [Taniai *et al.*, 2014]. Although the two versions share essentially the same concept, the algorithm presented in this chapter as well as [Taniai *et al.*, 2016a] has been polished and extended since [Taniai *et al.*, 2014].

In Chapter 5, we present our third study on joint hierarchical MRF inference for general dense correspondence and cosegmentation. This chapter primarily covers the contents of a published work [Taniai *et al.*, 2016b] with some additional details and proofs.

In Chapter 6, we present our fourth study on joint MRF inference for stereo scene flow and motion segmentation. This chapter includes the contents of an unpublished work. See also a publication list in pages 166–167.

Finally in Chapter 7, we conclude this thesis and show future directions of this work.

2

Background and Related Work

THIS chapter provides preliminary knowledge and concepts that are frequently referred in this dissertation. We first provides theoretical background of MRFs as a graphical model in Section 2.1. We then review discrete inference methods for MRFs based on graph cuts in Section 2.2 and then review methods based on message passing in Section 2.3. We also allocate Sections 2.4 and 2.5 for brief reviews of image segmentation and dense correspondence tasks that are addressed in this work.

2.1 Markov Random Field

This section provides a definition of Markov random fields in a more formal fashion as being a probabilistic model.

Let us first introduce some notations. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph where the node set $\mathcal{V} = \{1, 2, \cdots, N\}$ enumerates the indices of the nodes and the edge set $\mathcal{E} \subseteq (\mathcal{V} \times \mathcal{V})$ provides pairwise node neighbors. A subset $C \subseteq \mathcal{V}$ is a clique $C \in \mathcal{C}$ if all pairs of the nodes in C are neighbors. Upon the graph $\mathcal{G}$, we define a set of random variables $X = \{X_1, X_2, \cdots, X_N\}$, where each random variable X_i is sited at each of individual nodes $i \in \mathcal{V}$. For the random variables X, we define variables $x = \{x_1, x_2, \cdots, x_N\}$ that represent a particular configuration of X. Each variable x_i takes a value from a corresponding variable space $\mathcal{X}_i$. Here, $\mathcal{X}_i$ can be continuous or discrete. The space $\mathcal{X}$ for all the variables x is expressed as the product space of all individual variable spaces $\mathcal{X} = \Pi_i \mathcal{X}_i$. Let $P(X = x)$, or simply $P(x)$, be the probability that the random variables X take a particular configuration $x \in \mathcal{X}$.

2.1.1 Markov Property

The random variables X are then said to form a Markov random field, if they have the *local Markov property*, *i.e.*, each of individual variables x_i is *conditionally independent* of all the other variables given its neighbors $x_{\mathcal{N}(i)}$. This property can be formally expressed as

follows.

$$P(x_i|x_{\mathcal{V}\setminus\{i\}}) = P(x_i|x_{\mathcal{N}(i)}) \tag{2.1}$$

2.1.2 Hammersley-Clifford Theorem

Hammersley and Clifford in 1971 showed in an unpublished work that, under the Markov property condition, a strictly positive joint probability distribution $P(x)$ can be factorized in a Gibbs distribution form

$$P(x) = \frac{1}{Z} \exp\left\{ -\sum_{C\in\mathcal{C}} \phi_C(x_C) \right\}, \tag{2.2}$$

where $\phi_C(x_C)$ is a clique potential taking a set of variables x_C of nodes in a clique C, and Z is a normalization constant called the *partition function* defined as

$$Z = \sum_x \exp\left\{ -\sum_{C\in\mathcal{C}} \phi_C(x_C) \right\}. \tag{2.3}$$

This equivalence is known as the Hammersley-Clifford theorem, whose formal proof can be found, *e.g.*, in [Koller and Friedman, 2009].

By taking the negative log of the probability $P(x)$, we can also derive its energy function form as follows.

$$E(x) = \sum_{C\in\mathcal{C}} \phi_C(x_C) + \log Z \tag{2.4}$$

This energy function $E(x)$ is equivalent to $E(f)$ in Eq. (1.1) up to a constant $\log Z$, by considering that $\mathcal{V} \equiv \Omega$, $x_i \equiv f(i)$ and the clique set $\mathcal{C}$ here is defined to contain all of Ω, $\mathcal{N}$ and $\mathcal{C}$ in the unary, pairwise and higher-order terms in Eq. (1.1).

2.1.3 MAP-MRF Inference

Maximum a posteriori (MAP) inference on MRFs refers to the problem of finding a global maximizer x of $P(x)$ in Eq. (2.2), or equivalently, finding a minimizer x of $E(x)$ in Eq. (2.4). This is often called the *MAP-MRF inference*. In MAP-MRF inference, the partition function Z in $P(x)$ or $\log Z$ in $E(x)$ can be omitted from the objective function, because it is independent of x.

Besides MAP inference, there is another type of inference for MRFs known as the *maximum posterior marginal* (MPM) inference. In MAP inference, we are only interested in finding single maximizer/minimizer points x of $P(x)$ or $E(x)$. In MPM inference, on the other hand, we estimate the marginal distribution $P(x_i)$ for each variable x_i. While there are many methods for MAP-MRF inference such as graph cuts and max-product message passing algorithms that we will review later in this chapter, there are only a limited

number of methods for MPM-MRF inference such as sum-product message passing.

In this dissertation, we focus on MAP inference on various types of MRFs that appear in image segmentation and dense correspondence problems. For more discussions on MPM-MRF inference, the readers may refer to [Saito, 2016].

2.2 Inference Methods based on Graph Cuts

In this section, we review various inference methods for MRFs that use graph cuts [Kolmogorov and Zabin, 2004; Boykov and Kolmogorov, 2004] as a core discrete optimizer.

A series of techniques called "graph cuts" originally refer to algorithms for the *maximum-flow minimum-cut* problem in graph theory, which was studied in the field of operations research at least from 1960s [Hammer, 1965] and later applied to image restoration problems in 1980s [Greig *et al.*, 1986, 1989]. In the computer vision community, the graph cut method was introduced in 1990s [Ishikawa and Geiger, 1998a,b; Boykov *et al.*, 1998; Roy and Cox, 1998], and since then it was broadly applied to many computer vision and graphics tasks including but not limited to image restoration [Boykov *et al.*, 1998], image segmentation [Ishikawa and Geiger, 1998b; Boykov and Jolly, 2001; Rother *et al.*, 2004], stereo matching [Ishikawa and Geiger, 1998a; Roy and Cox, 1998; Kolmogorov and Zabih, 2001, 2002; Wei and Quan, 2005; Woodford *et al.*, 2008], optical flow [Lempitsky *et al.*, 2007, 2008; Xu *et al.*, 2012], photomontage [Agarwala *et al.*, 2004] and texture synthesis [Kwatra *et al.*, 2003]. Meanwhile, its theoretical properties and relationship to submodular function optimization were more studied and reported in 2000s [Ishikawa, 2003; Kolmogorov and Zabin, 2004; Kolmogorov and Rother, 2007].

In the following sections, we review basic and well-established inference methods using graph cuts for various types of pairwise MRFs, and later discuss techniques for higher-order MRFs.

2.2.1 S-T Max-Flow and Min-Cut for Binary Submodular MRFs

Most of the inference methods based on graph cuts are rooted to the optimization problem of energy functions based on binary submodular MRFs. Binary MRFs refer to MRFs with the binary label space $\mathcal{L} = \{0, 1\}$. Such binary labels are directly used in binary image processing such as image segmentation [Boykov and Jolly, 2001; Rother *et al.*, 2004; Ayed *et al.*, 2013; Kitamura *et al.*, 2016], video segmentation [Bai *et al.*, 2009; Pham *et al.*, 2010; Ayed *et al.*, 2015] and gray-image binarization [Gorelick *et al.*, 2014; Tang *et al.*, 2014]. Other than 2D MRFs on image grids, there is also an interesting application of binary MRFs in multi-view 3D mesh reconstruction [Vu *et al.*, 2012].

Submodularity is a mathematical concept describing a property of discrete functions [Fujishige, 2005]. It is usually defined for set functions $F(S) : 2^{\Omega} \rightarrow \mathbb{R}$ (*i.e.*,

functions F that take a set $S \subseteq \Omega$ as a variable), and is stated as follows.

Definition 2.1. ***Submodular function***. *A function $F(S)$ is sudmodular if and only if it satisfies the following inequity for any $S, T \subseteq \Omega$.*

$$F(S \cap T) + F(S \cup T) \leq F(S) + F(S) \tag{2.5}$$

There is another definition for submodular functions as follows.

Definition 2.2. ***Submodular function***. *A function $F(S)$ is sudmodular if and only if it satisfies the following inequity for any $S \subseteq \Omega$ and $i, j \in (\Omega \setminus S)$.*

$$F(S \cup \{i\}) - F(S) \geq F(S \cup \{i, j\}) - F(S \cup \{i\}) \tag{2.6}$$

The second definition is more intuitive, *i.e.*, functions are submodular if they have decreasing gain. When the inequalities in Eqs. (2.5) and (2.6) hold equal, $F(S)$ is said to be *modular*, which is similar to linearity in the continuous domain. Also, the negative of a submodular function is called a *supermodular function*. While submodular functions can be efficiently minimized in polynomial times [Schrijver, 2000; Iwata *et al.*, 2001], minimization of non-submodular functions is NP-hard.

Our energy function $E(f)$ with binary variables $f_p \in \{0, 1\}$ can be equivalently considered as a set function, by replacing the binary variables with a set variable S as

$$S = \{p \,|\, p \in \Omega, f_p = 1\}. \tag{2.7}$$

With this conversion and submodularity of Eq. (2.5), the submodularity for the energy functions $E(f)$ of binary pairwise MRFs can be derived as follows.

Theorem 2.1. ***Submodularity for binary pairwise MRFs*** *[Kolmogorov and Zabin, 2004]. The energy functions $E(f)$ of binary pairwise MRFs in Eq. (1.1) are submodular, if and only if all the pairwise terms ϕ_{pq} satisfy the following inequity:*

$$\phi_{pq}(0, 0) + \phi_{pq}(1, 1) \leq \phi_{pq}(1, 0) + \phi_{pq}(0, 1) \tag{2.8}$$

When the above condition is satisfied, the minimization of $E(f)$ can be replaced by the max-flow min-cut problem on a specific graph [Kolmogorov and Zabin, 2004], which can be *exactly* solved in a polynomial time.

More specifically, we make a *s-t graph* such as illustrated in Figure 2.1. Here, we have nodes (colored yellow in Figure 2.1) that correspond to individual sites $p \in \Omega$ in $E(f)$. These nodes are mutually connected by edges (black arrows in Figure 2.1) that represent neighborhood $(p, q) \in \mathcal{N}$ in $E(f)$. We additionally have two special nodes, the

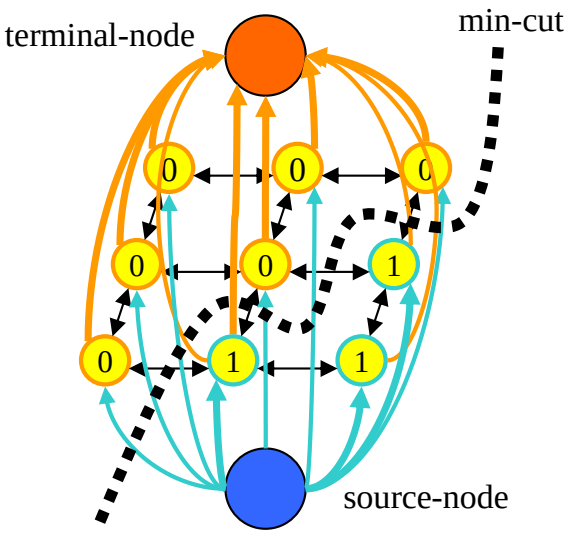


Figure 2.1 Illustration of s-t minimum cut for 2D grids. In graph cut optimization, we make a graph with nodes corresponding to sites p and additionally two special nodes called the source and terminal. Basically, the unary term costs are encoded into the weights of source-to-node and node-to-terminal edges (blue and orange edges), and the pairwise term costs are encoded into the weights of node-to-node edges (black edges). The optimal labeling is obtained as the minimum-cost cut that separates the nodes into the source and terminal sides.

source and terminal, which are connected to site nodes p. By following some conversion rules [Kolmogorov and Zabin, 2004], these edge weights are set to some non-negative values that represent unary and pairwise terms in $E(f)$. The conversion is intuitive when $\phi_{pq}(0,0) = \phi_{pq}(1,1) = 0$. In this case, the weights of bidirectional node-to-node edges are set to the pairwise term values $\phi_{pq}(1,0)$ and $\phi_{pq}(0,1)$, and the weights of source-to-node and node-to-terminal edges are set to the unary term values $\phi_p(0)$ and $\phi_p(1)$. By properly constructing this s-t graph, the minimum cut problem on the s-t graph (*i.e.*, the problem of finding a cutting surface that separates the graph nodes p into the source and terminal sides with the minimum edge-cutting cost) becomes equivalent to the minimization of the binary energy function $E(f)$. After obtaining a minimum cut (see the dashed line in Figure 2.1) using an max-flow min-cut solver [Boykov and Kolmogorov, 2004], a minimizer of $E(f)$ is then obtained as follows.

$$f_p \leftarrow \begin{cases} 1 & \text{if } p \text{ belongs to the source side by the minimum cut} \\ 0 & \text{if } p \text{ belongs to the terminal side by the minimum cut} \end{cases}. \tag{2.9}$$

In binary image tasks such as foreground-background segmentation, this binary labeling directly indicates a class labeling, *e.g.*, whether each node (or pixel) is foreground or background.

2.2.2 Roof Duality (QPBO) for Binary Non-submodular MRFs

When binary MRF energies are non-submodular, their minimization is NP-hard and thus cannot be optimally solved. This can be intuitively explained from the point of view that how s-t graphs are constructed (Section 2.2.1). The max-flow min-cut algorithms require all the graph edges be non-negative. However, when binary MRF energies are non-submodular, corresponding s-t graphs inevitably produce negative edge weights, which cannot be handled by the solvers. This is more discussed as *graph representability* in [Kolmogorov and Zabin, 2004]. For non-submodular energies, earlier work used some naive approximation (*e.g.*, Rother *et al.* [2005] truncated non-submodular terms to be submodular), but later Kolmogorov and Rother [2007] introduced the roof duality technique, which has become quit common nowadays.

The *roof duality* [Boros *et al.*, 1991; Hammer *et al.*, 1984] or often known as *quadratic pseudo boolean optimization* (QPBO) in the computer vision community, was introduced to the community by Kolmogorov and Rother [2007] as a technique to mitigate the issue of non-submodular energy optimization. With this technique, we can obtain a *partial solution* that tells us the optimal labels for a subset of the nodes. We briefly explain this technique below.

Roof Duality

The binary pairwise MRF energies can be equivalently expressed as quadratic functions with binary variables $x_p = f(p) \in \{0, 1\}$ as follows.

$$E(\mathbf{x}; U, V) = \sum_p \sum_q V_{pq} x_p x_q + \sum_p U_p x_p + c \tag{2.10}$$

Here, the coefficients V_{pq} and U_p are parameters that determine the function $E(f)$, and c is a constant. From the submodularity condition of Eq. (2.8), it can be shown that only the quadratic terms with positive coefficients produce non-submodular energies. Let us denote such positive coefficients as $V^+ = \{V_{pq} \,|\, V_{pq} > 0\}$.

Then, the roof duality introduces new binary variables $\bar{x}_p$, which have the following *flipped meaning* (but are not hard constrained to)

$$\bar{x}_p = 1 - x_p. \tag{2.11}$$

Using the two forms of variables $\mathbf{x}$ and $\bar{\mathbf{x}}$, an auxiliary function is introduced as follows.

$$A(\mathbf{x}, \bar{\mathbf{x}}; U, V) = E(\mathbf{x}; U, V \setminus V^+) + E(\mathbf{1} - \bar{\mathbf{x}}; U, V \setminus V^+) + B(\mathbf{x}, \bar{\mathbf{x}}; V^+) \tag{2.12}$$

Here, the first and second terms represent the quadratic functions in Eq. (2.10) in two ways without the positive quadratic terms. The last term accounts for the positive

quadratic terms $V^{+}_{pq}x_p x_q$ also in two ways as follows.

$$B(\mathbf{x}, \bar{\mathbf{x}}; V^{+}) = \sum_{p}\sum_{q} V^{+}_{pq}\Big[x_p(1-\bar{x}_q) + (1-\bar{x}_p)x_q\Big]. \tag{2.13}$$

Unlike the other two terms, this term does not equivalently express the original energies because x_p and $\bar{x}_p$ are treated as independent variables. Because of this relaxation, non-submodular terms are now converted to submodular terms. Therefore, the auxiliary energy function $A(\mathbf{x}, \bar{\mathbf{x}}; U, V)$ is submodular in $\mathbf{x}$ and $\bar{\mathbf{x}}$, and thus can be minimized via graph cuts as explained in Section 2.2.1.

By the roof duality technique, we obtain two independent solutions $\mathbf{x}$ and $\bar{\mathbf{x}}$. A *partial optimal solution* $\mathbf{z}$ is then obtained as follows.

$$z_p = \begin{cases} 1 & \text{if } x_p = 1 - \bar{x}_p = 1 \\ 0 & \text{if } x_p = 1 - \bar{x}_p = 0 \\ \textit{unknown} & \text{otherwise} \end{cases}. \tag{2.14}$$

This is equivalent to saying that if $\mathbf{x}$ and $\bar{\mathbf{x}}$ agree in their meanings, then their values are valid. The partial solution $\mathbf{z}$ tells us that there exists a global minimum $\mathbf{x}^*$ of the energy function in Eq. (2.10) such that $x^*_p = z_p$ for all labeled nodes p.

The unknown labels in $\mathbf{z}$ have to be somehow determined in practical use. However, there is a useful property called *persistency* that can mitigate this issue.

Theorem 2.2. ***Persistency*** *[Kolmogorov and Rother, 2007]. Let $\mathbf{z}$ be a partial optimal solution and $\mathbf{y}$ be an arbitrary solution. When we make a new solution $\mathbf{x}$ by*

$$x_p \leftarrow \begin{cases} z_p & \textit{if } z_p \neq \textit{ unknown} \\ y_p & \textit{otherwise} \end{cases}, \tag{2.15}$$

then the energy at $\mathbf{x}$ never increases from $\mathbf{y}$, i.e.*,*

$$E(\mathbf{x}) \leq E(\mathbf{y}). \tag{2.16}$$

Although the roof duality technique or QPBO offers an alleviation for the non-submodular energy optimization, it is effective only when the number of non-submodular terms is relatively small. Otherwise, many nodes will be left unlabeled [Gorelick *et al.*, 2014; Olsson *et al.*, 2013]. This is problematic especially in binary image processing. For such highly non-submodular functions, we may use message passing methods [Kolmogorov, 2006] or may approximate non-submodular pairwise terms by unary terms in an iterative manner [Gorelick *et al.*, 2014; Tang *et al.*, 2014; Taniai *et al.*, 2015]. This is more discussed in Chapter 3.

2.2.3 Expansion Moves for Discrete MRFs

Boykov *et al.* [2001] have proposed the expansion move algorithm for efficiently optimizing multi-labeling MRFs using graph cuts. The multi-valued labels are often used to represent pixel intensities in image restoration [Boykov *et al.*, 1998], or depth values (disparities) in stereo matching [Kolmogorov and Zabih, 2001], or object indices in segmentation [Delong *et al.*, 2012]. Algorithm 2.1 summarizes the expansion move algorithm.

Algorithm 2.1: EXPANSION MOVE ALGORITHM [Boykov *et al.*, 2001]

1 Define the discrete label space: $\mathcal{L} = \{\boldsymbol{\ell}^{(0)}, \boldsymbol{\ell}^{(1)}, ..., \boldsymbol{\ell}^{(K-1)}\}$;
2 Initialize the labeling: $f_p \leftarrow \boldsymbol{\ell}^{(0)}$;
3 **repeat**
4 **foreach** *label* $\alpha \in \mathcal{L}$ **do**
5 Solve a binary labeling problem: $f \leftarrow \operatorname{argmin} E(f' | f'_p \in \{f_p, \alpha\})$;
6 **end**
7 **until** *convergence*;

In this algorithm, the multi-labeling problem is reduced to a sequence of binary-labeling problems, where each node is allowed to either retain at its current value f_p or take a new proposal label $\alpha \in \mathcal{L}$. This sequential process is also illustrated in Figure 2.2.

At line 5 of Algorithm 2.1, these binary labeling problems are repeatedly solved using the s-t max-flow min-cut. Here, they can be *optimally* solved, if only the pairwise terms ϕ_{pq} in $E(f)$ meet the following submodularity of expansion moves [Boykov *et al.*, 2001; Kolmogorov and Rother, 2007]:

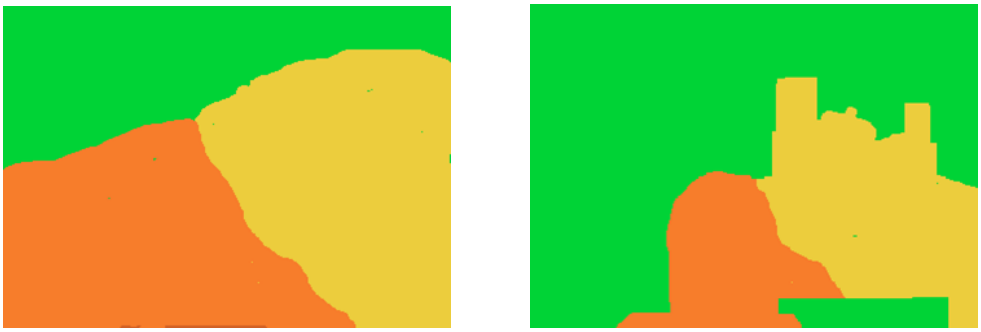
$$\phi_{pq}(\alpha, \alpha) + \phi_{pq}(\beta, \gamma) \leq \phi_{pq}(\beta, \alpha) + \phi_{pq}(\alpha, \gamma). \tag{2.17}$$

When this optimality condition is satisfied, then we can show that the energy $E(f)$ does not increase over the iterations.

When not satisfied, *i.e.*, when binary energies $E(f')$ at line 5 of Algorithm 2.1 are not submodular, QPBO (or the roof duality discussed in Section 2.2.2) is usually used to solve the binary labeling problems. In this case, unknown labels of the partial solution are filled by the current solution values. This way we can still show the monotonic convergence of the energy from the persistency property shown in Theorem 2.2.

2.2.4 Fusion Moves for Continuous MRFs

Lempitsky *et al.* [2010] have proposed the fusion move algorithm for efficiently optimizing continuous MRFs. The continuous-valued labels are often used to represent depth values (disparities) in stereo [Woodford *et al.*, 2009] or flow vectors in optical



(a) initial solution

(b) ●-expansion

(c) ●-expansion

(d) ●-expansion

(e) ●-expansion

(f) ●-expansion

(g) ●-expansion

(h) ●-expansion

Figure 2.2 Illustration of the expansion move algorithm [Boykov *et al.*, 2001]. Starting with (a) an initial solution, the expansion move algorithm visits each proposal label, and tries to update each pixel's currently assigned label by the proposal label. As shown through (a) to (h), this process repeatedly expands the regions where the proposal label is currently assigned (*e.g.*, the green region is expanded in (a) to (b), and in (g) to (h)). The images are cited from [Boykov *et al.*, 2007].

flow [Lempitsky *et al.*, 2008; Xu *et al.*, 2012]. The fusion move algorithm is generalization to the expansion move algorithm, and is summarized in Algorithm 2.2.

Algorithm 2.2: FUSION MOVE ALGORITHM [Lempitsky *et al.*, 2010]

1 Generate solution proposals: $\mathcal{P} = \{g^{(0)}, g^{(1)}, ..., g^{(K-1)}\}$;
2 Initialize the labeling: $f \leftarrow g^{(0)}$;
3 **repeat**
4 **foreach** *solution proposal* $g \in \mathcal{P}$ **do**
5 Solve a binary labeling problem: $f \leftarrow \operatorname{argmin} E(f' | f'_p \in \{f_p, g_p\})$;
6 **end**
7 **until** *convergence*;

In this algorithm, we first generate many *weak* solution proposals $\mathcal{P} = \{g^{(0)}, g^{(1)}, ..., g^{(K-1)}\}$. Those proposals are usually generated by external simple methods. The number K of proposals is usually an order of 100s [Lempitsky *et al.*, 2008; Woodford *et al.*, 2009]. The fusion move algorithm then sequentially fuses each proposal $g \in \mathcal{P}$ to the current solution f, by solving a binary labeling problem. In each binary labeling problem, a binary variable f'_p at each node either retains at its current value f_p or takes a new value g_p suggested by the proposal g for p.

Notice that if each proposal is provided spatially constant as $g_p^{(i)} = s^{(i)}$ for $^\forall p \in \Omega$, then the fusion move algorithm reduces to the expansion move algorithm in Algorithm 2.1. However, when proposals are given arbitrarily unlike expansion moves, we cannot guarantee that the energies produced by fusion moves are submodular. Therefore, it is essential in the fusion algorithm that we use QPBO (Section 2.2.2) for solving a series of binary labeling problems.

2.2.5 Multi-Model Fitting for Continuous MRFs

The fusion move algorithm is suitable for inferring a continuous-valued labeling where nodes are individually assigned (possibly) different values. However, we often require a more regularized representation where nodes are assigned and segmented by a small number of continuous and high-dimensional models. Figure 2.3 illustrates such an example where line models are assigned to the clutter of 2D points.

Isack and Boykov [2012] have proposed a multi-model fitting algorithm, which is a simple extension to the expansion move algorithm (Algorithm 2.1) and is summarized in Algorithm 2.3. This algorithm optimizes the energy function $E(f)$ by alternating between optimizing the assignment f and updating the models $\mathcal{L}$. Specifically, at line 3 of Algorithm 2.3, the energy is optimized for the labeling f while the models $\mathcal{L}$ are fixed. This is solved using the expansion move algorithm (Algorithm 2.1). At line 4, the models $\mathcal{L}$ are updated to reduce the energy $E(f)$ while the assignment f is fixed. This is done,

Algorithm 2.3: MULTI-MODEL FITTING ALGORITHM [Isack and Boykov, 2012]

1 Initialize models (can be done randomly): $\mathcal{L} \leftarrow \{\boldsymbol{\ell}^{(0)}, \boldsymbol{\ell}^{(1)}, ..., \boldsymbol{\ell}^{(K-1)}\}$;
2 **repeat**
3 | Assign models: $f \leftarrow \text{argmin } E(f|\mathcal{L})$;
4 | Update models: $\mathcal{L} \leftarrow \text{argmin } E(\mathcal{L}|f)$;
5 **until** *convergence*;

e.g., by fitting geometric models to inlier points using least squares in a case of geometric model fitting. Figure 2.3 shows the process of this algorithm in a line fitting problem.

For this kind of problem, Delong *et al.* [2012] have proposed a useful higher-order term that penalizes the number of models appearing in f to reduce the model complexity. This term behaves as a regularizer for over-fitting. Although this term originally forms a higher-order term, it can be efficiently optimized as submodular pairwise terms during the expansion move algorithm. We discuss this more in the next section.

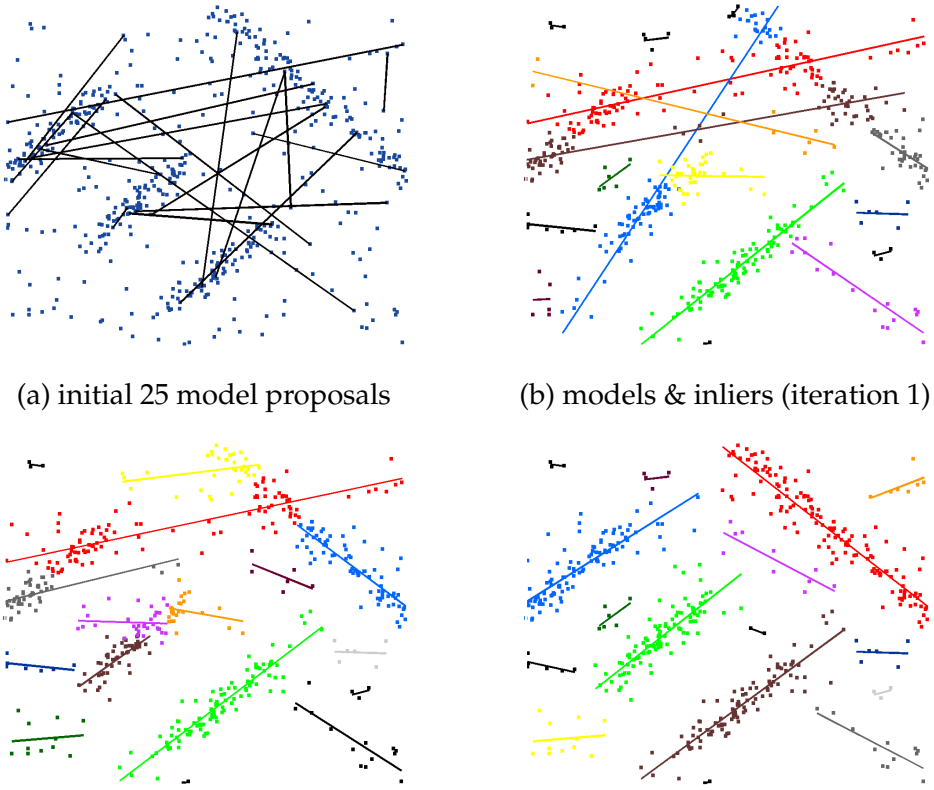
2.2.6 Order Reduction for Higher-order MRFs

We have thus far reviewed graph cut based optimization techniques for various pairwise or first-order MRFs. For higher-order MRFs, Kolmogorov and Zabin [2004] have shown a conversion from second to first-oder MRFs, and later Ishikawa [2011] has shown that any higher-order MRFs can be reduced to first-order MRFs. However, these techniques convert higher-order terms into new pairwise terms whose number exponentially increases as the order of the source higher-order terms increases. Also, these pairwise terms can be non-submodular requiring some treatment such as QPBO as used in [Woodford *et al.*, 2009; Ishikawa, 2011]. Therefore, these order reduction approaches are effective when the order of energies is relatively low. Otherwise, there will be many pairwise non-submodular terms, whose optimization is computationally expensive and difficult. We will study more general higher-order energy optimization in Chapter 3.

Meanwhile, there are some special forms of higher-order terms that can be more efficiently transformed and optimized. The $\mathcal{P}^n$ Potts model [Kohli *et al.*, 2007], robust $\mathcal{P}^n$ Potts model [Kohli *et al.*, 2009], label costs [Delong *et al.*, 2012] are all this kind of higher-oder terms. They also share a similar concept, *i.e.*, they softly enforce multiple nodes to take the same label. To illustrate this, let us consider the following higher-order term as a toy example that is defined for binary variables $x_p \in \{0, 1\}$ at nodes p in a clique Ω_c.

$$\phi_c(\mathbf{x}) = \begin{cases} 0 & \text{if } x_p = 0, {}^{\forall}p \in \Omega_c \\ \lambda & \text{otherwise} \end{cases}, \tag{2.18}$$

This is a simplified version of the $\mathcal{P}^n$ Potts model [Kohli *et al.*, 2007]. This term softly



(a) initial 25 model proposals

(b) models & inliers (iteration 1)

(c) models & inliers (iteration 2)

(d) models & inliers (iteration 5)

Figure 2.3 Multi-model fitting. The images are from [Isack and Boykov, 2012].

enforces x_p for all the nodes $p \in \Omega_c$ to be consistently 0. When minimizing this higher-oder term $\phi_c(\mathbf{x})$, we can derive an equivalent function $\phi'_c(\mathbf{x}, a)$ using an additional auxiliary variable $a \in \{0, 1\}$ as follows.

$$\min_{\mathbf{x}} \phi_c(\mathbf{x}) = \min_{\mathbf{x},a} \phi'_c(\mathbf{x}, a) \tag{2.19}$$

$$= \min_{\mathbf{x},a} \lambda \left[a + (1 - a) \sum_{p \in \Omega_c} x_p \right] \tag{2.20}$$

We can see the equivalence by confirming that, if $\mathbf{x} = \mathbf{0}$, then $\min_{\mathbf{x},a} \phi'_c(\mathbf{x}, a) = \phi'_c(\mathbf{0}, 0) = 0$, and if $\mathbf{x} \neq \mathbf{0}$ then $\min_{\mathbf{x},a} \phi'_c(\mathbf{x}, a) = \phi'_c(\mathbf{x}, 1) = \lambda$. Notice that this new representation $\phi'_c(\mathbf{x}, a)$ has only unary and submodular pairwise terms that can be optimally minimized via graph cuts. Generally, when higher-oder terms are submodular, there is the possibility that those terms can be reduced to pairwise submodular forms by introducing auxiliary variables (but not always possible).

The label costs [Delong *et al.*, 2012] extend this regional consistency term to the multi-label case, where the number of labels used in the resulting f is directly penalized to

reduce the model complexity. During inference using expansion moves, this term is dynamically converted to submodular pairwise terms using similar order reduction by adding auxiliary variables.

2.3 Inference Methods based on Message Passing

In addition to graph cut based methods, the message passing principle makes another body of discrete inference methods for MRFs. Although those methods are not in the primal scope of this dissertation, we in this section provide an overview recapping message passing algorithms and their variants.

As discussed in Section 2.1.3, message passing can be used for both MAP and MPM inference problems but in different forms of algorithms known as the max-product and sum-product message passing, respectively. As MAP inference being our primary focus, we here only explain algorithms based on max-product message passing.

2.3.1 Belief Propagation

First, we rewrite the energy function of pairwise MRFs in Eq. (1.1) by the following equivalent form.

$$E(f) = \sum_{p \in \Omega} \phi_p(f_p) + \sum_{p \in \Omega} \sum_{q \in \mathcal{N}(p)} \phi_{pq}(f_p, f_q). \tag{2.21}$$

The max-product belief propagation (MPBP) is an algorithm for estimating the minimizer f of $E(f)$. During the inference by MPBP, each node owns the disbelief (or the negative log of the likelihood) about its values f_p, which is defined at each node p and for $^\forall f_p \in \mathcal{L}$ as

$$B_p(f_p) = \phi_p(f_p) + \sum_{q \in \mathcal{N}(p)} M_{q \to p}(f_p). \tag{2.22}$$

Here, the disbelief is defined as the sum of the unary term $\phi_p(f_p)$ and messages $M_{q \to p}(f_p)$ from the neighbors. The message $M_{q \to p}(f_p)$ represents "node q's opinion of the [negative log of the] likelihood that node p takes value f_p" [Besse *et al.*, 2014], which is formally defined as

$$M_{q \to p}(f_p) = \min_{f_q} \phi_{pq}(f_p, f_q) + B_q(f_q) - M_{p \to q}(f_q). \tag{2.23}$$

This can be interpreted as "the belief at q, modified by the pairwise term, and neglecting p's contribution to q's belief" [Besse *et al.*, 2014].

The MPBP algorithm updates the beliefs B_p at individual nodes by applying the update rule of Eq. (2.22). As an important property of the MPBP algorithm, it converges to the optimal solution in two-pass (forward and backward) of updates when the graph structure has no cycle (*i.e.*, the graph is a tree). The MAP solution f_p^* at each node p is

then retrieved as

$$f_p^* = \underset{f_p \in \mathcal{L}}{\operatorname{argmin}}\, B_p(f_p). \tag{2.24}$$

When the graph has cycles, a BP algorithm called *loopy belief propagation* (LBP) [Murphy *et al.*, 1999] is known for approximately estimating the solution. In this case, the beliefs are usually updated in a sequential schedule visiting individual nodes in some ordering, where the messages are initialized by zero at the beginning. However, the general convergence property of LBP is unknown.

A naive implementation of the MPBP algorithm has a computation order of $O(|\Omega||\mathcal{L}|^2)$, since the message computation of Eq. (2.23) takes $O(|\mathcal{L}|)$ and that is repeated for $^\forall f_p \in \mathcal{L}$ and $^\forall p \in \Omega$ to compute beliefs $B_p(f_p)$ in one iteration. Felzenszwalb and Huttenlocher [2004, 2006] have proposed an efficient MPBP algorithm, which reduced the overall computation order to $O(|\Omega||\mathcal{L}|)$. This acceleration is achieved by computing the messages of Eq. (2.23) in $O(1)$ using distance transform techniques, which can be applied to certain forms of pairwise terms ϕ_{pq}.

2.3.2 Tree-Reweighted Message Passing

The tree-reweighted belief propagation (TRW-T) [Wainwright *et al.*, 2002] is a variant of belief propagation. The method makes use of the optimality property of MPBP for tree structures. Specifically, it decomposes the minimization problem of $E(f)$ on a loopy graph $\mathcal{G}$ into many subproblems of minimizing energies $E_i(f_i)$ on tree-shaped subgraphs $\mathcal{G}_i$ of $\mathcal{G}$. Here, the energy function $E_i(f_i)$ on each subgraph $\mathcal{G}_i$ is known to give a lower bound as $E_i(f_i) \leq E(f_i)$, and the minimization of $E_i(f_i)$ can be optimally solved by MPBP. The solutions of those subproblems are then aggregated in a way so as to maximize the lower bound, to obtain the final MAP solution f.

Although TRW-T has no convergence property, Kolmogorov [2006] has extended it to a sequential algorithm (TRW-S) that can monotonically reduce the energy $E(f)$ over the iterations. This TRW-S is known to be a standard message passing algorithm for MAP inference on general loopy graph structures.

2.3.3 Particle Belief Propagation and PatchMatch

The message passing algorithms that we have seen so far are all for discrete pairwise MRFs. For continuous MRF inference, particle belief propagation (PBP) [Ihler and McAllester, 2009] and its max-product algorithm (MP-PBP) [Kothapa *et al.*, 2011] have been proposed. In these algorithms, each node p has its own discrete solution space $\mathcal{L}_p$, which contains a certain number of candidate labels or *particles* $\mathcal{L}_p = \{\boldsymbol{\ell}_p^{(1)}, \boldsymbol{\ell}_p^{(2)}, \cdots, \boldsymbol{\ell}_p^{(K)}\}$. During the message passing inference, each node value f_p is assigned a label from the particles $\mathcal{L}_p$ while the particles are also updated by the Markov chain Monte Carlo

(MCMC) resampling.

Apart from PBP, Barnes *et al.* [2009] have proposed algorithms called *PatchMatch* for efficiently estimating nearest neighbor fields. The nearest neighbor field estimation is a variant of dense correspondence estimation between two images, where the nearest neighbors for the patches in the source image are searched in the target image. Therefore, it is equivalent to minimizing an energy function $E(f)$ with only the unary data term that evaluates patch dissimilarity. PatchMatch achieves highly efficient inference by using spatial propagation and randomization search schemes. It is summarized in Algorithm 2.4. At line 4 in Algorithm 2.4, a node p collects current labels of neighbors,

Algorithm 2.4: PATCHMATCH ALGORITHM [Barnes *et al.*, 2009]

```
1 Initialize f randomly ;
2 repeat
3     foreach nodes p ∈ Ω in a sequential (raster-scan) order do
4         f_p ← argmin φ_p(s) with s ∈ {f_p, f_q | q ∈ N(p)} ;
5         for m = 1 to M do
6             f_p ← argmin φ_p(s) with s ∈ {f_p, f_p + Δ/2^m} ;
7         end
8     end
9 until convergence;
```

and tries to update its current label f_p using those labels as candidates. Because nearby pixels have tendency to share similar solutions, this spatial propagation can effectively provide plausible solutions. At line 6, the current solution f_p is iteratively refined using randomization search. Here, the perturbation value Δ is exponentially decreased over the iterations. This algorithm was later generalized using k-nearest neighbor search in [Barnes *et al.*, 2010].

Later on, Besse *et al.* [2012, 2014] have pointed out a close relationship between the generalized PatchMatch [Barnes *et al.*, 2010] and the aforementioned MP-PBP [Kothapa *et al.*, 2011], leading to a unified algorithm named *PatchMatch belief propagation* (PMBP) for efficient inference of continuous pairwise MRFs.

These PBP based methods have an issue of producing duplicated particles in each $\mathcal{L}_p$. Pacheco *et al.* [2014] have proposed an optimized selection technique of particles. Their technique keeps diversity of particles when updating $\mathcal{L}_p$. This increases insensitivity to initializations.

2.3.4 Semi-Global Matching

The semi-global matching (SGM) algorithm [Hirschmuller, 2008] was originally presented as an efficient stereo matching method, but later Drory *et al.* [2014] have proven that its core optimization algorithm is a variant of message passing. As we will use this

algorithm in Chapter 6, we provide detailed descriptions of SGM below.

Let us consider the following energy function based on a pairwise MRF.

$$E(f) = \sum_{p \in \Omega} \phi_p(f_p) + c \sum_{(p,q) \in \mathcal{N}} \phi_{pq}(f_p, f_q). \tag{2.25}$$

Here, f takes an integer label space $\mathcal{L} = \{D_{\min}, \cdots, D_{\max}\}$, and the coefficient $c > 0$ is some constant explained later. While the unary term ϕ_p can be arbitrarily defined, the pairwise term is usually defined as

$$\phi_{pq}(f_p, f_q) = \begin{cases} 0 & \text{if } f_p = f_q \\ P_1 & \text{if } |f_p - f_q| = 1 \\ P_2 & \text{otherwise} \end{cases}. \tag{2.26}$$

Here, P_1 and P_2 ($0 < P_1 < P_2$) are smoothness penalties. This term enforces the label difference $|f_p - f_q|$ between neighbors to be at most one. Also, the neighbors $(p, q) \in \mathcal{N}$ in Eq. (2.25) are usually 4 or 8-connected adjacent pixel pairs on the image pixel grid.

To optimize such a 2D MRF energy function, Hirschmuller [2008] have proposed to decompose the 2D MRF into many 1D MRFs along 4 or 8 cardinal directions $\mathbf{r}$ and minimize them using dynamic programming. This is done by recursively updating the following cost arrays $L_{\mathbf{r}}(\mathbf{p}, d)$ along 1D scan lines in the directions $\mathbf{r}$ from the image boundaries.

$$L_{\mathbf{r}}(\mathbf{p}, d) = \phi_p(d) + \min_{d' \in D} \left[L_{\mathbf{r}}(\mathbf{q}, d') + \phi_{pq}(d, d') \right] - \min_{d' \in D} L_{\mathbf{r}}(\mathbf{q}, d'). \tag{2.27}$$

Here, $\mathbf{p}$ denotes the 2D pixel coordinate of p and $\mathbf{q} = \mathbf{p} - \mathbf{r}$ is its previous pixel coordinate in the direction $\mathbf{r}$. By introducing the following normalized scan-line costs

$$\bar{L}_{\mathbf{r}}(\mathbf{p}, d) = L_{\mathbf{r}}(\mathbf{p}, d) - \min_{d' \in D} L_{\mathbf{r}}(\mathbf{p}, d'), \tag{2.28}$$

the updating rule of Eq. (2.27) is simplified as follows.

$$L_{\mathbf{r}}(\mathbf{p}, d) = C_{\mathbf{p}}(d) + \min_{d' \in D} \left[\bar{L}_{\mathbf{r}}(\mathbf{q}, d') + \phi_{pq}(d, d') \right] \tag{2.29}$$

$$= C_{\mathbf{p}}(d) + \min\{\bar{L}_{\mathbf{r}}(\mathbf{q}, d), \bar{L}_{\mathbf{r}}(\mathbf{q}, d-1) + P_1, \bar{L}_{\mathbf{r}}(\mathbf{q}, d+1) + P_1, P_2\} \tag{2.30}$$

Then, the scan-line costs by the 8 directions are aggregated as

$$S(\mathbf{p}, d) = \sum_{\mathbf{r}} L_{\mathbf{r}}(\mathbf{p}, d), \tag{2.31}$$

from which the disparity estimate at each pixel p is retrieved as

$$f_p = \underset{d \in D}{\operatorname{argmin}}\, S(\mathbf{p}, d). \tag{2.32}$$

Drory *et al.* [2014] have proven that this algorithm is a variant of TRW-T [Wainwright *et al.*, 2002]. The coefficient c in Eq. (2.25) is a scaling factor that accounts for an over-weighting effect on the unary term during SGM ($c = 1/n$ where n is the number of scan-line directions $\mathbf{r}$).

Drory *et al.* [2014] have also proposed an uncertainty measure $\mathcal{U}$ that is computed as

$$\mathcal{U}(p) = \min_d \sum_{\mathbf{r}} L_{\mathbf{r}}(\mathbf{p}, d) - \sum_{\mathbf{r}} \min_d L_{\mathbf{r}}(\mathbf{p}, d). \tag{2.33}$$

$\mathcal{U}(p)$ is lower-bounded by 0, and becomes 0 when minimizers of individual scan-line costs agree. Since the first and second term in Eq. (2.33) are respectively computed in Eqs. (2.32) and (2.27), the computation of $\mathcal{U}(p)$ essentially does not require computational overhead.

Facciolo *et al.* [2015] have recently proposed an extension to the SGM optimizer named *more-global matching* (MGM). This algorithm incorporates messages not only from the previous pixel in the same scan-line but also from a neighboring pixel in the *previous scan-line*, which leads to more global inference than 1D MRFs.

2.4 Image Segmentation

In this section, we review seminal works in image segmentation and also explain basic problem settings of interactive segmentation and cosegmentation.

2.4.1 Interactive Segmentation

Boykov and Jolly [2001] have proposed an interactive image segmentation using graph cuts. In their method, a user inputs scribble-annotations ($\Omega_1, \Omega_0 \subset \Omega$) for foreground and background pixels in an image I, from which color models $\{\theta_1, \theta_0\}$ of foreground and background are learned. The energy function $E(f)$ is formulated based on a binary pairwise MRF in Eq. (1.1) where binary labels 1 and 0 indicate foreground and background, respectively. The unary data term is given as

$$\phi_p(f_p) = \begin{cases} -f_p C_\infty & \text{if } p \in \Omega_1 \\ f_p C_\infty & \text{if } p \in \Omega_0 \\ -f_p \log P(I_p|\theta_1) - (1 - f_p) \log P(I_p|\theta_1) & \text{otherwise} \end{cases}. \tag{2.34}$$

Here, C_∞ is the infinite cost that hard-constrains f_p to take 1 or 0 if p is in user-scribbles of foreground or background. Otherwise, f_p is softly constrained by the negative log of the likelihood of each pixel color I_p, based on the learned color probability distributions $P(\cdot|\theta_0)$ and $P(\cdot|\theta_1)$ expressed as histograms. The pairwise smoothness term is defined based on the Potts model as

$$\phi_{pq} = \lambda e^{-|I_p - I_q|^2/2\sigma^2} |f_p - f_q|, \tag{2.35}$$

which essentially regularizes the length of segmentation boundaries. Here, the term is weighted to induce segmentation boundaries to occur at image edges. The contrast-sensitivity parameter σ^2 is learned automatically as the expectation value of $|I_p - I_q|^2$ over all neighbor pairs $(p, q) \in \mathcal{N}$ based on a Gaussian image noise assumption. The energy is binary submodular, thus it can be optimally minimized via graph cuts.

Rother *et al.* [2004] have proposed an interactive segmentation named *GrabCut* that uses a bounding box as user's input. The energy formulation is essentially the same with [Boykov and Jolly, 2001], except that the color models are expressed by the Gaussian mixture models and initialized using the bounding box. They also have proposed an iterative algorithm where the segmentation f and the color models $\{\theta_1, \theta_0\}$ are alternately updated. This procedure has been shown to alternately minimize the joint energy $E(f, \theta_1, \theta_0)$, whose optimization strategies have been more studied in [Vicente *et al.*, 2009; Tang *et al.*, 2014]. GrabCut has become a standard segmentation method and is broadly used, *e.g.*, as a built-in tool of Microsoft Office products.

Although the boundary-length regularization term in Eq. (2.35) is commonly used in segmentation because of its nice submodularity property, it suffers from a *short-boundary bias* especially when segmenting thin structures [Jegelka and Bilmes, 2011]. For this issue, Jegelka and Bilmes [2011] have proposed a smoothness term using truncated boundary-length penalties that successfully overcome the issue. Combined with the standard color likelihood term, the energy function imposes non-submodular higher-order inference. An approximate optimization algorithm by [Jegelka and Bilmes, 2011] uses a iterative procedure motivated from submodular function optimization. Kohli *et al.* [2013] have later shown that its exact inference is computationally tractable.

2.4.2 Cosegmentation

Rother *et al.* [2006] have proposed an automatic image segmentation problem named *cosegmentation* that does not require any user interaction. Given two images, cosegmentation is originally cast as a problem of simultaneously finding object regions in each image that show the same or similar objects.

They also have proposed a distribution matching approach for segmentation, which enforces the appearance consistencies of corresponding objects by minimizing the dis-

tance between feature distributions in the regions. This imposes high-order constraints. Rother *et al.* [2006] have proposed to optimize the higher-order MRF by iteratively minimizing some approximation functions via graph cuts.

Since [Rother *et al.*, 2006], cosegmentation problems have been applied in broader settings including a case of multiple input images [Batra *et al.*, 2010; Joulin *et al.*, 2010; Kim *et al.*, 2011; Vicente *et al.*, 2011; Joulin *et al.*, 2012; Kowdle *et al.*, 2012] and also a case of multi-class segmentation [Kim *et al.*, 2011; Joulin *et al.*, 2012; Kowdle *et al.*, 2012].

2.5 Dense Correspondence

In this section, we review various types of image dense correspondence problems in computer vision. We first discuss the binocular stereo vision problem in Section 2.5.1, which is probably the most fundamental among all dense correspondence problems and is also considered one of the most classic problems in computer vision. We here provide a comprehensive and in-depth overview of binocular stereo vision, because some of its subjects often commonly appear in other dense correspondence problems. Later in Sections 2.5.2–2.5.4, we briefly review more generalized problems such as optical flow and scene flow.

2.5.1 Binocular Stereo Vision

Stereo vision or stereo matching is a technique for estimating the 3D geometries of static scenes using multiple images taken from different view points. Of various settings of stereo vision, the binocular stereo problem that assumes two input images taken from a stereo camera rig, makes a fundamental building block in stereo vision. Figure 2.4 illustrates the typical settings of binocular stereo vision. Here, a depth value z at a 2D image coordinate $\mathbf{p}$ on the left view image can be triangulated by

$$z = fB/d \tag{2.36}$$

when $\mathbf{p}$'s corresponding point $\mathbf{p}' = \mathbf{p} - (d, 0)^T$ on the right image as well as the camera's focal length f and baseline B are known. Therefore, the objective of binocular stereo vision amounts to estimating the horizontal shift d or so-called the *disparity* at each pixel. In the rest of this section, we express the mapping f of $E(f)$ in Eq. (1.1) by the disparity map d where $d_p = f_p$.

Generally in stereo, for each pixel $\mathbf{p}$ in the source (reference) image, its corresponding point $\mathbf{p}'$ in the target image is constrained to lie on a 1D line. This is known as *epipolar constraints* or *epipolar lines*. The settings shown in Figure 2.4 are a special case known as *rectified binocular stereo*, where the epipolar lines are horizontal, thus $\mathbf{p}$ and $\mathbf{p}'$ have the same vertical image coordinate. It is also known that any image pairs can be transformed

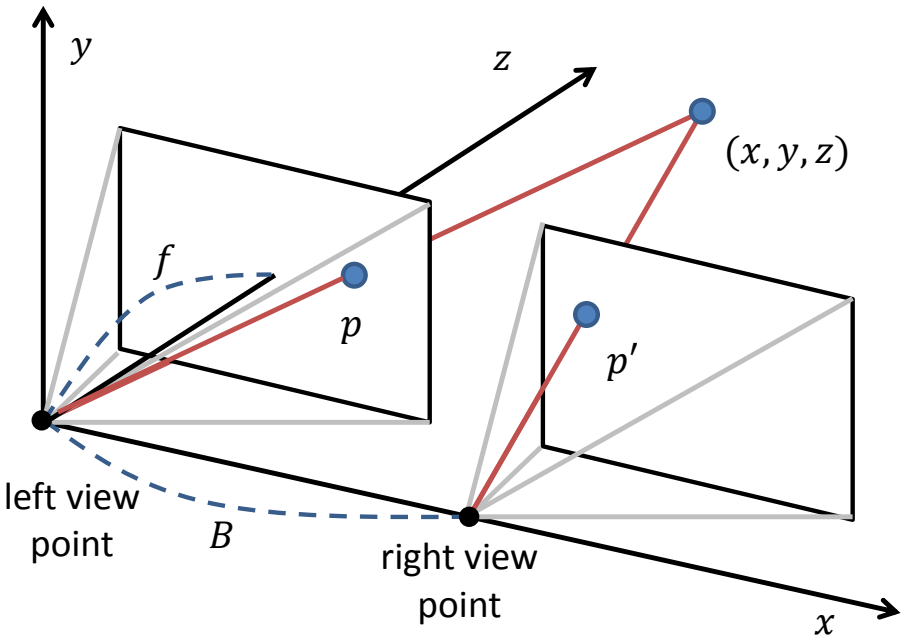


Figure 2.4 Basic setups of rectified binocular stereo vision. Two front-parallel cameras with the focal length f are placed at $(0, 0, 0)$ and $(B, 0, 0)$. A 3D point (x, y, z) is projected onto the left and right image planes at pixel coordinates p and p', respectively.

into this rectified settings using rectification techniques [Hartley and Zisserman, 2004; Monasse, 2011], given intrinsic and extrinsic calibration parameters of the two cameras.

Stereo vision problems consist of several subjects —camera calibration, photo-consistency measure, cost aggregation, regularization, optimization, occlusion—, each of which has been often individually studied as a fundamental problem of stereo. These problems are also often posed in other dense correspondence problems. In the followings, we overview these subjects while focusing on the binocular stereo problem. More details and theories of multiple view stereo can be found in [Hartley and Zisserman, 2004].

Camera Calibration

Camera calibration refers to recovering *extrinsic* and *intrinsic parameters* of cameras. Extrinsic parameters define the pose of a camera by a 3D translation vector $\mathbf{t}$ and rotation matrix R. They determine the geometric transformation from the 3D world coordinates to local camera coordinates. Extrinsic parameters can be estimated from images using structure-from-motion techniques [Hartley and Zisserman, 2004]. Intrinsic parameters such as focal length f, principal point, and lens distortion coefficients define the perspective projection transformation from 3D camera coordinates to 2D image coordinates.

In binocular stereo vision, we typically assume that image pairs have been already rectified by known camera parameters. In this case, the primary goal of stereo vision

becomes estimating disparities at individual pixels, which can be done without any information of camera calibration. Performances of stereo algorithm are often evaluated by errors in disparity. For example, the Middlebury stereo benchmark [Scharstein and Szeliski, 2002] uses disparity error rates with some thresholds.

Photo-Consistency Measure

The design of accurate and robust photo-consistency measures (or matching costs) is fundamental in stereo vision and other correspondence estimation. This corresponds to the design of the unary data term $\phi_p(\cdot)$ in MRFs of Eq. (1.1). The most naive measure is SAD (sum of absolute difference). Birchfield and Tomasi [1998] have also proposed a pixel dissimilarity measure that is insensitive to image sampling. However, direct comparison of image intensities is not robust to illumination changes.

As more robust measures, normalized cross correlation (NCC) between image patches can be used but its computation is relatively expensive. Zabih and Woodfill [1994] have proposed the CENSUS transform that encodes an image patch into a binary feature vector by comparing intensity differences of pixel pairs in the patch. The distance between CENSUS features can be efficiently computed as the Hamming distance using bit operations. Also, image gradient-based measures are often used by combining with intensity-based measures linearly [Klaus *et al.*, 2006] or selectively [Xu *et al.*, 2012]. See also [Hirschmuller and Scharstein, 2007] for evaluations on some of photo-consistency measures in stereo.

More recently, matching costs based on local features learned by convolutional neural networks (CNNs) have been proposed [Zbontar and LeCun, 2015, 2016]. In this approach, feature vectors are densely extracted at individual pixels by converting image patches through a CNN. Distances of those feature vectors are then evaluated as matching costs, using L_2 norm or through a subsequent neural network.

Cost Aggregation

Patch-based photo-consistency measures are more robust and reliable than those measured by single pixel pairs. The aforementioned NCC, CENSUS and local feature based matching costs are considered as patch-based measures. Patch-based matching costs can be also computed by aggregating pixelwise matching costs in local support windows, which is called *cost aggregation* in the literature.

Patch-based cost aggregation relies on an implicit assumption of constant disparity. That is, all pixels in a support windows have the same disparity with that of the center pixel. As discussed in [Bleyer *et al.*, 2011], this assumption is unlikely to hold for two reasons: (1) When pixels in the window lie on a different surface than the center pixel; (2) When the local region is not front-parallel and highly slanted. The first issue causes the

flattening effect, which refers to an artifact of extended object surfaces at depth boundaries. The second issue causes the *front-parallel bias*, which produces an artifact that disparity maps look like a staircase at slanted surfaces. These issues are traded off by matching reliability. That is, when we use larger windows for higher reliability, the constant disparity assumption is more unlikely to hold and we will more suffer from the artifacts.

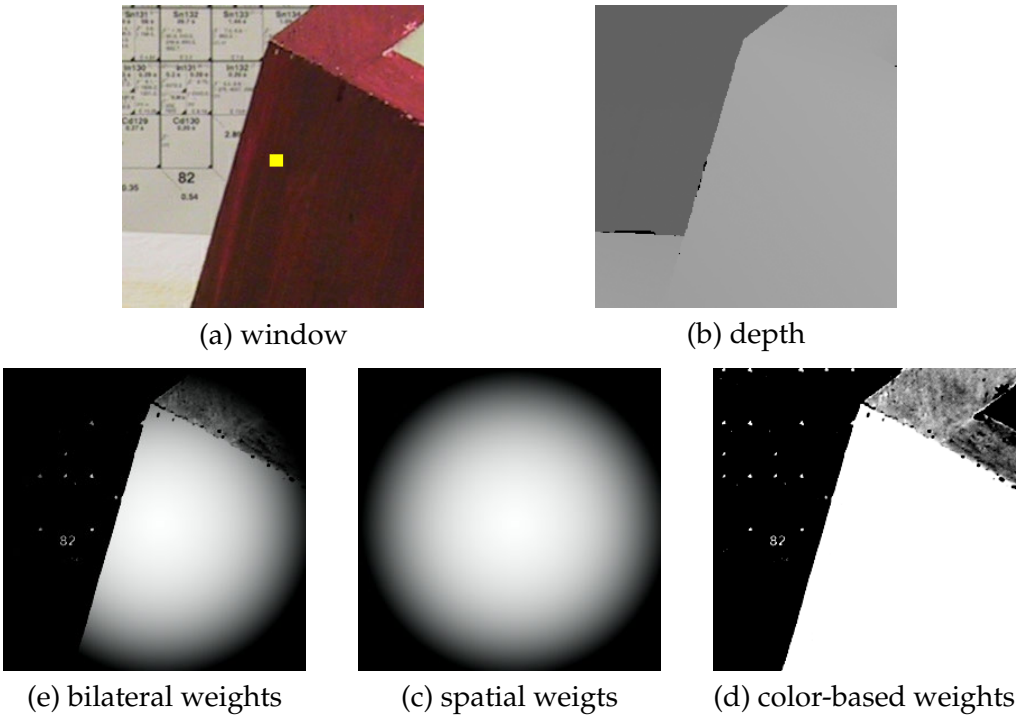
The first issue is well handled by incorporating adaptive window approaches [Hosni *et al.*, 2012] that use *soft support windows* for aggregation. The window weights account for likelihoods of support pixels lying on the same surface with the center pixel. The concept is illustrated in Figure 2.5. The cost aggregation using adaptive weights can be seen as applying edge-preserving filtering on cost maps of pixelwise matching costs. For this purpose, Yoon and Kweon [2005] first used the joint bilateral filtering [Tomasi and Manduchi, 1998], and later Rhemann *et al.* [2011] used the guided image filtering [He *et al.*, 2013]. Hence, development of fast and accurate edge-aware filtering [He *et al.*, 2010, 2013; Lu *et al.*, 2012] is also an important subject in stereo.

For the second case, Bleyer *et al.* [2011] have recently shown that, by estimating the local 3D disparity plane for each patch, we can effectively address the front-parallel bias by large matching windows. In this approach, however, the label space becomes a continuous 3D space. Bleyer *et al.* [2011] have proposed to use PatchMatch [Barnes *et al.*, 2009] for densely estimating 3D planes, but it is not applicable to pairwise MRF inference.

Regularization

Estimating disparities by relying on only photo-consistency cues is ambiguous and noisy. Smoothness regularization is therefore often employed, which corresponds to adding the pairwise smoothness term $\phi_{pq}(d_p, d_q)$ to the unary data term $\phi_p(d_p)$ in Eq. (1.1) forming a pairwise MRF. In stereo the methods having the pairwise smoothness term are called *global methods*, whereas those using only the unary data term are often called *local methods*.

The simplest and broadly used regularizer is the picewise-constant smoothness term $|d_p - d_q|$ that penalizes disparity deviations between neighboring pixels. This is often truncated as $\min\{\tau, |d_p - d_q|\}$ to allow depth discontinuities at object boundaries. However, this model has the front-parallel bias that causes a staircase artifact at slanted surfaces. Woodford *et al.* [2009] have proposed a second-order smoothness term which takes a higher-order form of $|d_{q1} - 2d_p + d_{q2}|$ using a tuple of three consecutive pixels (q_1, p, q_2). However, it requires complicated optimization due to higher-order energies. Olsson *et al.* [2013] have proposed a second-order smoothness term using a plane-based formulation, which can be expressed in a pairwise form.



(a) window (b) depth

(e) bilateral weights (c) spatial weigts (d) color-based weights

Figure 2.5 Concept of adaptive support windows. (a) The support window may contain pixels that lie on a different surface than the center (yellow) pixel, as shown in (b) the depth map. The bilateral adaptive windows [Yoon and Kweon, 2005] use window weights shown in (e) by combining (c) spatial weights and (d) color-based weights. In the context of cost-volume filtering, the weights can be seen as filter kernels.

Optimization

Optimization in stereo refers to minimizing the energy function $E(f)$. For local methods that have only the unary term, this is done simply in a winner-take-all manner. Otherwise, we need to infer pairwise or higher-oder MRFs using methods such we have seen in Sections 2.2 and 2.3.

For standard global stereo methods (*i.e.*, those using pairwise MRFs with a discrete label space of integer disparities), their optimization is well established [Szeliski *et al.*, 2008] and can be done by directly applying discrete optimizers such as BP and the expansion move algorithm using graph cuts. However, when we infer continuous-valued disparities $d_p \in [D_{\text{min}}, D_{\text{max}}]$ or disparity planes $d_p = a_p u + b_p v + c_p$ that are parameterized by $(a_p, b_p, c_p) \in \mathbb{R}^3$, we need discrete-continuous optimization that is not trivial even for local methods. In Chapter 4, we address the optimization problem of a global stereo method that estimates the disparity plane at every pixel.

Occlusion

If a pixel in one image is invisible in other target images (because the corresponding point is occluded by an object surface or because it is out of field-of-view), we cannot estimate the correspondence in principle. This problem is known as occlusion in stereo. These occluded pixels cause artifacts without proper handling.

Occlusion is usually handled either in a post processing [Rhemann *et al.*, 2011] or during inference by incorporating occlusion into the model [Kolmogorov and Zabih, 2001, 2002; Wei and Quan, 2005; Woodford *et al.*, 2008]. The post-processing is usually done by 1) estimating disparity maps on both images, 2) filtering out outliers using consistency check for the two disparity maps, and 3) filling the holes using *e.g.* weighted median filtering [Rhemann *et al.*, 2011; Zhang *et al.*, 2014]. In the latter approach, occluded pixels are geometrically reasoned and matching costs at those pixels are ignored. This introduces higher-order terms that make the inference complicated [Woodford *et al.*, 2008].

2.5.2 Optical Flow

Optical flow generalizes the binocular stereo problem by assuming that the two images are taken at different times from possibly different view points. This gives raise to many difficulties in optical flow that are discussed below.

Dynamic Object Motions

Objects shown in images can dynamically move between the two time steps. While rigid motions are constrained by the epipolar geometry due to the global camera motion and the depths of object points, dynamic non-rigid motions are quite arbitrary and must be explicitly estimated. Optical flow motions are generally expressed as 2D translation vectors or flow vectors, which causes the following issue of broader label spaces.

Broader Search Space

While correspondence points are efficiently expressed by 1D disparity values in stereo due to the epipolar constraints, we have to search 2D displacement or flow vectors at individual pixels in optical flow. When motions between the images are large, naive discrete inference approaches become computationally very expensive.

Because of this, early works on optical flow rely on continuous inference [Horn and Schunck, 1981; Lucas and Kanade, 1981]. However, these methods tend to converge into bad local solutions due to highly non-convex energy functions. Also, they are sensitive to initializations making them difficult to handle large motions.

Lempitsky *et al.* [2008] have proposed a discrete-continuous approach to optical flow using the fusion move algorithm [Lempitsky *et al.*, 2010] (Section 2.2.4). The proposals were generated using continuous methods of [Horn and Schunck, 1981; Lucas and Kanade, 1981] and the fused solution was further refined using continuous optimization based on gradient descent. Xu *et al.* [2012] have also proposed a fusion-based method for large motion optical flow, which uses proposals generated by feature-based tracking as well as continuous solvers in a coarse-to-fine framework. Then, as another line of inference approaches, PatchMatch [Barnes *et al.*, 2009, 2010] (Section 2.3.3) and its variants [Barnes *et al.*, 2009, 2010; Besse *et al.*, 2012, 2014; Lu *et al.*, 2013] have emerged.

These discrete inference approaches essentially subsample the full 2D flow space using proposals or random sampling. Very recently, Chen and Koltun [2016] have proposed FullFlow, which solves combinatorial optimization on a standard pairwise MRF but using the full search space without any subsampling. They have shown that such expensive computations are still tractable using TRW-S [Kolmogorov, 2006] with some acceleration techniques.

2.5.3 General Dense Correspondence

Liu *et al.* [2011] have proposed a more general optical flow problem where we estimate correspondence between images taken from different scenes but showing similar objects. Figure 2.6 shows examples of correspondence estimates for image pairs from different scenes. General dense correspondence can be used for non-parametric scene parsing and label transferring tasks. In their method named *SIFT Flow*, they use densely extracted SIFT features [Lowe, 2004] to robustly evaluate matching consistencies. Their pairwise MRF model is inferred using BP in a coase-to-fine framework of [Felzenszwalb and Huttenlocher, 2006] for dealing with large motions.

Since the seminal work of Liu *et al.* [2011], many follow-up studies have appeared [Hassner *et al.*, 2012; Kim *et al.*, 2013; Hur *et al.*, 2015; Yang *et al.*, 2014]. Among them, Hassner *et al.* [2012] have dealt with scaling differences. Kim *et al.* [2013] have proposed a hierarchical regularization using a spatial pyramid, which has been further extended by Hur *et al.* [2015]. Yang *et al.* [2014] have proposed a simplified local method that uses cost-volume filtering instead of pairwise smoothness terms for enforcing smoothness. In Chapter 5, we also address a similar problem but estimate dense correspondence jointly with cosegmentation.

2.5.4 Stereo Scene Flow

3D scene flow originally refers to a task of estimating 3D flow in the world [Vedula *et al.*, 1999, 2005], which describes the motions of 3D object points between two time steps. While optical flow estimates 2D flow motions on the 2D image domain, scene flow

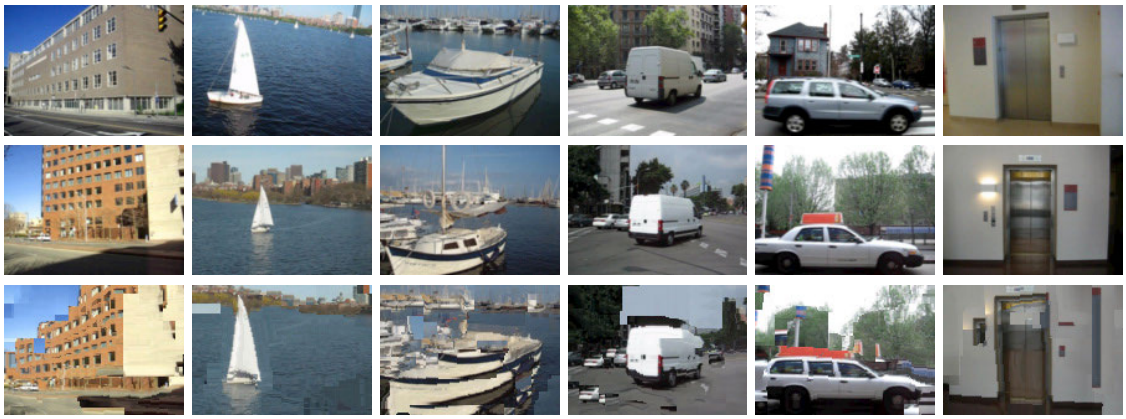


Figure 2.6 SIFT Flow. First and second rows: input image pairs. Third row: warped second images that are aligned onto the first images. The images are cited from [Liu *et al.*, 2011].

extends it by estimating 3D motions in the 3D world.

Stereo scene flow [Huguet and Devernay, 2007; Li and Sclaroff, 2008; Wedel *et al.*, 2011] is another representation of the same task, which estimates dense disparity and optical flow maps for sequences of stereo image pairs. When camera intrinsics and camera ego-motion are known, we can recover 3D flow from sequences of disparity and optical flow maps.

Earlier works on stereo scene flow assumed static cameras or cameras moving in relatively simple scenes. The task has then shifted to deal with more challenging and realistic scenes with many moving objects captured by a moving stereo camera [Čech *et al.*, 2011; Vogel *et al.*, 2011, 2013, 2014, 2015; Menze and Geiger, 2015; Lv *et al.*, 2016], while introducing better regularization, models, optimization methods, etc. Menze and Geiger [2015] also for the first time evaluated scene flow algorithms on realistic KITTI benchmark – earlier versions of KITTI did not have ground truth flow for moving objects. In Chapter 6, we also address the stereo scene flow problem jointly with motion segmentation and camera ego-motion estimation.

3

Binary MRF Inference for Segmentation and Low Level Vision

BINARY labeling problems naturally appear in many computer vision tasks such as image segmentation and binarization of gray images, and even play a central role in more general multi-valued labeling problems. Even though the variable space is as small as binary, non-submodularity and higher-orderness of the energy functions pose fundamental difficulties, making the inference problems NP-hard. Yet, the use of such sophisticated models is often required to overcome undesired biases or achieve greater performances that are unattainable by pairwise and submodular MRF formulations. In this chapter, we study fundamental inference problems on binary MRFs with non-submodular and higher-order terms, and develop a new theoretical insight that unifies several existing optimization methods.

3.1 Introduction

Many low-level vision problems such as image segmentation, binarization, denoising, and tracking are often formulated as binary energy minimization [Boykov and Kolmogorov, 2004; Rother *et al.*, 2006; Ayed *et al.*, 2010; Tang *et al.*, 2014; El-Zehiry and Grady, 2010]. For example, in image segmentation, the use of Markov random field formulations [Geman and Geman, 1984] and graph cuts [Kolmogorov and Zabin, 2004; Boykov and Kolmogorov, 2004] has been becoming one of primary approaches [Boykov and Jolly, 2001; Rother *et al.*, 2004; Price *et al.*, 2010; Tang *et al.*, 2014; Gorelick *et al.*, 2014; Ayed *et al.*, 2013; Gorelick *et al.*, 2013, 2012; Taniai *et al.*, 2012; Pham *et al.*, 2011; Ayed *et al.*, 2010; Rother *et al.*, 2006]. In this approach, the energy function is typically formulated as

$$E(S) = R(S) + Q(S), \tag{3.1}$$

where $R(S)$ describes appearance consistencies between resulting segments S and given information about target regions, and $Q(S)$ enforces smoothness on segment-boundaries.

The form of $R(S)$ is often restricted to simple linear (*i.e.*, pixelwise unary) forms [Boykov *et al.*, 2001; Rother *et al.*, 2004; Price *et al.*, 2010] because graph cuts allow globally optimal inference only for unary and submodular pairwise forms of energies [Kolmogorov and Zabin, 2004]. However, recent studies [Tang *et al.*, 2014; Ayed *et al.*, 2013; Gorelick *et al.*, 2013, 2012; Taniai *et al.*, 2012; Pham *et al.*, 2011; Ayed *et al.*, 2010; Rother *et al.*, 2006] have shown that the use of higher-order information (or non-linear terms) can yield outstanding improvements over conventional pixelwise consistency measures.

In general, higher-order terms involve difficult optimization problems. Recent promising approaches try reducing energies by iteratively minimizing either first-order approximations (gradient descent approach) [Gorelick *et al.*, 2013, 2012] or upper-bounds (bound optimization approach) [Tang *et al.*, 2014; Ayed *et al.*, 2013; Taniai *et al.*, 2012; Pham *et al.*, 2011; Ayed *et al.*, 2010] of non-linear functions using graph cuts. The bound optimization approach has some advantages over the gradient descent approach [Ayed *et al.*, 2013]: It requires no parameters (*e.g.*, step-size) and never worsens the solutions during iterations. But we must in turn derive appropriate bounds for individual functions. A notable work is *auxiliary cuts* (AC) by Ayed *et al.* [2013], where they derive general bounds for broad classes of non-linear functionals for segmentation. However, the bounds derived in [Ayed *et al.*, 2013] are formulated to successively reduce target regions; thus the resulting segments are restricted within initial segments. Such a property actually limits the applications and accuracy of the method.

In order to derive more accurate and useful bounds, we revisit a submodular-supermodular procedure (SSP) [Narasimhan and Bilmes, 2005], a general bound optimization scheme for supermodular functions. We then propose a bound optimization method as generalization of SSP. Unlike SSP, our method can be used even for non-supermodular functions; and unlike AC, it allows bi-directional optimization (see Figure 3.1 for an illustration in segmentation) and can produce more accurate approximation bounds. We further show that our method can be seen as generalization of AC and some state-of-the-art method [Gorelick *et al.*, 2014] for pairwise non-submodular functions.

When compared with gradient descent methods (*e.g.*, FTR [Gorelick *et al.*, 2013] and LSA-TR [Gorelick *et al.*, 2014] that are based on the trust region principle), our method has advantages of faster convergence and better ability to avoid bad local minimums. It is intuitively illustrated in Figure 3.2, where we compare our approach and the gradient-descent approaches. Since the gradient-based methods use Taylor-based local approximations, they tend to converge into weak local minimums and their convergence is also slow. In contrast, as we will see later, our method produces piecewise linear approximation bounds that are updated in a coarse-to-fine manner over iterations. The proposed bounds more globally approximate the energy functions, leading to faster convergence. Also, the coarse-to-fine scheme can avoid implausible weak solutions. Such a tendency will be verified in experiments.

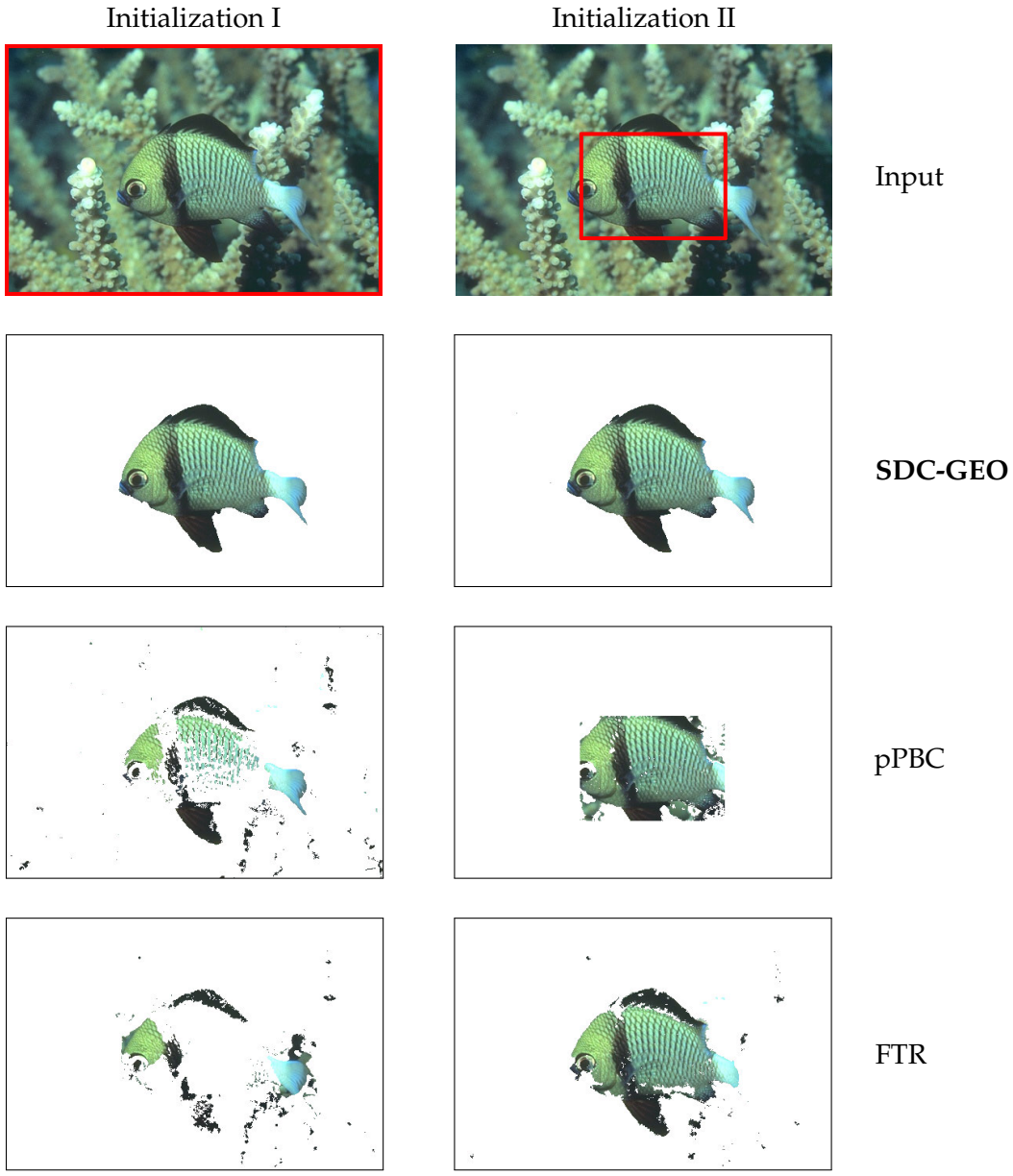
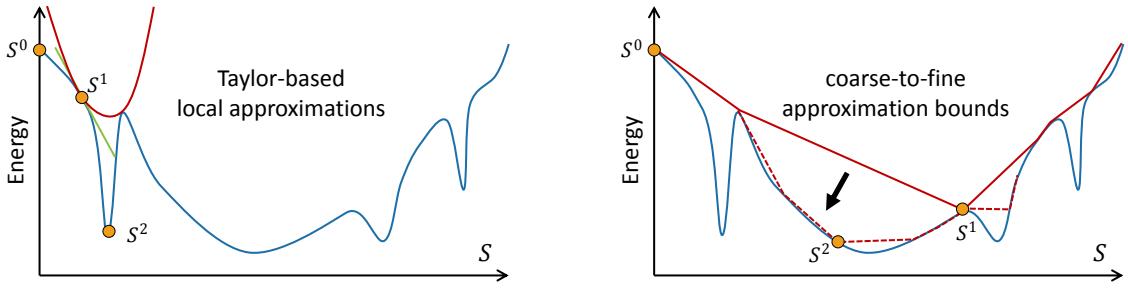


Figure 3.1 Matching foreground color distribution using the proposed **SDC-GEO**, parametric pseudo bound cuts (pPBC) [Tang *et al.*, 2014], and fast trust region (FTR) [Gorelick *et al.*, 2013] with two types of initialization. pPBC can only successively reduce the initial segment, while our method allows arbitrary directions of optimization and is thus robust to initialization. (L_2 distance for 64^3 bins of RGB histograms are used)



(a) Gradient-based approximations

(b) Proposed piecewise linear approximations

Figure 3.2 Intuitive illustration of the proposed piecewise linear approximations in comparison with the gradient-based approximations. The blue and red lines, and orange points show the energy function $E(S)$, its approximations, and solutions S^0, S^1, S^2 obtained at each iteration, respectively. (a) In the gradient descent approach (*e.g.*, FTR [Gorelick *et al.*, 2013] and LSA-TR [Gorelick *et al.*, 2014]), the energy function is approximated by local gradients (green) with *trust region constraints* (or step-size). The overall approximation at S^1 is depicted as the red curve, and its minimization results in falling in a bad local minimum S^2. (b) In our approach, the energy function is approximated by piecewise linear upper-bound functions. Here, the approximation bounds are updated in a coarse-to-fine manner (*i.e.*, solid to dashed red lines) in order to avoid bad local minimums.

In this study, we make the following contributions:

- We propose an optimization method for broad classes of higher-order and pairwise non-submodular functions that allows arbitrary directions of convergence and outperforms the state-of-the-art [Gorelick *et al.*, 2014, 2013; Tang *et al.*, 2014; Ayed *et al.*, 2013].
- Our method generalizes existing optimization methods including from early [Narasimhan and Bilmes, 2005] to state-of-the-art [Gorelick *et al.*, 2014; Tang *et al.*, 2014; Ayed *et al.*, 2013, 2010] methods.

In the rest of this chapter, we first show our scope of the problems in Section 3.1.1. Then, after briefly reviewing SSP [Narasimhan and Bilmes, 2005] in Section 3.2, we propose our method in Section 3.3 as a generalization to SSP. Relationships between our method and other existing methods [Gorelick *et al.*, 2014; Tang *et al.*, 2014; Ayed *et al.*, 2013, 2010] are further discussed in Section 3.4. Finally, we experimentally evaluate the proposed method in Section 3.5, and summarize this study in Section 3.6.

3.1.1 Scope of the Problems

Our objective is to seek the binary variable S such that it minimizes $E(S)$ of Eq. (3.1). In this study, we focus on three types of energy functions. Before defining those functions, we define the following function as the basis of all types.

Pairwise Submodular Functions

Let $s_i \in \{0, 1\}$ be a binary variable defined for pixels $i \in \Omega$ in the image domain Ω, and we define $S = \{i | s_i = 1\}$ as a *segment* in the domain Ω. If $R(S)$ in Eq. (3.1) is a linear product of a function $h_i = h(i) : \Omega \rightarrow \mathbb{R}$ and S

$$R(S) = \langle h, S \rangle = \sum_{i \in \Omega} h_i s_i \tag{3.2}$$

and $Q(S)$ is the sum of pairwise functions

$$Q(S) = \sum_{(i,j) \in \Omega} m_{ij} s_i s_j, \tag{3.3}$$

and if all the quadratic terms are non-positive ($m_{ij} \leq 0$), then $E(S)$ is submodular and can be globally minimized via graph cuts [Boykov and Kolmogorov, 2004; Kolmogorov and Zabin, 2004] in polynomial time.

Type-1: Higer-Order Supermodular Energies

We consider the energies $E(S)$ with pairwise submodular $Q(S)$ and the following form of $R(S)$:

$$R_{\text{type1}}(S) = \sum_z R_z(S) = \sum_z f_z(\langle g_z, S \rangle) \tag{3.4}$$

where $f_z(x)$ is convex and $g_z(i) : \Omega \rightarrow \mathbb{R}^+$ is a positive function. Hence, $R_z(S)$ is supermodular, *i.e.*, if it satisfies the following inequality for any $X, Y \subseteq \Omega$:

$$R_z(X) + R_z(Y) \leq R_z(X \cap Y) + R_z(X \cup Y). \tag{3.5}$$

Note that $R'_z(S) = -R_z(S)$ is submodular, if $R_z(S)$ is supermodular. While any submodular functions can be minimized in polynomial time [Schrijver, 2000], the minimization of supermodular functions is NP-hard.

Higher-order supermodular functions have been used, for example, as a L_p-distance histogram constraint $R(S) = \sum_{z \in Z} \left| h_z - n_z^S \right|^p$ for co-segmentation [Rother *et al.*, 2006; Vicente *et al.*, 2010; Mukherjee *et al.*, 2009] and tracking [Jiang, 2012], where $z \in Z$ is a bin of color or feature histograms, h_z is the given target histogram, and n_z^S is the number of pixels in S that fall into the bin z. A volumetric constraint $R(S) = |V_0 - |S||^p$ has been also used for medical image analysis [Gorelick *et al.*, 2012, 2013].

Type-2: Fractional Higer-Order Energies

We further deal with non-supermodular functions:

$$R_{\text{type2}}(S) = \sum_z f_z\left(\langle g_z, S\rangle / \langle w_z, S\rangle\right) \tag{3.6}$$

such that $R_z(S) = f_z\left(\langle g_z, S\rangle / \langle w_z, S_0\rangle\right)$ becomes a Type-1 term for fixed S_0. Examples of such functions include the KL-divergence $R(S) = -\sum_z p_z \log\left(n_z^S/|S| + \epsilon\right) + const.$ and the Bhattacharyya coefficient $R(S) = -\sum_z \sqrt{p_z n_z^S/|S|}$, both are used for image segmentation [Ayed *et al.*, 2010, 2013; Tang *et al.*, 2014]. Here, $\sum_z p_z = 1$ is the target distribution.

Type-3: Pairwise Non-submodular Energies

We also consider pairwise non-submodular energies, *i.e.*, $Q(S)$ of Eq. (3.3) containing non-submodular or supermodular terms (*i.e.* $m_{ij} > 0$). QPBO [Kolmogorov and Rother, 2007] is often used for such functions, but it leaves many variables *unlabeled* when the amount of non-submodular terms is significant [Gorelick *et al.*, 2014].

3.2 Submodular-Supermodular Procedure

Before presenting our method, we review SSP [Narasimhan and Bilmes, 2005], an optimization method for general supermodular functions, and later propose our method as its generalization.

SSP is classified as a bound optimization approach, where a tight upper bound function $\hat{E}(S|S^t)$ given an auxiliary variable S^t is derived for $E(S)$, *i.e.*,

$$E(S) \leq \hat{E}(S|S^t) \quad \text{and} \quad E(S^t) = \hat{E}(S^t|S^t). \tag{3.7}$$

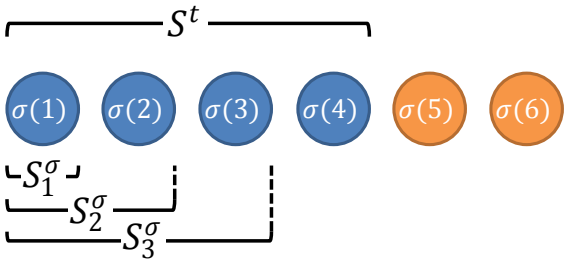
Then, the bound is iteratively minimized as

$$S^{t+1} = \arg\min \hat{E}(S|S^t), \quad (t = 0, 1, 2, ...). \tag{3.8}$$

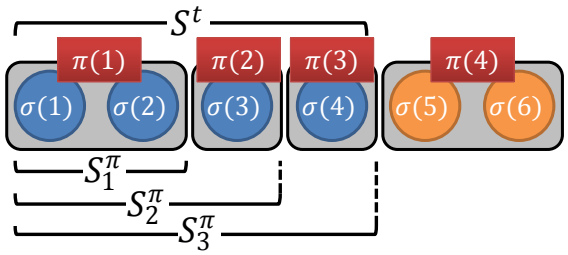
Here, it is guaranteed that the energy does not go up, *i.e.* $E(S^t) \geq E(S^{t+1})$ holds for any t, because $E(S^t) = \hat{E}(S^t|S^t) \geq \hat{E}(S^{t+1}|S^t) \geq E(S^{t+1})$.

3.2.1 Permutation Bounds

SSP derives tight bounds for supermodular functions based on the *superdifferential* [Fujishige, 2005; Iyer *et al.*, 2013], which is a similar concept to the subderivative of continuous functions. Given a supermodular functions $R(S)$ and $S^t \subseteq \Omega$, a modular (or linear)



(a) Chain in SSP [Narasimhan and Bilmes, 2005]



(b) Our generalized chain

Figure 3.3 Illustration of the chain and permutation.

function $H(S) := \langle h, S\rangle + R(\emptyset)$ that satisfies

$$H(S) - H(S^t) \geq R(S) - R(S^t) \tag{3.9}$$

for any $S \subseteq \Omega$, is called a *supergradient* of R at S^t. We denote $\partial R(S^t)$ the set of all the supergradients of R at S^t, which is called the superdifferential. Notice that if $H(S^t) = R(S^t)$ holds, then $H(S)$ gives a tight upper bound to $R(S)$. Such extreme points of $\partial R(S^t)$ may be obtained using the following theorem.

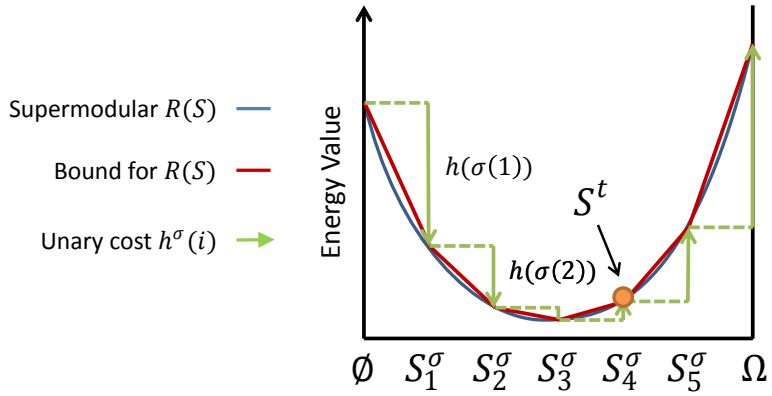
Theorem 3.1. *(Theorem 6.11 in [Fujishige, 2005]): For any $S^t \subseteq \Omega$, a modular function $H(S) = \langle h, S\rangle + R(\emptyset)$ is an extreme point of $\partial R(S^t)$, if and only if there exists a maximal chain*

$$C : \emptyset = S_0 \subset S_1 \subset \cdots \subset S_n = \Omega, \tag{3.10}$$

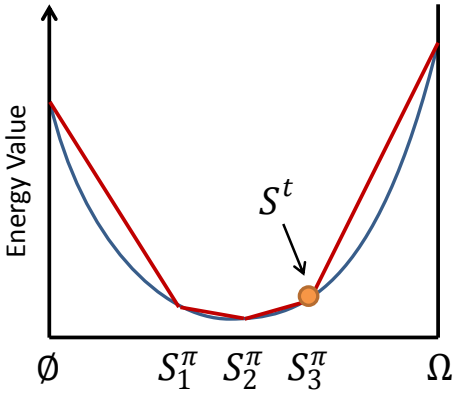
with $S_j = S^t$ for some j, such that

$$H(S_j) - H(S_{j-1}) = R(S_j) - R(S_{j-1}),\ (j = 1, ..., |\Omega|). \tag{3.11}$$

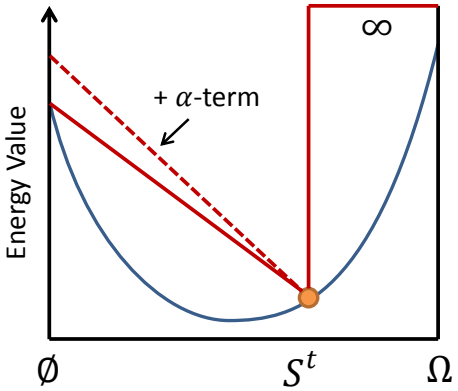
Based on this theorem, SSP [Narasimhan and Bilmes, 2005] derives a supergradient $H^\sigma(S|S^t)$ of $R(S)$ at S^t by the following greedy algorithm. Let $\sigma(j) : \{1, 2, ..., |\Omega|\} \to \Omega$ be a permutation of Ω that assigns the elements in S^t to the first $|S^t|$ positions, *i.e.*, $\sigma(j) \in S^t$ if and only if $j \leq |S^t|$. A maximal chain C^σ is then defined as $S_0^\sigma = \emptyset$ and



(a) Bound in SSP [Narasimhan and Bilmes, 2005]



(b) Our generalized bound



(c) Bound in AC [Ayed *et al.*, 2013]

Figure 3.4 Illustration of upper bounds for supermodular functions. The supermodular function $R(S)$ and its bounds are visualized by blue and red lines, respectively. The green arrows in (a) show the unary costs $h(\sigma(j))$ of a supergradient $H^{\sigma}(S|S^t)$.

$S^\sigma_j = \{\sigma(1), \sigma(2), ..., \sigma(j)\}$, so $S^\sigma_{|S^t|} = S^t$. See Figure 3.3 (a) for an illustration. Using this chain C^σ, a supergradient $H^\sigma(S|S^t)$ is obtained as

$$H^\sigma(S|S^t) = \langle h^\sigma, S\rangle + R(\emptyset), \tag{3.12}$$

where each unary cost $h^\sigma(i)$ is given by

$$h^\sigma(\sigma(j)) = R(S^\sigma_j) - R(S^\sigma_{j-1}),\ \ (j = 1, 2, .., |\Omega|). \tag{3.13}$$

Figure 3.4 (a) illustrates how $h^\sigma(\sigma(j))$ are computed. Since $R(S)$ is supermodular, variables in earlier positions of σ are assigned lower unary costs, *i.e.*, more prone to be labeled as 1 via cost minimization. Therefore, if we knew a priori how likely each variable is 1, the ideal permutation would arrange variables in order of decreasing likelihood so as to maximize the likelihood via cost minimization. Here, the bounds $H^\sigma(S)$ approximate $R(S)$ tightly at along the chain of solutions $\{S^\sigma_j\}$. However, because $H^\sigma(S)$ can largely deviate from $R(S)$ at other than $\{S^\sigma_j\}$, SSP is problematic when likelihood or permutation is given inaccurate.

3.3 Proposed Method

In the following sections, we first present our key idea by extending SSP [Narasimhan and Bilmes, 2005]. We then show how to apply it to and optimize the three types of functions in Sec. 3.3.2, and describe implementation details in Sec. 3.3.3.

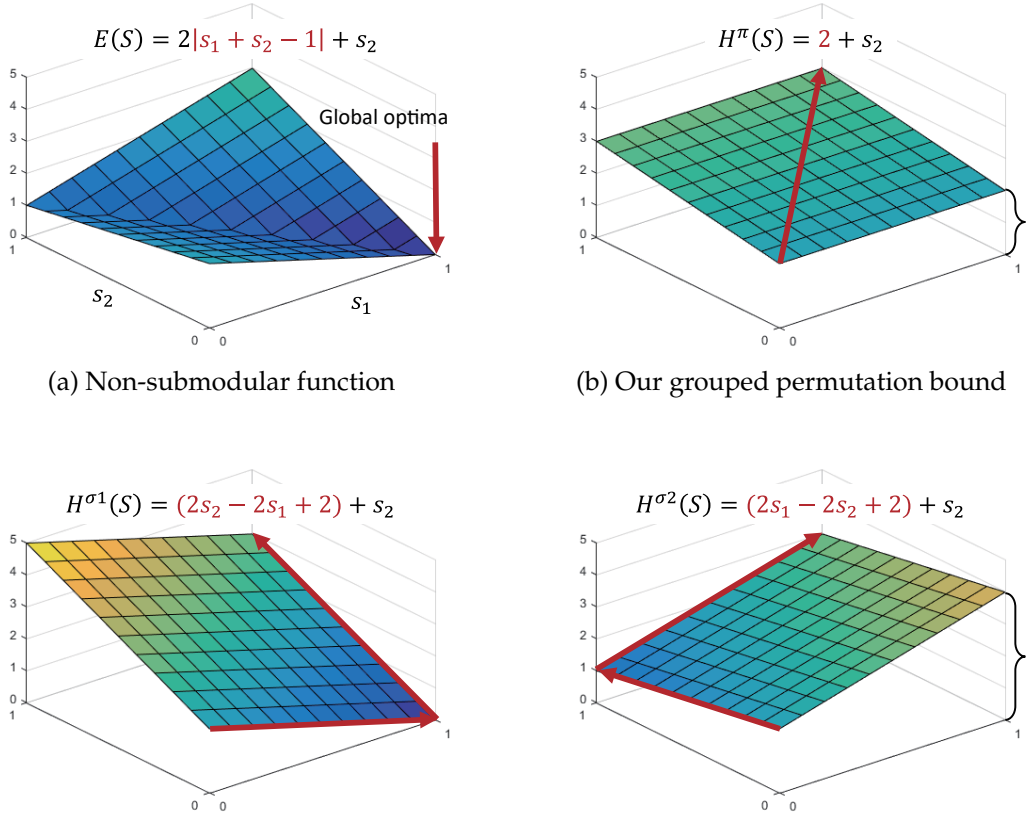
3.3.1 Grouped Permutation Bounds

In this section, we derive general bounds for Type-1 terms $R(S) = f(\langle g, S\rangle)$ by extending the SSP's permutation scheme. First, we introduce a *grouped permutation* π, which is made by grouping SSP's ordered-elements $\sigma = \{\sigma(1), \sigma(2), \cdots, \sigma(|\Omega|)\}$ into M ($M \leq |\Omega|$) groups: $\pi(1), \pi(2), \cdots, \pi(M) \subseteq \Omega$. Each group $\pi(j)$ contains some consecutive elements of σ: $\pi(j) = \{\sigma(j'), \sigma(j'+1), \cdots, \sigma(j'+m)\}$, and groups are mutually disjoint: $\pi(j) \cap \pi(j') = \emptyset$ if $j \neq j'$. Using this grouped permutation π we define a chain C^π: $S^\pi_0 = \emptyset$ and $S^\pi_j = \pi(1) \cup \pi(2) \cup ... \cup \pi(j)$ as illustrated in Figure 3.3 (b). Here, we make sure that any group does not across $\sigma(|S^t|)$ and $\sigma(|S^t|+1)$, so there exists $S^\pi_j = S^t$ for some j. Then, our bound for $R(S)$ is defined similarly to that of SSP in Eq. (3.12) as

$$H^\pi(S|S^t) = \langle h^\pi, S\rangle + R(\emptyset), \tag{3.14}$$

where unary costs $h^\pi(i) : \Omega \to \mathbb{R}$ are defined for $i \in \pi(j)$ and $j = 1, 2, ..., M$ as

$$h^\pi(i) = g(i)\left[R(S^\pi_j) - R(S^\pi_{j-1})\right] / \langle g, \pi(j)\rangle\,. \tag{3.15}$$



(a) Non-submodular function

(b) Our grouped permutation bound

(c) SSP's bound (best permutation)

(d) SSP's bound (worst permutation)

Figure 3.5 Illustrations of our grouped permutation bound and SSP's bound. (a) A supermodular function with two variables, whose optimal solution is $(s_1, s_2) = (1, 0)$ or $S = \{1\}$. (b) Our grouped permutation bound with the full grouping $\pi = \{1, 2\}$. (c) SSP's bound with the best permutation $\sigma 1$ $(1 \rightarrow 2)$. (d) SSP's bound with the worst permutation $\sigma 2$ $(2 \rightarrow 1)$. Notice that the SSP's bounds are tight to the function $E(S)$ along the chain of solutions determined by the permutations (red arrows). If the optimal solution exists on the chain such as shown in (c), the SSP's bound is tight at the optimal solution. However, when the permutation is inaccurate as shown in (d), our grouped permutation produces a tighter bound than that of SSP.

The essence of this formulation becomes clearer if we assume $g = 1$ so that $H^{\pi}(S|S^t)$ becomes piecewise-mean-approximations of SSP's bound $H^{\sigma}(S|S^t)$, which is visualized in Figure 3.4 (b) using the example permutation π shown in Figure 3.3 (b). Note that our bound $H^{\pi}(S|S^t)$ becomes equivalent to the supergradients $H^{\sigma}(S|S^t)$ of SSP, if $\pi(j) = \{\sigma(j)\}$.

Proposition 3.1. *The function $H^{\pi}(S|S^t)$ satisfies the conditions of Eq. (3.7) and is thus a tight upper bound for $R(S)$.*

Proof. See Appendix A.1. □

One may wonder why our grouped permutation bound is better than SSP's bound that looks tighter than ours. However, we should pay attention to the x-axes of Figure 3.4,

which are defined along a 1D chained path $S \in \{S_j^\sigma\}$ ($j = 0, 1, \cdots, |\Omega|$). The path can be seen as a single slice of the high dimensional variable space of S. Indeed, SSP's bound can be tighter than ours but only along a single path in $2^{|\Omega|}$ of all possible states $^\forall S \subseteq \Omega$. If the optimal solution exists on the path (*i.e.*, the permutation σ is very accurate), SSP's bound is the best choice. However, when the permutation σ is unreliable, the optimal solution is unlikely on the path. In this case, our bound will be a better approximation.

To visually illustrate this property, let us consider a supersubmodular function defined with two binary variables $s_1, s_2 \in \{0, 1\}$ as $E(S) = 2|s_1 + s_2 - 1| + s_2$. This function is visualized in Figure 3.5 (a). By approximating the supermodular term $|s_1 + s_2 - 1|$ using our bound, we obtain a linear approximation as shown in Figure 3.5 (b). Similarly, SSP's bounds with the best and worst permutations are shown in Figures 3.5 (c) and (d). When the permutation of SSP is inaccurate, our grouped permutation produces better an approximation bound.

This analysis gives us a strategy of how to make the grouped permutation. That is, we make fine/coarse approximation bounds when elementwise permutations σ are accurate/inaccurate. Specifically, when the permutation of $\sigma(j)$ and $\sigma(j+1)$ is unreliable, we do not take the risk of making a bad bound by explicitly permuting the two. Instead, we put them into the same group in order to treat them equally and leave a decision (*i.e.*, which is more likely to be labeled as 1) to other interactions, *e.g.*, pairwise smoothness terms. As we will show in Sec. 3.3.3, our method makes coarse-to-fine approximation bounds over iterations. This helps to avoid weak local minimums at early iterations and reach a strong local minimum at convergence.

3.3.2 Optimization Procedure

We optimize $E(S)$ by iteratively minimizing its approximation function $\hat{E}(S|S^{t-1})$ derived by our grouped permutation bounds. Here, the minimization of $\hat{E}(S)$ is achieved using a max-flow/min-cut algorithm [Boykov and Kolmogorov, 2004].

Type-1: We derive a bound $H_z^\pi(S|S^t)$ for each $R_z(S)$ of $R_{\text{type1}}(S)$, and set $\hat{E}(S|S^t) = \sum_z H_z^\pi(S|S^t) + Q(S)$. Here, it is guaranteed that minimization of $\hat{E}(S)$ does not increase the energy $E(S)$. Therefore, its optimization procedure is a simple iteration algorithm shown in Algorithm 3.1 (without lines 5 and 6).

Type-2: Similarly to [Ayed *et al.*, 2013; Tang *et al.*, 2014], we approximate $R_{\text{type2}}(S)$ by partially fixing S at S^t as

$$\tilde{R}_{\text{type2}}(S|S^t) = \sum_z f_z\left(\langle g_z, S\rangle / \langle w_z, S^t\rangle\right). \tag{3.16}$$

Here, $\tilde{R}_{\text{type2}}(S|S^t)$ is a Type-1 function. Therefore, we can approximate $\tilde{R}_{\text{type2}}(S|S^t)$ using our bounds $H^\pi(S|S^t)$. As long as $S \subseteq S^t$ is forced, this linear function H^π

Table 3.1 The proposed linear conversions of non-submodular terms $m_{ij}s_is_j$ with comparisons to the conversions of SSP [Narasimhan and Bilmes, 2005] and LSA [Gorelick *et al.*, 2014]. Below, we omit m_{ij} which is multiplied to all terms in the table. Notice that our linear conversion can bee seen as generalization of both SSP and LSA-AUX.

(s_i^t, s_j^t)	Ours	SSP	LSA-AUX	LSA-TR
$(0,0)$	$\frac{1}{2}s_i + \frac{1}{2}s_j$ if $\pi(1)=\{i,j\}$ s_j if $\pi(1)=\{i\}$ s_i if $\pi(1)=\{j\}$	s_j if $\sigma(1)=i$ s_i if $\sigma(1)=j$	$\frac{1}{2}s_i + \frac{1}{2}s_j$	0
$(0,1)$	s_i	s_i	s_i	s_i
$(1,0)$	s_j	s_j	s_j	s_j
$(1,1)$	$\frac{1}{2}s_i + \frac{1}{2}s_j$ if $\pi(1)=\{i,j\}$ s_j if $\pi(1)=\{i\}$ s_i if $\pi(1)=\{j\}$	s_j if $\sigma(1)=i$ s_i if $\sigma(1)=j$	$\frac{1}{2}s_i + \frac{1}{2}s_j$	$s_i + s_j - 1$

Algorithm 3.1: OPTIMIZATION FOR TYPE-1,3 [TYPE-2]

1 Initialize S^0 ;
2 **for** $t = 0, 1, 2, \cdots$ **do**
3 Create a permutation π // for all Types ;
4 $S^{t+1} \leftarrow \operatorname{argmin} \hat{E}(S|S^t)$ // for Type-1,3 ;
5 $[\{S^\lambda\} \leftarrow \operatorname{argmin} \hat{E}(S|S^t) + \lambda(|S^t| - |S|) \text{ for } {}^\forall\lambda]$;
6 $[S^{t+1} \leftarrow \operatorname{argmin} E(S) \text{ for } S \in \{S^\lambda, S^t\}]$ // for Type-2 ;
7 **end**

makes a tight bound for R_{type2} [Ayed *et al.*, 2013]. However, we rather do not restrict S for allowing bi-directional optimization. In this case, the minimization of $\hat{E}(S|S^t) = H^\pi(S|S^t) + Q(S)$ may increase the actual energy $E(S)$. For this we use the pseudo bound optimization scheme of [Tang *et al.*, 2014], where we make a family of relaxed bounds $\hat{E}_\lambda(S) = \hat{E}(S|S^t) + \lambda(|S^t| - |S|)$, and we exhaustively search $S^\lambda = \operatorname{argmin} \hat{E}_\lambda(S)$ for all $\lambda \in (-\infty, \infty)$ by using parametric maxflow [Kolmogorov *et al.*, 2007]. Then we choose the best S^λ that minimizes $E(S)$. Therefore, the optimization procedure in Algorithm 3.1 takes lines 5 and 6 instead of 4 for Type-2.

Type-3: We make a bound $\hat{E}(S)$ by approximating each of non-submodular terms $R(S) = m_{ij}s_is_j$ ($m_{ij} > 0$) with a linear function $h_is_i + h_js_j + const$. Its form depends on both current values (s_i^t, s_j^t) and the permutation π of $\{i,j\}$, and given as $h(i') = \left[R(S^\pi_{j'}) - R(S^\pi_{j'-1})\right] / |\pi(j')|$ similarly to Eq. (3.15). We summarize the conversion in Table 3.1.

3.3.3 Implementation Details

We make permutations σ and π according to the signed distance from the boundary of S^t. In [Rother *et al.*, 2006], the Euclidean distance was used, but we use geodesic

Figure 3.6 Illustrations of geodesic distance [Criminisi *et al.*, 2008].

distance [Criminisi *et al.*, 2008] for more effectively creating the permutations. Figure 3.6 shows examples of both distances. We denote the geodesic distance for segments S by $D(i|S, I) : \Omega \rightarrow \mathbb{R}$, and $D(i|S) \leq 0$ for $i \in S$. See Eq. (7) of [Criminisi *et al.*, 2008] and also Appendix A.2 for its definition. As discussed in [Criminisi *et al.*, 2008], $D(i|S, I)$ is efficiently computed in $O(|\Omega|)$ using an approximate algorithm [Toivanen, 1996].

Using the geodesic distance, we construct the bound $H^{\pi}(S|S^t)$ in each iteration as follows. Firstly, we compute $D(i|S^t)$ for the current segments S^t. Secondly, we make a permutation σ such that $D(\sigma(j)) \leq D(\sigma(j+1))$. Finally, we make a grouped permutation π from σ. We process $\sigma(j)$ from $\sigma(2)$ to $\sigma(|\Omega|)$, and put $\sigma(j)$ into the same group with $\sigma(j-1)$ if $D(\sigma(j)) - D(\sigma(j-1)) \leq \tau$, while making sure the group does not across $\sigma(|S^t|)$ and $\sigma(|S^t|+1)$. Basically, the size of the threshold τ reflects how much the permutation σ by $D(i)$ is reliable. We empirically use a grouping threshold given by $\tau = (\mu + as)/(t+1)^{\kappa}$ $(t = 0, 1, 2, ...)$, where μ and s are the mean and standard deviation of distance differences $|D(\sigma(j)) - D(\sigma(j-1))|$. We use this monotonically decreasing thresholds, because as iterations proceed the segments S^t are expected to be more accurate and so permutations σ by $D(i|S^t)$ becomes accordingly more reasonable. Also, when $S^0 = \Omega$ or $S^0 = \emptyset$, so $D(i|S^t)$ cannot be defined, we set $\pi(1) = \Omega$. This makes the full linear approximations of $R(S)$ drawn from $R(\emptyset)$ to $R(\Omega)$, which give reasonable initial approximations.

3.4 Relationship with Prior Art

In this section, we discuss relationships between our method and other previous methods Auxiliary Cuts (AC) [Ayed *et al.*, 2013], Parametric Pseudo Boolean Cuts (pPBC) [Tang *et al.*, 2014] and Local Submodular Approximations (LSA) [Gorelick *et al.*, 2014].

3.4.1 Auxiliary Cuts and pPBC

The authors of [Ayed *et al.*, 2013] derived a bound for $R_{\text{type1}}(S)$ by using the Jensen's inequality and assuming $S \subseteq S^t$. Its essential form is given as

$$A_\alpha(S|S^t) = B_{\text{shr}}(S|S^t) + \alpha R(S^t)\left(1 - \frac{\langle 1, S\rangle}{\langle 1, S^t\rangle}\right), \tag{3.17}$$

where $\alpha \geq 0$ is a parameter and $B_{\text{shr}}(S|S^t)$ is expressed as

$$B_{\text{shr}}(S|S^t) = R(\emptyset) + \frac{R(S^t) - R(\emptyset)}{\langle g, S^t\rangle}\langle g, S\rangle. \tag{3.18}$$

From Eq. (3.18), $B_{\text{shr}}(S|S^t)$ can be seen as the linear approximations of $R(S)$ drawn from $R(\emptyset)$ to $R(S^t)$. See Figure 3.4 (c), where the solid red line visualizes this linear approximation bound, the infinite bound reflects the restriction $S \subseteq S^t$, and the dotted line depicts the effect of the α-term in Eq. (3.17). The use of this bound results in successively *shrinking* segments: $S^0 \supseteq S^1 \supseteq S^2$. pPBC [Tang *et al.*, 2014] extends AC by exhaustively searching the best $\alpha \in (-\infty, \infty)$ in each iteration using parametric maxflow [Kolmogorov *et al.*, 2007].

To point out a relationship to our method, we use another bound $B_{\text{exp}}(S|S^t)$ for $R(S)$, which is similar to $B_{\text{shr}}(S|S^t)$ and can be derived by [Ayed *et al.*, 2013]'s derivations as

$$B_{\text{exp}}(S|S^t) = R(S^t) + \frac{R(\Omega) - R(S^t)}{\langle g, \Omega \setminus S^t\rangle}\langle g, S \setminus S^t\rangle. \tag{3.19}$$

Here, S^t is restricted to $S^t \subseteq S$, so the iterative minimization of $B_{\text{shr}}(S|S^t)$ successively *expands* the segments S. This bound can be seen as the linear approximation of $R(S)$ drawn from $R(S^t)$ to $R(\Omega)$.

We now show that these bounds $B_{\text{shr}}(S|S^t)$ and $B_{\text{exp}}(S|S^t)$ can be derived using our grouped permutation bounds. When a grouped permutation is given as $\pi(1) = S^t$ and $\pi(2) = \bar{S}^t$ with $\bar{S}^t := \Omega \setminus S^t$, our bound $H^\pi(S|S^t)$ becomes the following form:

$$H_{\text{full}}(S|S^t) = B_{\text{shr}}(S \cap S^t|S^t) + B_{\text{exp}}(S \cap \bar{S}^t|S^t). \tag{3.20}$$

Notice that it no longer requires the restrictions for S (*i.e.*, $S \subseteq S^t$ or $S^t \subseteq S$) used in [Ayed *et al.*, 2013; Tang *et al.*, 2014], which are turned out to be unnecessary by our derivation. Also notice that H_{full} does not depend on the permutation π. Although H_{full} does not accurately approximate its original function $R(S)$, it is still useful when permutations are inaccurate (*e.g.* at first iterations). Our method is designed to behave between AC [Ayed *et al.*, 2013] and SSP [Narasimhan and Bilmes, 2005] and to produce coarse-to-fine approximation bounds as iterations proceed, by using the monotonically

decreasing grouping-threshold τ defined in Section 3.3.3.

3.4.2 Local Submodular Approximations

Very recently, an optimization method for pairwise non-submodular energies called LSA [Gorelick *et al.*, 2014] has been proposed and shown to outperform other state-of-the-art methods such as TRW-S [Kolmogorov, 2006] and QPBO [Kolmogorov and Rother, 2007]. This method approximates non-submodular pairwise terms by linear terms. In [Gorelick *et al.*, 2014], two types of approximation conversions are proposed. LSA-TR applies a Taylor-based approximation and uses the gradient-descent framework of FTR [Gorelick *et al.*, 2013]. LSA-AUX uses a bound-based approximation and uses the bound optimization framework. The linear conversions of LSA-TR and LSA-AUX are summarized in Table 3.1.

As you can see from the table, our linear conversion includes the conversion of LSA-AUX. In fact, our conversion with a full-grouping π produces the same bounds with LSA-AUX. Furthermore, as mentioned in [Gorelick *et al.*, 2014], there are other types of bounds that can be made by the permutation scheme of SSP [Narasimhan and Bilmes, 2005]. Our method generalizes both conversions of SSP and LSA-AUX, and adaptively chooses either conversion for each term.

3.5 Experiments

For evaluation, we use four variations of our method: **SDC-GEO** is the proposed method described in Section 3.3. **SDC-DIST** uses the standard Euclidean distance for making permutations σ, but the other settings are the same with SDC-GEO. **SSP-DIST** is SSP [Narasimhan and Bilmes, 2005] that follows the implementations by Rother *et al.* [2006]. It is basically the same with SDC-DIST but uses no mean approximations for bound constructions. We also use **SSP-GEO**, which is the same with SSP-DIST but uses the geodesic distance. Note that when $S^0 = \Omega$ at the first iterations, we make permutations σ randomly for SSP-DIST and SSP-GEO, based on 10×10-pixels of patches as described in [Rother *et al.*, 2006].

We also compare with three state-of-the-art methods for higher-order energies: **AC** [Ayed *et al.*, 2013] uses the bound A_α of Eq. (3.17). **pPBC** [Tang *et al.*, 2014] uses the same bound but exhaustively chooses the best α using parametric maxflow [Kolmogorov *et al.*, 2007]. **FTR** [Gorelick *et al.*, 2013] is the state-of-the-art of gradient-descent methods. For pairwise energies, we compare with **LSA-AUX,-TR** [Gorelick *et al.*, 2014] and **pPBC-T,-B,-L** [Tang *et al.*, 2014] as state-of-the-art.

All the methods are implemented in C++, and run on a system with a 3.5GHz Core i7 CPU and 16GB RAM.

3.5.1 Segmentation via Distribution Matching

Similarly to [Tang *et al.*, 2014; Ayed *et al.*, 2013; Rother *et al.*, 2006], we evaluate the performances of the methods using the GrabCut dataset [Rother *et al.*, 2004]. For pure evaluations of optimization performances, we learn the target histograms from the ground truth[1]. We use a standard 16-neighbor pairwise smoothness term similar to [Rother *et al.*, 2004]: $Q(S) = \lambda \sum_{ij} \max(w_{ij}, \epsilon)|s_i - s_j|/|p_i - p_j|$ where p_i is pixel coordinates and $w_{ij} = \exp(-\beta|I_i - I_j|^2)$. Here, β is automatically estimated as $\beta = 2E[|I_i - I_j|^2]$ the expectation over all neighbor pairs. We use RGB-histograms.

Type-1: L_2 and L_1 Distance of Histograms

For Type-1 terms, we use the L_2 and L_1 distances between histograms. For $\{\lambda, \epsilon\}$, we use $\{1.0, 0.5\}$ and $\{0.5, 0.5\}$ for the L_2 and L_1-distances, respectively. S^t is trivially initialized as $S^0 \leftarrow \Omega$. For both SDC-GEO and SDC-DIST, we use a grouping threshold $\tau = (\mu + 8s)/(t+1)^{2.5}$.

Tables 3.2 and 3.3 summarize the performance comparisons, showing average misclassified pixel rates, energy values of $E(S)$ and $R(S)$, running times, and individual-image comparisons with SDC-GEO. Among the seven methods, our proposed method SDC-GEO outperforms the others for all error and energy scores. In some cases, SDC-GEO completely outperforms pPBC, AC, FTR, and SSP-DIST for all individual images. Note that the error rates of SSP-DIST are better than the rates originally reported in [Rother *et al.*, 2006] because the definition of pixel feature histograms is different[2]. Comparing the results of SDC-DIST and SSP-DIST with L_2 and 64^3 bins, SDC-DIST finds more accurate segmentations in spite of the higher energies. This is because in SSP-DIST the appearance consistencies are forced regardless of how permutations σ and corresponding bounds are inaccurate, resulting in highly non-smooth, visibly bad local minimas. In Figure 3.7, we show example results of L_2 and L_1 with 64 bins. Figures 3.9 and 3.10 show the plots of the accuracy transitions using the L_2 and L_1-distances w.r.t. the number of bins. As shown, SDC-GEO is robust to the difference of the number of bins. In order to show robustness to initialization, we compare SDC-GEO, pPBC, and FTR using two types of initialization shown in Figure 3.1. Unlike pPBC (and AC) that can only reduce the target regions, our method is robust to initialization and finds very accurate solutions even for such difficult camouflage images.

[1] If the target histograms are inaccurate, the minimum solutions of $E(S)$ are deviated from the ground truth [Taniai *et al.*, 2012], and the error rate criteria thus does not reflect the actual performances of the optimization methods.

[2] [Rother *et al.*, 2006] uses a normalized 2D color vector and texton as a pixel feature, since the method is intended for co-segmentation and image retrieval.

Type-2: Bhattacharyya Distance and KL Divergence

We use Bhattacharyya distance and KL divergence as Type-2. As described in Section 3.3.2, we use the pseudo bound optimization scheme of [Tang *et al.*, 2014] using parametric maxflow [Kolmogorov *et al.*, 2007] for our method SDC-GEO. We show the performance comparisons with pPBC, AC, and FTR in Table 3.4, where our method outperforms the others in error and energy values. It is worth noting that although SSP [Narasimhan and Bilmes, 2005] is originally oriented for supermodular functions, our extended method successfully optimizes non-supermodular functions (Type-2).

3.5.2 Type-3: Image Deconvolution

We make a blurred image $\tilde{I}$ by a mean filter and additive Gaussian noises: $\tilde{I}_i = \frac{1}{9}\sum_{j\in W_i} I_j + \mathcal{N}(0,\sigma^2)$, where W_i is a 3×3 window centered at i. We recover the original image I by minimizing $E(S) = \sum_{i\in\Omega}(\tilde{I}_i - \frac{1}{9}\sum_{j\in W_i} s_j)^2$. In Figure 3.11, we show example results of our method, pPBC [Tang *et al.*, 2014], and LSA [Gorelick *et al.*, 2014] for two noise levels $\sigma = 0.15, 0.30$. In Figure 3.12, we show the plots of energies, squared errors ($\sum_i |s_i - I_i|^2$), and running times. LSA-TR and pPBC-T,-L reach the lower energies but inaccurate results. Our method and LSA-AUX perform best in terms of both accuracy and efficiency.

3.5.3 Type-3: Curvature Regularization

We apply our method to a curvature regularization model of [El-Zehiry and Grady, 2010] in image binarization. We show the input image and results by our method, pPBC [Tang *et al.*, 2014], and LSA [Gorelick *et al.*, 2014] in Figure 3.13. The plots in Figure 3.14 show energies at convergence w.r.t. regularization weights. Our method always reaches the lowest energies and is most stable among all methods.

3.6 Summary

In this study we have revisited SSP [Narasimhan and Bilmes, 2005], an early approach to higher-order energy optimization, and proposed our method as generalization of SSP. The key idea of our method is piecewise mean approximation bounds, which are designed to produce coarse-to-fine approximation bounds during iterations. We further show that our method has close connections to some state-of-the-art methods [Ayed *et al.*, 2013; Tang *et al.*, 2014; Gorelick *et al.*, 2014]. Although the proposed method shows promising improvements over state-of-the-art methods [Tang *et al.*, 2014; Gorelick *et al.*, 2013, 2014; Ayed *et al.*, 2013], we would like to further push the envelope by improving the definition of geodesic distance and the thresholding scheme for making grouped permutations.

Table 3.2 Evaluations on the GrabCut dataset [Rother *et al.*, 2004] using L_2-distance. We compare the proposed SDC-GEO with SDC-DIST, SSP-GEO, SSP-DIST, pPBC [Tang *et al.*, 2014], AC [Ayed *et al.*, 2013], FTR [Gorelick *et al.*, 2013]. We show average error rates, $E(S)$, $R(S)$, times over 50 images. The last column shows the number of images for which the proposed method (**SDC-GEO**) outperforms each method. We use 192^3 and 64^3 bins. SSP-DIST is SSP [Narasimhan and Bilmes, 2005] that follows the implementations by Rother *et al.* [2006].

Method	Error (%)		$E(S)$		$R(S)$		Time (sec)		SDC-GEO vs	
(L_2-distance)	192^3	64^3	192^3	64^3	192^3	64^3	192^3	64^3	192^3	64^3
Ground truth	0	0	3569	3569	0	0	-	-	-	-
SDC-GEO	**0.095**	**0.288**	**3511**	**3809**	179	**235**	6.2	11.7	-	-
SDC-DIST	0.134	1.123	4121	18922	577	9946	2.3	2.7	32	46
SSP-GEO	0.116	1.132	3594	6477	**166**	377	4.9	15.1	24	45
SSP-DIST	0.312	2.575	4415	13969	295	877	2.0	4.2	32	50
pPBC	1.062	2.677	19381	190332	13187	176315	36.7	24.6	49	50
AC	1.214	3.542	19888	195729	13646	177947	**1.0**	**1.0**	50	50
FTR	1.859	3.167	21003	153212	17669	146394	36.8	121.2	50	48

Table 3.3 Evaluations on the GrabCut dataset [Rother *et al.*, 2004] using L_1-distance. See Table 3.2 for the descriptions.

Method	Error (%)		$E(S)$		$R(S)$		Time (sec)		SDC-GEO vs	
(L_1-distance)	192^3	64^3	192^3	64^3	192^3	64^3	192^3	64^3	192^3	64^3
Ground truth	0	0	1785	1785	0	0	-	-	-	-
SDC-GEO	**0.033**	**0.205**	**1804**	**1882**	54	**120**	5.4	9.4	-	-
SDC-DIST	0.041	0.242	1813	1967	70	229	1.9	3.8	38	42
SSP-GEO	0.043	0.943	1813	2766	**31**	298	3.7	9.7	28	46
SSP-DIST	0.075	1.341	1898	3666	50	411	1.6	2.4	34	48
pPBC	0.154	0.583	2013	2632	309	997	17.4	18.9	48	48
AC	0.339	1.281	2502	4336	762	2476	**0.6**	**0.6**	50	50
FTR	0.147	0.366	1908	2105	277	495	46.1	100.7	46	41

Table 3.4 Evaluations on the GrabCut dataset [Rother *et al.*, 2004] using the Bhattacharyya distance and KL divergence. We use 64^3 bins and the bounding box initialization.

Method	Error (%)		$E(S)$		Time (sec)	
	Bhat.	KL-div.	Bhat.	KL-div.	Bhat.	KL-div.
SDC-GEO	**0.373**	**0.515**	**-14906**	**7025**	19.8	12.9
pPBC [Tang *et al.*, 2014]	0.498	0.818	-14870	7057	1.8	1.5
AC [Ayed *et al.*, 2013]	18.29	16.07	-11878	7490	**0.4**	**0.4**
FTR [Gorelick *et al.*, 2013]	0.435	1.076	-14894	7049	2.9	6.4

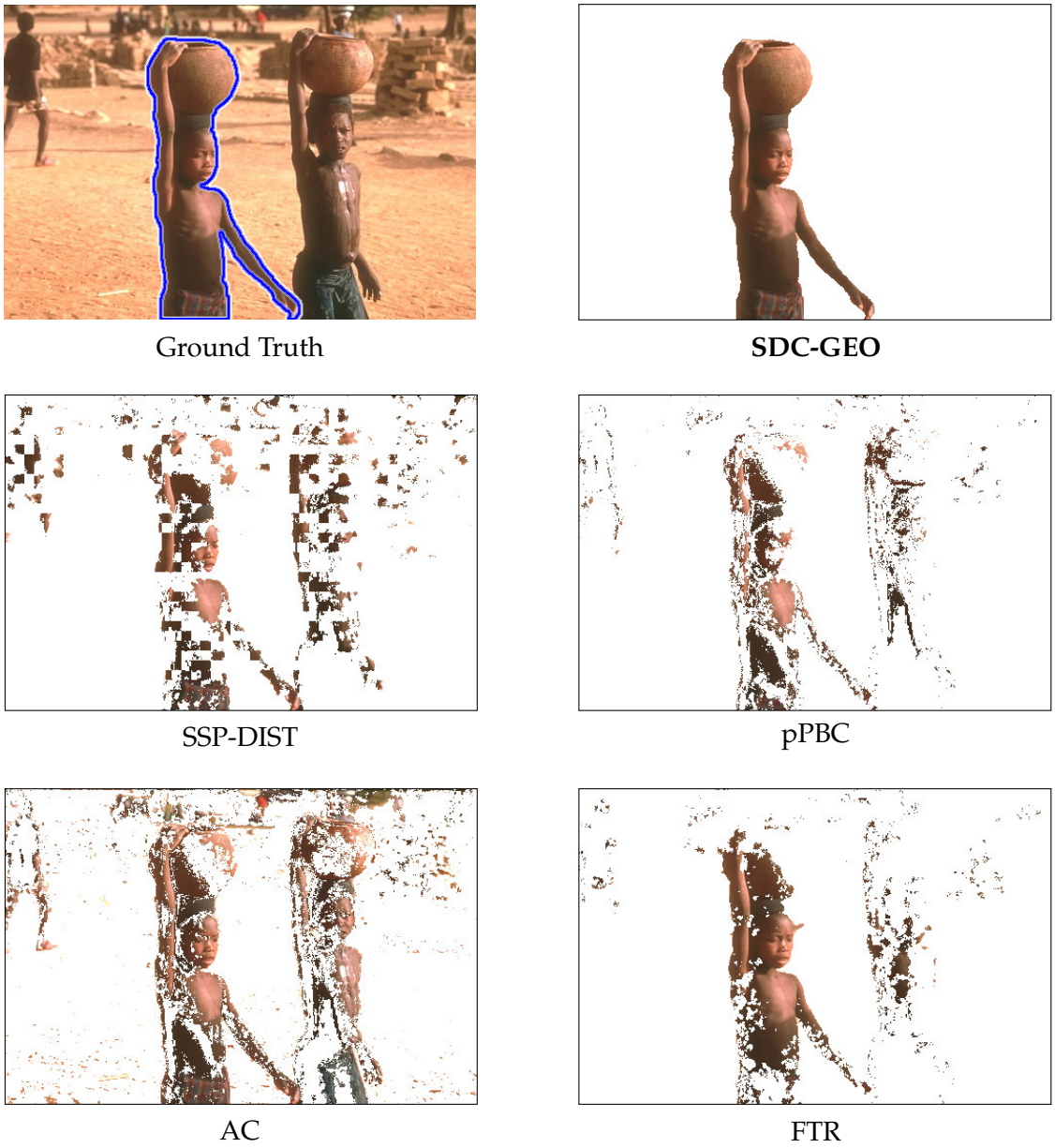


Figure 3.7 Examples of L_2-distance matching. We show the ground truth segmentation (blue boundary), results of the proposed SDC-GEO, SSP-DIST [Rother *et al.*, 2006], pPBC [Tang *et al.*, 2014], AC [Ayed *et al.*, 2013] and FTR [Gorelick *et al.*, 2013]. The segments are initialized as all foreground, and 64^3 bins of histograms are used.

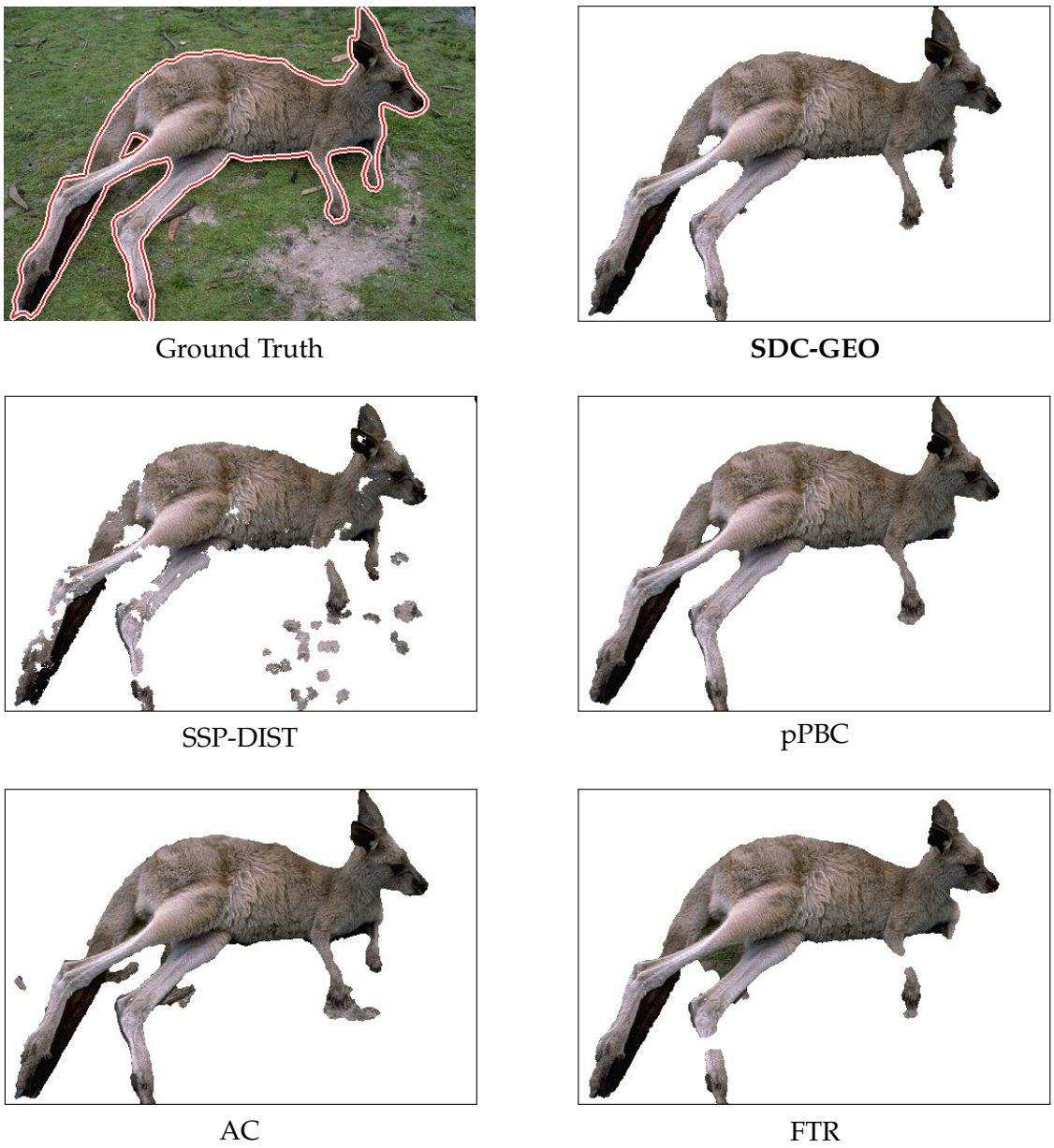


Figure 3.8 Examples of L_1-distance matching. We show the ground truth segmentation (blue boundary), results of the proposed SDC-GEO, SSP-DIST [Rother *et al.*, 2006], pPBC [Tang *et al.*, 2014], AC [Ayed *et al.*, 2013] and FTR [Gorelick *et al.*, 2013]. The segments are initialized as all foreground, and 64^3 bins of histograms are used.

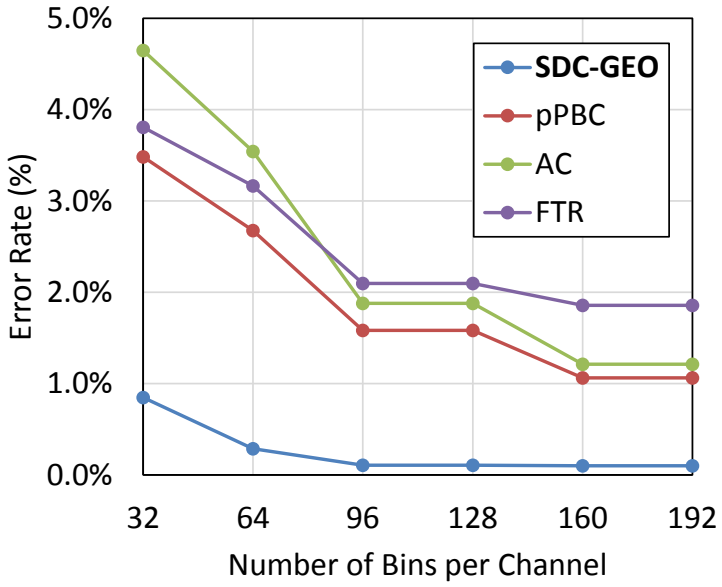


Figure 3.9 Error rate transitions w.r.t. the number of bins using L_2 distance. We compare the proposed SDC-GEO with pPBC [Tang *et al.*, 2014], AC [Ayed *et al.*, 2013] and FTR [Gorelick *et al.*, 2013]. Our method significantly outperforms state-of-the-art methods.

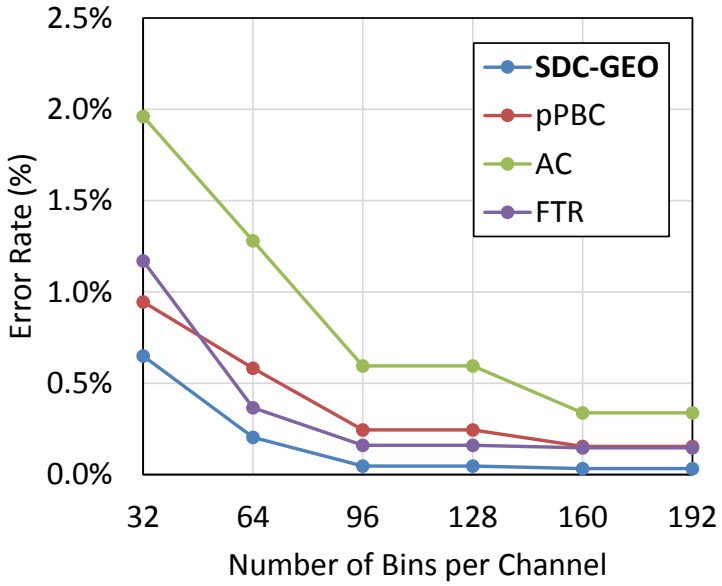


Figure 3.10 Error rate transitions w.r.t. the number of bins using L_1 distance. We compare the proposed SDC-GEO with pPBC [Tang *et al.*, 2014], AC [Ayed *et al.*, 2013] and FTR [Gorelick *et al.*, 2013]. Similarly to the L_2 distance case, Our method outperforms state-of-the-art methods.

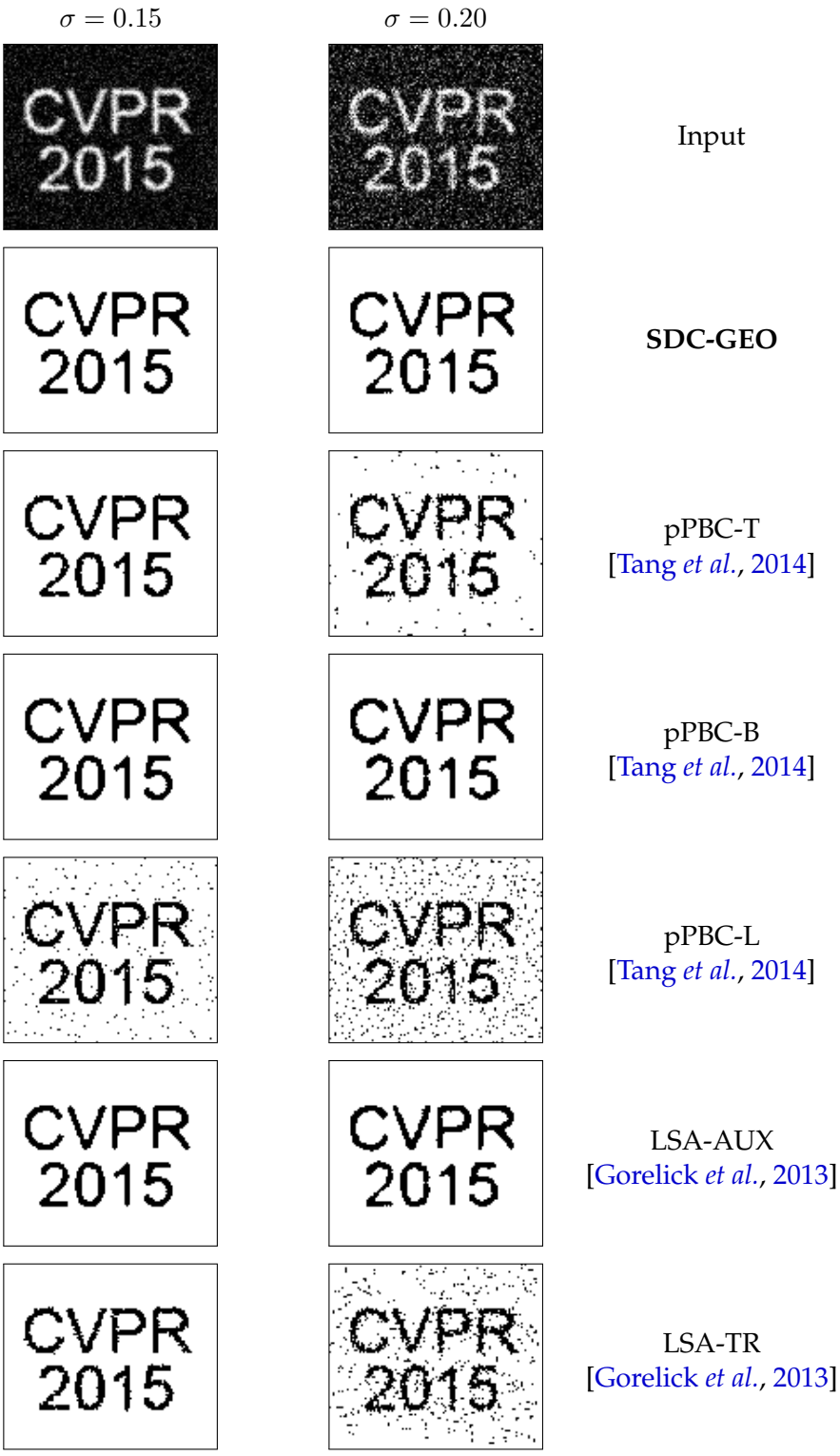


Figure 3.11 Image deconvolution results. We show results for two images with noise levels of $\sigma = 0.15$ (left) and $\sigma = 0.30$ (right). Our method is accurate and stable.

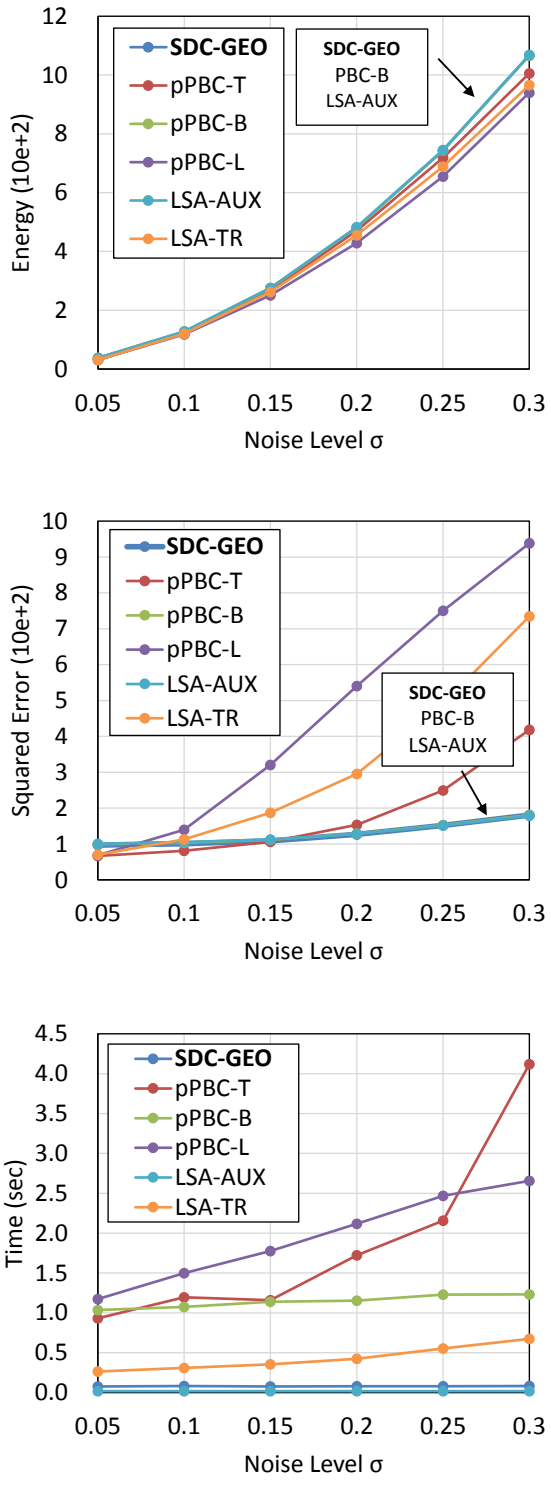


Figure 3.12 Performance comparisons of image deconvolution. Convergence energies, square errors, and running times w.r.t. noise levels are shown. The values are averaged over 30 random noise images at each point. We use $\tau = (\mu + s)/(t+1)^{2.5}$ for our method. Notice that our method obtains more accurate solutions in spite of their higher energies, because our course-to-fine scheme avoids bad local minimums.

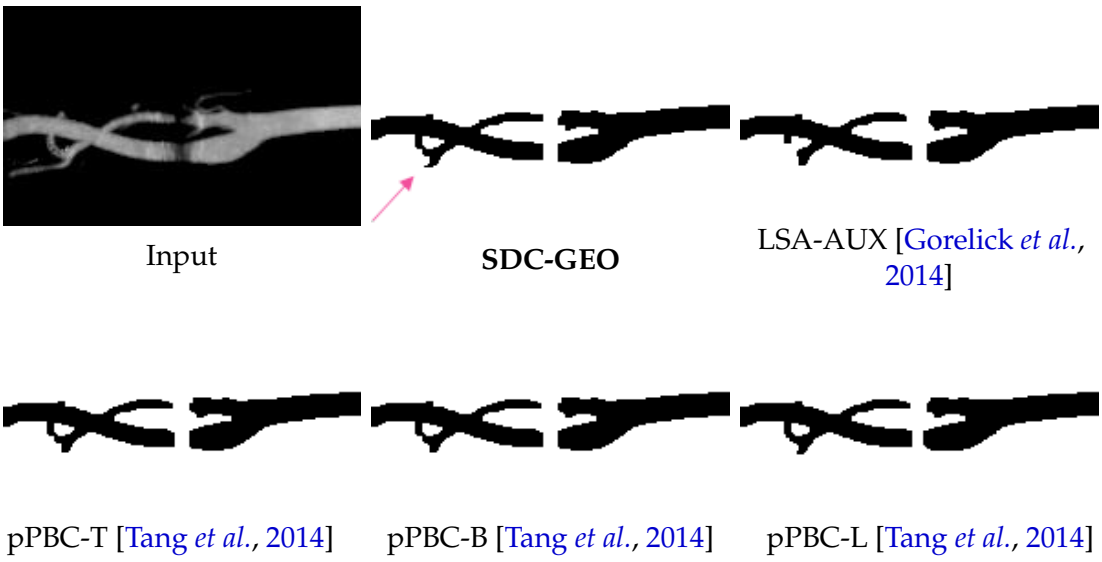


Figure 3.13 Results of curvature optimization at the weight of 25.

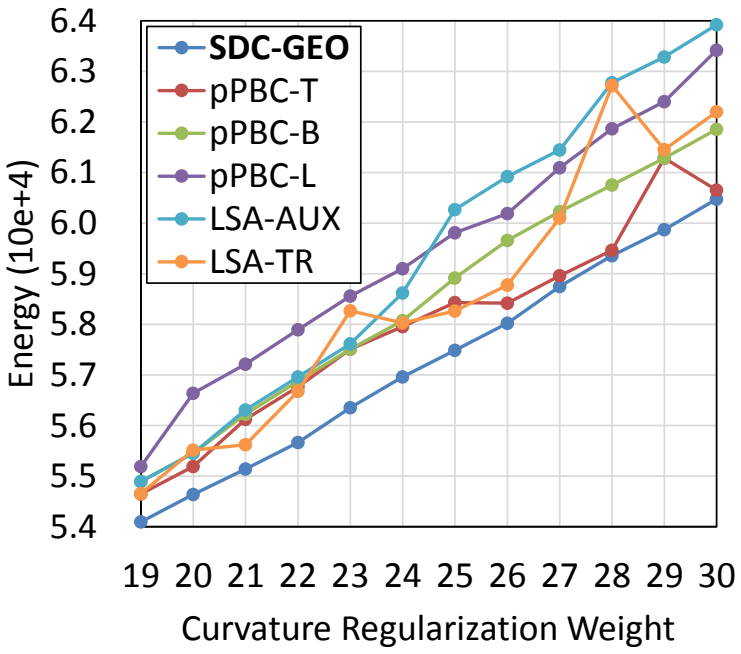


Figure 3.14 Convergence energies w.r.t. curvature regularization weights. We use $\tau = \mu/(t+1)^{2.5}$. Our method stably finds solutions that have lower energies than the others.

4

Continuous MRF Inference for Binocular Stereo Vision

STEREO vision recovers the 3D surfaces of a scene given a stereo image pair. The problem amounts to estimating the dense correspondence field between the image pair, namely, the disparity map. Being one of the oldest problems in computer vision and the most fundamental problem of dense correspondence, stereo matching has been well studied from approaches using discrete labeling formulations. However, such approaches suffer from discretization artifacts, and there is thus spontaneous demand for continuous representations leading to introducing continuous MRFs. Recent studies have also shown that the use of continuous MRFs allows us to use better matching measures and regularization enabling much higher accuracy. In this chapter, we propose a highly accurate stereo matching method, focusing on a challenging inference problem of a pairwise MRF that has a three-dimensional continuous label space.

4.1 Introduction

Recent years have seen significant progress in accuracy of stereo vision. One of the breakthroughs is the use of slanted patch matching [Bleyer *et al.*, 2011; Besse *et al.*, 2012; Lu *et al.*, 2013; Olsson *et al.*, 2013; Heise *et al.*, 2013]. In this approach, the disparity d_p of each pixel p is over-parameterized by a local disparity plane

$$d_p = a_p u + b_p v + c_p \tag{4.1}$$

defined on the image domain (u, v), and rather than directly estimating d_p, the triplet (a_p, b_p, c_p) is estimated for each pixel p. The matching window is then transformed according to this disparity plane, which produces linearly-varying disparities within the window and thereby achieves accurate photo-consistency measures between matching pixels even with large matching windows. While stereo with standard 1D discrete disparity labels [Wang and Yang, 2011; Kolmogorov and Zabih, 2002, 2001; Boykov *et al.*,

2001] can be directly solved by discrete optimizers such as graph cuts (GC) [Kolmogorov and Zabin, 2004; Boykov and Kolmogorov, 2004] and belief propagation (BP) [Yedidia *et al.*, 2000; Felzenszwalb and Huttenlocher, 2004], such approaches cannot be directly used for continuous 3D labels due to the huge or infinite label space $(a, b, c) \in \mathbb{R}^3$.

Recent successful methods [Bleyer *et al.*, 2011; Besse *et al.*, 2012; Lu *et al.*, 2013; Heise *et al.*, 2013] use PatchMatch inference [Barnes *et al.*, 2009, 2010] to efficiently infer correct 3D planes using spatial propagation; each pixel's candidate label is, in raster-scan order, refined and then propagated to next pixels. Further in [Besse *et al.*, 2012], this sequential algorithm is combined with BP yielding an efficient optimizer PMBP for pairwise Markov random fields (MRFs) [Geman and Geman, 1984]. In terms of MRF optimization, however, BP is considered a *sequential optimizer*, which improves each variable individually keeping others conditioned at the current state. In contrast, GC improves all variables simultaneously by accounting for interactions across variables, and this global property helps optimization avoid local minimas [Szeliski *et al.*, 2008; Woodford *et al.*, 2009]. In order to take this advantage and efficiently infer 3D planes by GC, it is important to use spatial propagation. Nevertheless, incorporating such spatial propagation into GC-based optimization is not straightforward, because inference using GC proceeds rather *all-nodes-simultaneously*, not *one-by-one-sequentially* like PatchMatch and BP.

In this study, we introduce a new move making scheme, *local expansion moves*, that enables spatial propagation in GC optimization. The local expansion moves are presented as many α-expansions [Boykov *et al.*, 2001] defined for a small extend of regions at different locations. Each of this small or local α-expansion tries improving the current labels in its local region in an energy minimization manner using GC. Here, those current labels are allowed to move to a candidate label α, which is given uniquely to each local α-expansion in order to achieve efficient label searching. At the same time, this procedure is designed to propagate a current label in a local region for nearby pixels. For natural scenes that often exhibit locally planar structures, the joint use of local expansion moves and GC has a useful property. It allows multiple pixels in a local region to be assigned the same disparity plane *by a single min-cut* in order to find a smooth solution. Being able to simultaneously update multiple variables also helps to avoid trapped at a bad local minima.

The advantages of our method are as follows.

- Our local expansion move method produces *submodular moves* that guarantee the optimal labeling at each min-cut (subproblem optimal), which in contrast is not guaranteed in general fusion moves [Lempitsky *et al.*, 2010].
- This optimality property and spatial propagation allow randomized search, rather than employ external methods to generate plausible initial proposals as done in fusion

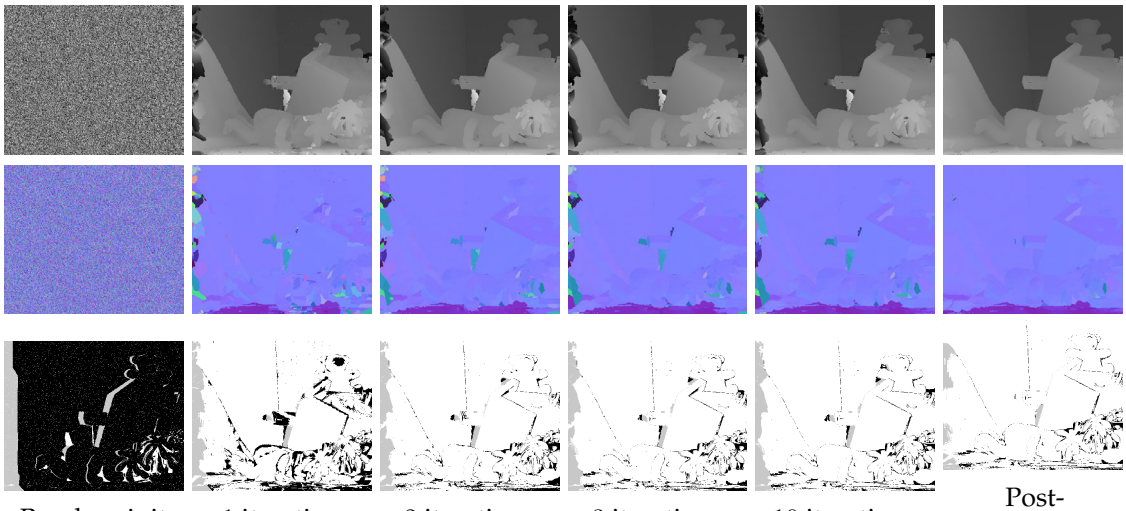


Figure 4.1 Evolution of our stereo matching estimates. From top to bottom, we show disparity maps, normal maps of disparity planes, and error maps with 0.5 pixel threshold. In our framework, we start with random disparities that are represented by per-pixel 3D planes, *i.e.*, disparities (top) and normals (middle). We then iteratively apply our local expansion moves using GC (middles) to update and propagate local disparity planes. Finally, the resulting disparity map is further refined at a post-processing stage using left-right consistency check and weighted median filtering (rightmost).

approaches [Lempitsky *et al.*, 2010; Woodford *et al.*, 2009; Olsson *et al.*, 2013], which may limit the possible solutions.

- Our method achieves greater accuracy than BP [Besse *et al.*, 2012] thanks to the good properties of GC and local expansion moves.
- Unlike other PatchMatch based methods [Bleyer *et al.*, 2011; Besse *et al.*, 2012; Heise *et al.*, 2013], our method can effectively incorporate the fast cost filtering technique of [Lu *et al.*, 2013]. In this manner, we can efficiently reduce the computation complexity of unary terms from $O(|W|)$ to approximately $O(1)$, removing the dependency from the support window size $|W|$.
- Unlike PMBP [Besse *et al.*, 2012], our method is well suited for parallelization using both CPU and GPU.[1] With multiple CPU cores, each of our local α-expansions (*i.e.*, min-cut computations) can be individually performed in parallel. With a GPU implementation, the computation of unary terms can be efficiently performed in a parallel manner for further acceleration.

This study is an extension to our conference paper [Taniai *et al.*, 2014]. The extensions are summarized as follows. We add theoretical verifications on the preferability of

[1] Although BP is usually parallelizable on GPU as well as CPU, PMBP differs from BP's standard settings in that it defines label space *uniquely and distinctively* for each pixel and *propagate* it; both make parallelization indeed non-trivial.

our method for piecewise planar scenes in Section 4.3.1, and also on the subproblem optimality of our algorithm in Section 10. Furthermore, the efficiency of our algorithm is improved by two ways; In Section 10 we show the parallelizablity of our local expansion move algorithm; In Section 4.3.5 we show that the fast cost filtering technique of [Lu *et al.*, 2013] can be used in our method. The effectiveness of both extensions is thoroughly evaluated in the experiments, and we show that even a CPU implementation of the proposed method achieves about 2.1x faster running times than our previous GPU implementation [Taniai *et al.*, 2014], with comparable or even greater accuracy.

4.2 Related Work

4.2.1 MRF Stereo Methods

MRF stereo methods can be categorized into three approaches: discrete stereo, segment-based stereo, and continuous stereo.

Discrete Stereo

Discrete stereo [Wang and Yang, 2011; Kolmogorov and Zabih, 2002, 2001; Boykov *et al.*, 2001] formulates stereo matching as a discrete multi-labeling problem, where each pixel is individually assigned one of pre-defined discrete disparity values. For this problem, many powerful discrete optimizers, such as BP [Yedidia *et al.*, 2000; Felzenszwalb and Huttenlocher, 2004], TRW-S [Kolmogorov, 2006], and GC [Kolmogorov and Zabin, 2004; Boykov and Kolmogorov, 2004], can be directly used. Successful results are shown using GC with expansion moves [Boykov *et al.*, 2001; Szeliski *et al.*, 2008]. In expansion moves, the multi-labeling problem is reduced to a sequence of binary-labeling problems, each of which can be exactly solved by GC, if only pairwise potentials ψ meet the following submodularity of expansion moves [Kolmogorov and Rother, 2007; Boykov *et al.*, 2001]:

$$\psi(\alpha,\alpha)+\psi(\beta,\gamma)\leq\psi(\beta,\alpha)+\psi(\alpha,\gamma). \tag{4.2}$$

Segment-based Stereo

Segment-based stereo [Tao *et al.*, 2001; Hong and Chen, 2004; Klaus *et al.*, 2006; Wang and Zheng, 2008] assigns a 3D disparity plane for each of over-segmented image regions. The candidate planes are generated by fitting planes to a roughly estimated disparity map, and then the optimal assignment of the planes is estimated by, *e.g.*, GC with expansion moves [Boykov *et al.*, 2001; Hong and Chen, 2004] or BP [Felzenszwalb and Huttenlocher, 2004; Klaus *et al.*, 2006]. Although this approach yields continuous-valued disparities, it strictly limits the reconstruction to a piecewise planar representation. Also, results are subject to the quality of the segmentation.

Continuous Stereo

The last group, to which our method belongs, is continuous stereo [Woodford *et al.*, 2009; Bleyer *et al.*, 2011; Besse *et al.*, 2012; Olsson *et al.*, 2013; Lu *et al.*, 2013; Heise *et al.*, 2013], where each pixel is assigned a distinct continuous disparity value.

Some methods [Woodford *et al.*, 2009; Olsson *et al.*, 2013] use fusion moves [Lempitsky *et al.*, 2010], an operation that combines two disparity maps to make a better one (binary fusion) by solving a non-submodular binary-labeling problem using QPBO-GC [Kolmogorov and Rother, 2007; Lempitsky *et al.*, 2010]. In this approach, a number of continuous-valued disparity maps (or so-called *proposals* in the literature [Lempitsky *et al.*, 2010]) are first generated by other external methods (*e.g.*, segment-based stereo [Woodford *et al.*, 2009]), which are then combined as a sequence of binary fusions. Our method differs from this fusion-based approach in that we use spatial propagation and randomization search during inference, by which we only require a randomized initial solution instead of those generated by external methods. More importantly, binary energies produced in our method are always submodular, *i.e.*, each binary-energy minimization is optimally solved via GC (subproblem optimal).

A stereo method by Bleyer *et al.* [2011] proposes accurate photo-consistency measures using 3D disparity planes that are inferred by PatchMatch [Barnes *et al.*, 2009, 2010]. Heise *et al.* [2013] incorporate Huber regularization into [Bleyer *et al.*, 2011] using convex optimization. Besse *et al.* [2012] point out a close relationship between PatchMatch and BP and present a unified method called PatchMatch BP (PMBP) for pairwise continuous MRFs. PMBP is probably the closest approach to ours in spirit, but we use GC instead of BP for the inference. Therefore, our method is able to take advantage of better convergence of GC [Szeliski *et al.*, 2008] for achieving greater accuracy. In addition, our method allows efficient parallel computation of unary matching costs.

4.2.2 Cost-Volume Filtering

Patch-based stereo methods often use cost-volume filtering for fast implementations. Generally, computing a matching cost C for a patch requires $O(|W|)$ of computation, where $|W|$ is the size of the patch. However, given a cost-volume $\rho_d(p)$ that represents pixelwise raw matching costs $\|I(p) - I'(p-d)\|$ for a certain disparity label d, the patch-based matching costs can be efficiently computed by applying a filtering to the cost-volume as $C_d(p) = \sum_q \omega_{pq}\rho_d(q)$. Here, the filter kernel ω_{pq} represents the matching window at p. If we use a constant-time filtering, each matching cost $C_d(p)$ is efficiently computed in $O(1)$.

The box filtering can achieve $O(1)$ by using integral image but results in flattening object boundaries. For this boundary issue, Yoon and Kweon [2005] propose an adaptive support-window technique that uses the joint bilateral filtering [Petschnigg *et al.*, 2004]

for cost filtering. Although this adaptive window technique successfully deals with the boundary issue [Hosni *et al.*, 2012], it involves $O(|W|)$ of computation because of the complexity of the bilateral filtering. Recently, He *et al.* [2010, 2013] propose a constant-time edge-aware filtering named the *guided image filtering*. This filtering is employed in a cost-volume filtering method of [Rhemann *et al.*, 2011; Hosni *et al.*, 2013], achieving both edge-awareness and $O(1)$ of matching-cost computation.

In principle, stereo methods using PatchMatch inference [Bleyer *et al.*, 2011; Besse *et al.*, 2012; Heise *et al.*, 2013] cannot take advantage of the cost filtering acceleration, since in those methods the candidate disparity labels are given uniquely to each pixel and we cannot make a constant-label cost-volume $\rho_d(p)$. To this issue, Lu *et al.* [2013] extend [Bleyer *et al.*, 2011] to use superpixels as a unit of cost calculations. In their method, called PatchMatch filter (PMF), fast cost-volume filtering of [Rhemann *et al.*, 2011; Hosni *et al.*, 2013] is applied in small subregions and that approximately achieves $O(1)$ of complexity. We will show in Section 4.3.5 that their subregion filtering technique can be effectively incorporated into our method, and we achieve greater accuracy than their local stereo method [Lu *et al.*, 2013].

4.3 Proposed Method

This section describes our proposed method. Given two input images I_L and I_R, our purpose is to estimate the disparities of both images.

In Section 4.3.1, we first analyze the formulation of disparity planes used in the slanted patch matching approach [Bleyer *et al.*, 2011], and show its relationship to homographies. We then define our energy function in Section 4.3.2, and describe the fundamental idea of our optimizing strategy and its properties in Section 4.3.3. The whole optimization procedure is presented in Section 4.3.4, and we further discuss a fast implementation in Section 4.3.5.

4.3.1 Geometric Interpretation to Slanted Patch Matching

Slanted patch matching [Bleyer *et al.*, 2011; Besse *et al.*, 2012; Lu *et al.*, 2013; Heise *et al.*, 2013; Taniai *et al.*, 2014] is an important technique for patch-based stereo matching, where a patch is warped and transformed in the other view using a disparity plane in Eq. (4.1). More specifically, each patch is warped by the following form of affine transformations

$$\mathbf{u}' = \begin{bmatrix} 1-a & -b & -c \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{u}, \tag{4.3}$$

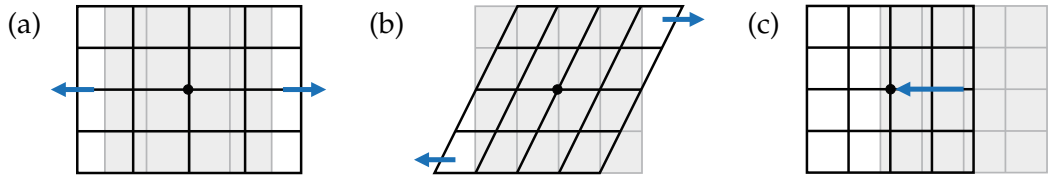


Figure 4.2 Effects of (a, b, c) in slanted patch matching [Bleyer *et al.*, 2011]. Square windows (gray) in a reference view are warped and transformed in the other view by disparity planes $d = au + bv + c$. Image courtesy of [Heise *et al.*, 2013].

where $\mathbf{u}$ and $\mathbf{u}'$ are homogeneous pixel coordinates in the left and right view images, respectively. As illustrated in Figure 4.2, this formulation produces linearly-varying disparities within a matching window, and can avoid a bias toward front-parallel surfaces that appears when using a constant disparity within a matching window [Bleyer *et al.*, 2011]. This approach implicitly assumes that the true disparity maps are piecewise linear. In this section we show the validity of this piecewise linear disparity assumption for the scenes with *piecewise planar surfaces*. As we will discuss later, the result of this section also supports our method design, since our model and inference both prefer piecewise linear disparity maps.

Let us assume a rectified stereo setting, *i.e.*, two cameras are placed at $\mathbf{x} = \mathbf{0}$ (left) and $\mathbf{x} = (B, 0, 0)^T$ (right), respectively, in the 3D world coordinates $\mathbf{x} = (x, y, z)^T \in \mathbb{R}^3$. The two cameras have the same focal length $\{f_x, f_y\}$ and the identity rotation $R = I$.

We then assume there exists a planar surface in the scene

$$a'_p x + b'_p y + c'_p z = h'_p \tag{4.4}$$

that is parameterized by (a'_p, b'_p, c'_p, h'_p) and observed at the pixel p in the left view. Using the perspective model

$$x = uz/f_x, \quad y = vz/f_y, \tag{4.5}$$

the 3D points (x, y, z) on the geometry plane of Eq. (4.4) are projected at (u, v, z) on the left image domain (u, v) as

$$z(u, v) = h'_p / \left(\frac{a'_p}{f_x} u + \frac{b'_p}{f_y} v + c'_p \right). \tag{4.6}$$

Here, $z(u, v)$ is so-called a depth map representing depth values z of the geometry plane at the left image coordinates (u, v). In the rectified stereo, it is well known that depth z and disparity d are related by

$$z = Bf_x/d. \tag{4.7}$$

By plugging this equation into Eq. (4.6), we finally obtain the geometry plane that is

transformed into the form of disparity plane of Eq. (4.1):

$$d(u,v) = \frac{Bf_x}{h'_p}\left(\frac{a'_p}{f_x}u + \frac{b'_p}{f_y}v + c'_p\right). \tag{4.8}$$

This result shows that, in the rectified stereo setting, the warping induced by a planar surface or a *homography* can be described by a single disparity plane with three degrees of freedom. Therefore, the piecewise linearity of disparity maps in the image coordinates is equivalent to the piecewise planarity of object surfaces in the world coordinates.

4.3.2 Formulation

We follow the slanted patch matching approach of [Bleyer *et al.*, 2011]. Here, each pixel p's disparity d_p is over-parameterized by a 3D plane $d_p = a_p u + b_p v + c_p$. Therefore, we seek a mapping $f_p = f(p) : \Omega \to \mathcal{L}$ that assigns a disparity plane $f_p = (a_p, b_p, c_p) \in \mathcal{L}$ for every pixel p in the left and right images. To estimate f, we use a pairwise MRF formulation by following conventional stereo matching methods [Olsson *et al.*, 2013; Wang and Yang, 2011; Kolmogorov and Zabih, 2002, 2001; Boykov *et al.*, 2001]. In the MRF framework, we seek f such that minimizes

$$E(f) = \sum_{p\in\Omega} \phi_p(f_p) + \lambda \sum_{(p,q)\in\mathcal{N}} \psi_{pq}(f_p, f_q). \tag{4.9}$$

The first term, called the *data term* or *unary term*, measures the photo-consistency between matching pixels. The disparity plane f_p defines a warp from a pixel p in one image to its correspondence in the other image. The second term is called the *smoothness term* or *pairwise term*, which penalizes discontinuity of disparities between neighboring pixel pairs $(p, q) \in \mathcal{N}$. We define these terms as below.

Data Term

To measure photo-consistencies, we use a data term that has been recently proposed by Bleyer *et al.* [2011]. The data term of p in the left image is defined as

$$\phi_p(f_p) = \sum_{s\in W_p} \omega_{ps}\, \rho(s|f_p). \tag{4.10}$$

Here, W_p is a square window centered at p. The weight ω_{ps} implements the adaptive support window proposed in [Yoon and Kweon, 2005], and is defined as

$$\omega_{ps} = e^{-\|I_L(p) - I_L(s)\|_1/\gamma}, \tag{4.11}$$

where γ is a user-defined parameter, and $\|\cdot\|_1$ represents the ℓ_1-norm.[2] Here, we assume RGB color intensities $I \in [0, 255]^3$. Note that this weight ω_{ps} will be re-defined in Section 4.3.5 for a fast implementation. Given a disparity plane $f_p = (a_p, b_p, c_p)$, the function $\rho(s|f_p)$ in Eq. (4.10) measures the pixel dissimilarity between a support pixel $s = (s_u, s_v)$ in the window W_p and its matching point in the right image

$$s' = s - (a_p s_u + b_p s_v + c_p, 0) \tag{4.12}$$

as

$$\begin{aligned}\rho(s|f_p) = (1-\alpha)\ &\min(\|I_L(s) - I_R(s')\|_1, \tau_{col}) \\ &+ \alpha\ \min(|\nabla_x I_L(s) - \nabla_x I_R(s')|, \tau_{grad}).\end{aligned} \tag{4.13}$$

Here, $\nabla_x I$ represents the x-component of the gray-value gradient of image I, and α is a factor that balances the weights of color and gradient terms. The two terms are truncated by τ_{col} and τ_{grad} to increase the robustness for occluded regions. We use linear interpolation for $I_R(s')$, and a sobel filter kernel of $[-0.5 \ \ 0 \ \ 0.5]$ for ∇_x. When the data term is defined on the right image, we swap I_L and I_R in Eqs. (4.11) and (4.13), and add the disparity value in Eq. (4.12).

Smoothness Term

For the smoothness term, we use a curvature-based, second-order smoothness regularization term [Olsson *et al.*, 2013] defined as

$$\psi_{pq}(f_p, f_q) = \max(\omega_{pq}, \epsilon)\ \min(\bar{\psi}_{pq}(f_p, f_q), \tau_{dis}), \tag{4.14}$$

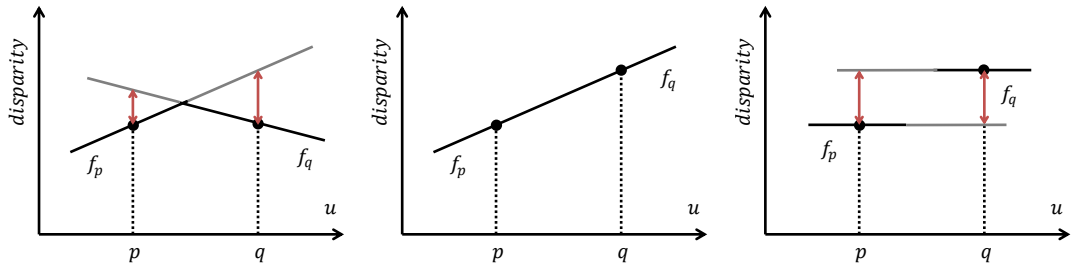
where ϵ is a small constant value that gives a lower bound to the weight ω_{pq} to increase the robustness for image noises. The function $\bar{\psi}_{pq}(f_p, f_q)$ penalizes the discontinuity between f_p and f_q in terms of disparity as

$$\bar{\psi}_{pq}(f_p, f_q) = |d_p(f_p) - d_p(f_q)| + |d_q(f_q) - d_q(f_p)|, \tag{4.15}$$

where $d_p(f_q) = a_q p_u + b_q p_v + c_q$. The first term in Eq. (4.15) measures the difference between f_p and f_q by their disparity values at p, and the second term is defined similarly at q. We visualize $\bar{\psi}_{pq}(f_p, f_q)$ as red arrows in Figure 4.3 (a). The $\bar{\psi}_{pq}(f_p, f_q)$ is truncated by τ_{dis} to allow sharp jumps in disparity at depth edges.

Notice that $\bar{\psi}_{pq}(f_p, f_q) = 2|c_p - c_q|$ when $a = b = 0$ is forced (see Figure 4.3 (c)); therefore, the smoothness term $\psi_{pq}(f_p, f_q)$ naturally extends the traditional truncated

[2] We have removed the spatial range weight of [Yoon and Kweon, 2005] that compares pixel positions. As mentioned in [Bleyer *et al.*, 2011; Hosni *et al.*, 2012], improvement due to this term is minor.



(a) $f_p \neq f_q$ (b) $f_p = f_q$ (c) $a = b = 0$

Figure 4.3 Illustration of the smoothness term proposed in [Olsson *et al.*, 2013]. (a) The smoothness term penalizes the deviations of neighboring disparity planes shown as red arrows. (b) When neighboring pixels are assigned the same disparity plane, it gives no penalty; thus, it enforces second order smoothness for the disparity maps. (c) When $a = b = 0$ is forced, this term reduces to a conventional linear model that has a front-parallel bias.

linear model [Boykov *et al.*, 2001], although it has a front-parallel bias and should be avoided [Woodford *et al.*, 2009; Olsson *et al.*, 2013]. Also, as shown in Figure 4.3 (b), this term becomes zero when $f_p = f_q$. This enforces piecewise linear disparity maps and so piecewise planar object surfaces. Furthermore, this term satisfies the following property for taking advantage of GC.

Lemma 4.1. *The term $\psi_{pq}(f_p, f_q)$ in Eq. (4.14) satisfies the submodularity of expansion moves in Eq. (4.2).*

Proof. See [Olsson *et al.*, 2013] and also Appendix B.1. □

4.3.3 Local Expansion Moves

In this section, we describe the fundamental idea of our method, local expansion moves, as the main contribution of this study. We first briefly review the original expansion move algorithm [Boykov *et al.*, 2001], and then describe how we extend it for efficiently optimizing continuous MRFs.

The expansion move algorithm [Boykov *et al.*, 2001] is a discrete optimization method for pairwise MRFs of Eq. (4.9), which iteratively solves a sequence of the following binary labeling problems

$$f^{(t+1)} = \underset{f'}{\operatorname{argmin}}\, E(f' \mid f'_p \in \{f_p^{(t)}, \alpha\}) \tag{4.16}$$

for all possible candidate labels $^\forall\alpha \in \mathcal{L}$. Here, the binary variable f'_p for each pixel p is assigned either its current label $f_p^{(t)}$ or a candidate label α. If all the pairwise terms in $E(f)$ meet the condition of Eq. (4.2), then the binary energies $E(f')$ in Eq. (4.16) are submodular and this minimization can thus be exactly solved via GC [Boykov *et al.*, 2001] (subproblem optimal). Here, it is guaranteed that the energy does not increase:

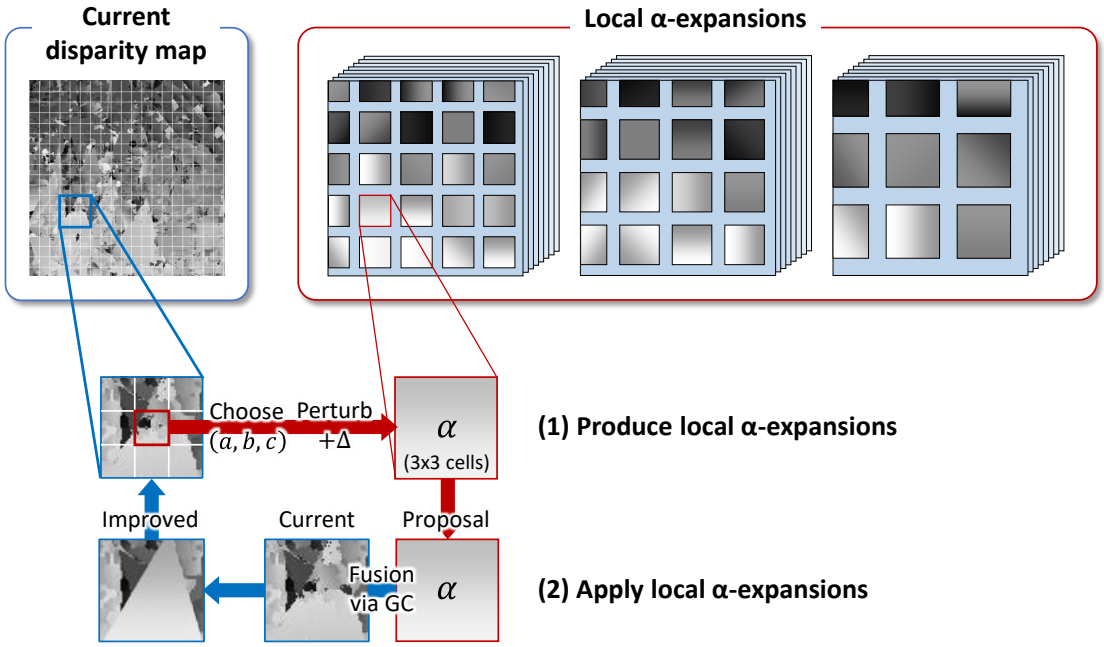


Figure 4.4 Illustration of the proposed local expansion moves. The local expansion moves consist of many small α-expansions (or *local α-expansions*), which are defined using grid structures such shown in the left part. These local α-expansions are defined at each grid-cell and applied for 3×3 cells of regions (or *expansion regions*). In the bottom part, we illustrate how each of local α-expansions works. (1) The candidate label α (*i.e.*, $\alpha = (a, b, c)$ representing a disparity plane $d = au + bv + c$) is produced by randomly choosing and perturbing one of the currently assigned labels in its center cell. (2) The current labels in the expansion region are updated by α in an energy minimization manner using GC. Consequently, a current label in the center cell of each local α-expansion can be propagated for its surrounding cells. In the right part, local α-expansions are visualized as small patches on stacked layers with three different sizes of grid structures. As shown here, using local α-expansions we can localize the scopes of label searching by their locations. Each layer represents a group of mutually-disjoint local α-expansions, which are efficiently performed individually in a parallel manner.

$E(f^{(t+1)}) \leq E(f^{(t)})$. However, the label space $\mathcal{L}$ in our setting is a three dimensional continuous space (a, b, c); therefore, such an exhaustive approach cannot be employed.

Our local expansion moves extend traditional expansion moves by two ways; *localization* and *spatial propagation*. By localization, we use different candidate labels α depending on the locations of pixels p in Eq. (4.16), rather than using the same α label for all pixels. This is reasonable because the distributions of disparities should be different from location to location, therefore the selection of candidate labels α should be accordingly different. By spatial propagation, we incorporate label propagation similar to the PatchMatch inference [Barnes *et al.*, 2009, 2010; Bleyer *et al.*, 2011] into GC optimization, and propagate currently assigned labels to nearby pixels via GC. The assumption behind this propagation is that, if a good label is assigned to a pixel, this label is likely a good estimate for nearby pixels as well. The localization and spatial propagation together

make it possible for us to use a powerful randomized search scheme, where we no longer need to produce initial solution proposals as usually done in the fusion based approach [Lempitsky *et al.*, 2010]. Below we provide the detailed descriptions of our algorithm.

Local α-Expansions for Spatial Propagation

We first define a grid structure that divides the image domain Ω into grid regions $C_{ij} \subset \Omega$, which are indexed by 2D integer coordinates $(i, j) \in \mathbb{Z}^2$. We refer to each of these grid regions as a *cell*. We assume a regular square cell and its size can be as small as 1×1 pixel. At a high level, the size of cells balances between the level of localization and the range of spatial propagation. Smaller sizes of cells can achieve finer localization but result in shorter ranges of spatial propagation. Later in Sec 4.3.4 we introduce different sizes of multiple grid structures for well balancing these two factors, but for now let us focus on using one grid structure.

Given a grid structure, we define a *local α-expansion* at each cell (i, j), which we specifically denote as *α_{ij}-expansion*. We further define two types of regions for each α_{ij}-expansion: its *center region* C_{ij} and *expansion region*

$$R_{ij} = C_{ij} \cup \left\{ \bigcup_{(m,n)\in\mathcal{N}(i,j)} C_{mn} \right\}, \tag{4.17}$$

i.e., 3×3 cells consisting of the center region C_{ij} and its eight neighbor cells.

In the bottom part of Figure 4.4, we focus on an expansion region and illustrate how an α_{ij}-expansion works. We first randomly select a pixel r from the center region C_{ij}, and take its currently assigned label as $(a, b, c) = f_r$. We then make a candidate label α_{ij} by perturbing this current label as $\alpha_{ij} = (a, b, c) + \Delta$. Finally, we update the current labels of pixels p in the expansion region R_{ij}, by choosing either their current labels f_p or the candidate label α_{ij}. Here, similarly to Eq. (4.16), we update the partial labeling by minimizing $E(f')$ with binary variables: $f'_p \in \{f_p, \alpha_{ij}\}$ for $p \in R_{ij}$, and $f'_p = f_p$ for $p \notin R_{ij}$. Consequently, we obtain an improved solution as its minimizer with a lower or equal energy.

Notice that making the expansion region R_{ij} larger than the label selection region C_{ij} is the key idea for achieving spatial propagation. We can see this since, in an α_{ij}-expansion without perturbation ($\Delta = \mathbf{0}$), a current label f_p in the center region C_{ij} can be propagated for its nearby pixels in R_{ij} as the candidate label α_{ij}.

We use this α_{ij}-expansion iteratively as shown in Algorithm 4.1. Similarity to the PatchMatch algorithm [Barnes *et al.*, 2009], this iterative algorithm consists of two steps: In the propagation step, we apply α_{ij}-expansions without perturbation to propagate labels from C_{ij} for R_{ij}; In the randomization step, we apply α_{ij}-expansions with an

Algorithm 4.1: ITERATIVE α_{ij}-EXPANSION

input : current f, target cell (i, j), perturbation size $|\Delta'|$
output: updated f (only labels f_p at $p \in R_{ij}$ are updated)

repeat propagation:
  $\alpha_{ij} \leftarrow f_r$ with randomly chosen $r \in C_{ij}$;
  $f \leftarrow \mathrm{argmin}\, E(f' \mid f'_p \in \{f_p, \alpha_{ij}\}, p \in R_{ij})$;
until K_{prop} *times*;
repeat randomization:
  $\alpha_{ij} \leftarrow f_r$ with randomly chosen $r \in C_{ij}$;
  $\alpha_{ij} \leftarrow \alpha_{ij} + \Delta'$;
  $f \leftarrow \mathrm{argmin}\, E(f' \mid f'_p \in \{f_p, \alpha_{ij}\}, p \in R_{ij})$;
  $|\Delta'| \leftarrow |\Delta'|/2$;
until K_{rand} *times (or* $|\Delta'|$ *is sufficiently small)*;

exponentially decreasing perturbation-size to refine the labels. We perform this iterative α_{ij}-expansion at every cell (i, j).

This local α-expansion method has the following useful properties.

- **Piecewise linearity**: It helps to obtain smooth solutions. In each α_{ij}-expansion, multiple pixels in the expansion region R_{ij} are allowed to move-at-once to the same candidate label α_{ij} at one binary-energy minimization, which contrasts to BP that updates only one pixel at once. Since a label represents a disparity plane here, our method helps to obtain piecewise linear disparity maps and thus piecewise planar surfaces as we have shown in Sec 4.3.1.
- **Cost filtering acceleration**: We can accelerate the computation of matching costs $\phi_p(f_p)$ in Eq. (4.10) by using cost-volume filtering techniques. We discuss this more in Sec 4.3.5.
- **Optimality and parallelizability**: With our energy formulation, it is guaranteed that each binary-energy minimization in Algorithm 4.1 can be optimally solved via GC. In addition, we can efficiently perform many α_{ij}-expansions in a parallel manner. We discuss these matters in the following sections.

Mutually-Disjoint Local α-Expansions

While the previous section shows how each local α-expansion behaves, here we discuss the scheduling of local α-expansions. We need a proper scheduling, because local α-expansions cannot be simultaneously performed due to the overlapping expansion regions.

To efficiently perform local α-expansions, we divide them into groups such that the local α-expansions in each group are mutually disjoint. Specifically, we assign each

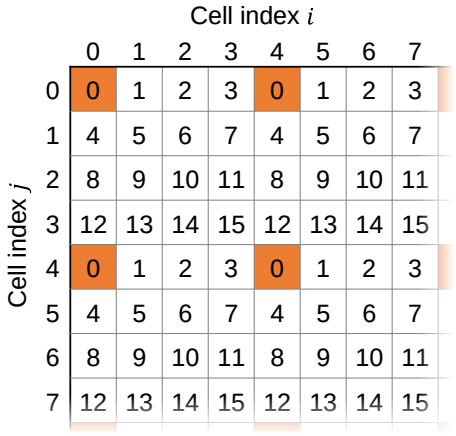


Figure 4.5 Group index for α_{ij}-expansions. We perform α_{ij}-expansions in the same group in parallel.

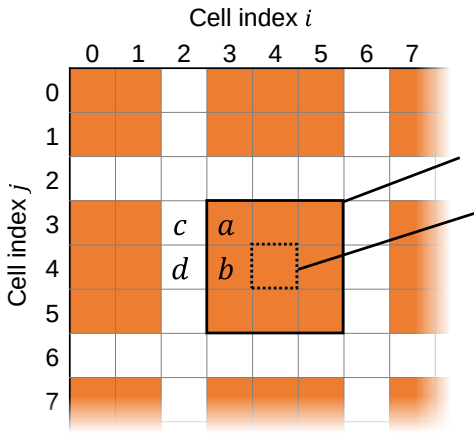


Figure 4.6 Expansion regions of mutually disjoint α_{ij}-expansions (group index $k = 0$). We leave white gaps between neighbors.

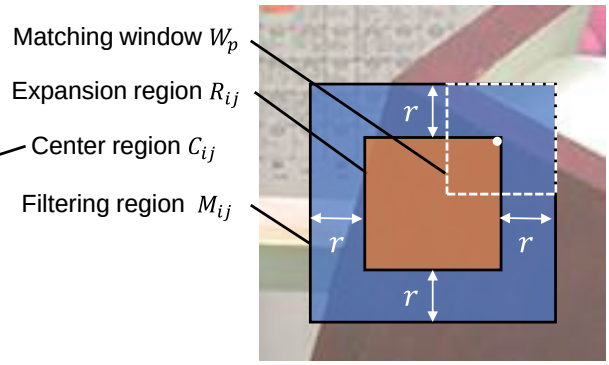


Figure 4.7 Filtering region M_{ij}. The margin width r corresponds with the radius of the matching window W_p.

α_{ij}-expansion a group index k given by

$$k = 4\,(j \bmod 4) + (i \bmod 4), \tag{4.18}$$

and perform the iterative α_{ij}-expansions in one group and another. As illustrated in Figure 4.5, this grouping rule picks α_{ij}-expansions at every four vertical and horizontal cells into the same group, and it amounts to 16 groups of mutually-disjoint local α-expansions. We visualize a group of disjoint local α-expansions as orange regions in Figure 4.6, and also as a single layer of stacks in the right part of Figure 4.4 with three different grid structures.

Notice that in each group, we leave *gaps* of one cell width between neighboring local α-expansions. These gaps are to guarantee submodularity and independency for local α-expansions. By the submodularity, we can show that our local α-expansions always produce submodular energies and can thus be optimally solved via GC. Because of this submodularity, we can use a standard GC algorithm [Boykov and Kolmogorov, 2004] instead of an expensive QPBO-GC algorithm [Kolmogorov and Rother, 2007], which is usually required in the fusion based approach. By the independency, we can show that the local α-expansions in the same group do not interfere with each other. Hence, we can perform them simultaneously in a parallel manner. The parallelization of GC algorithms are of interests in computer vision [Liu and Sun, 2010; Strandmark and Kahl, 2010]. Our scheme is simple and can use existing GC implementations. The proof of this submodularity and independency is presented in the next section.

Submodularity and Independency

To formally address the submodularity and independency of local α-expansions, we discuss it using the form of fusion moves [Lempitsky *et al.*, 2010]. Let us assume a current solution f and a group of mutually-disjoint α_{ij}-expansions to be applied. Simultaneously applying these α_{ij}-expansions is equivalent to the following fusion-energy minimization:

$$f^* = \operatorname{argmin} E(f' \mid f'_p \in \{f_p, g_p\}), \tag{4.19}$$

where the proposal solution g is set to $g_p = \alpha_{ij}$ if p belongs to any of expansion regions R_{ij} in this group; otherwise, p is in gaps so we assign $\phi_p(g_p)$ an infinite unary cost for forcing $f'_p = f_p$. This proposal disparity map g is visualized as a single layer of stacks in the right part of Figure 4.4. We prove the following lemmas:

Lemma 4.2. ***Submodularity***. *The binary energies in Eq. (4.19) are submodular,* i.e., *all the pairwise interactions in Eq. (4.19) meet the following submodularity of fusion*

moves [Lempitsky et al., 2010; Kolmogorov and Rother, 2007]:

$$\psi_{pq}(g_p, g_q) + \psi_{pq}(f_p, f_q) \leq \psi_{pq}(f_p, g_q) + \psi_{pq}(g_p, f_q). \tag{4.20}$$

Proof. All the pairwise terms $\psi_{pq}(f'_p, f'_q)$ in Eq. (4.19) can be summarized into three types: pairwise terms inside expansion regions $\psi_{ab}(f'_a, f'_b)$, inside gap regions $\psi_{cd}(f'_c, f'_d)$, and between expansion and gap regions $\psi_{ac}(f'_a, f'_c)$. Examples of the four pixels (a, b, c, d) are shown in Figure 4.6, where the cell size is shown as a 1×1 pixel here for simplicity. Notice that there is no $\psi_{pq}(f'_p, f'_q)$ that directly connects p and q in different expansion regions, since we use the eight-neighbor pairwise terms and their neighbor ranges cannot be longer than the gap width.

For $\psi_{ab}(f'_a, f'_b)$: It holds that $g_a = g_b$ because g_p is constant within an expansion region. By substituting $g_p = g_q = \alpha$, $f_p = \beta$, and $f_q = \gamma$ into Eq. (4.20), we obtain a relaxed condition known as the submodularity of expansion moves shown in Eq. (4.2). This condition holds for our pairwise term ψ_{pq}, as proved in Appendix B.1.

For $\psi_{cd}(f'_c, f'_d)$ and $\psi_{ac}(f'_a, f'_c)$: Because of the infinite unary costs of g_c and g_d, the binary variables f'_c and f'_d are forced to take their current labels f_c and f_d, respectively. Thus, $\psi_{ac}(f'_a, f'_c)$ becomes an a's unary potential $\psi_{ac}(f'_a, f_c)$, and $\psi_{cd}(f'_c, f'_d)$ becomes a constant energy $\psi_{cd}(f_c, f_d)$, both are submodular.[3] □

Lemma 4.3. ***Independency.*** *The assignments to f'_p and f'_q in Eq. (4.19) do not influence each other, if p and q are in different expansion regions.*

Proof. The f'_p and f'_q have interactions if and only if there exists a chain of pairwise interactions $C = \{\psi(f'_{s_0}, f'_{s_1}), \psi(f'_{s_1}, f'_{s_2}), \cdots, \psi(f'_{s_{n-1}}, f'_{s_n})\}$ that connects $p = s_0$ and $q = s_n$.

As discussed in the above proof of submodularity, there is no direct interaction $\psi(f'_p, f'_q)$. Therefore, such a chain must contain two types of pairwise terms $\psi_{cd}(f'_a, f'_c)$ and $\psi_{ac}(f'_c, f'_d)$ described above in order to get across the gap between p and q. These two terms become a unary potential and a constant, respectively. Therefore, there is no such chain of pairwise interactions connecting p and q. □

4.3.4 Optimization Procedure

Using the local expansion moves shown in the previous section, we present the overall optimization procedure in this section and summarize it in Algorithm 4.2.

[3] Therefore, when implementing an α_{ij}-expansion using min-cut in a subregion, we create nodes for $^{\forall}p \in R_{ij}$, and add $\psi_{ac}(f'_a, f_c)$ as unary potentials of nodes a at the inner edge of R_{ij} as well as the unary and pairwise terms defined inside R_{ij}. See also [Boykov *et al.*, 2001] for the conversion of pairwise terms into edge capacities under expansion moves.

Algorithm 4.2: OVERVIEW OF OPTIMIZATION PROCEDURE

```
1  Define three levels of grid structures: H = {h_1, h_2, h_3} ;
2  Initialize the current solution f randomly ;
3  Initialize the perturbation size |Δ| ;
4  repeat
5      foreach grid structure h ∈ H do
6          foreach disjoint group k = 0, 1, ..., 15 do
7              foreach cell (i, j) in the disjoint group k do [in parallel]
8                  Apply iterative α_ij expansion (f, (i, j), |Δ|) ;
9              end
10         end
11     end
12     |Δ| ← |Δ|/2 ;
13 until convergence;
14 Post processing ;
```

This algorithm begins with defining grid structures. Here, we use thee different sizes of grid structures for better balancing localization and spatial propagation.

At line 2 of Algorithm 4.2, the solution f is randomly initialized. To evenly sample the allowed solution space, we take the initialization strategy described in [Bleyer *et al.*, 2011]. Specifically, for each $f_p = (a_p, b_p, c_p)$ we select a random disparity z_0 in the allowed disparity range $[0, \text{dispmax}]$. Then, a random unit vector $n = (n_x, n_y, n_z)$ and z_0 are converted to the plane representation by $a_p = -n_x/n_z$, $b_p = -n_y/n_z$, and $c_p = -(n_x p_u + n_y p_v + n_z z_0)/n_z$.

In the main loop through lines 4–13, we select one grid level h from the pre-defined grid structures (line 5), and apply the iterative α_{ij} expansions of Algorithm 4.1 for each cell of the selected structure h (lines 6–10). As discussed in Sec 10, we perform the iterative α_{ij} expansions by dividing into disjoint groups defined by Eq. (4.18). Because of this grouping, the α_{ij} expansions in the loop at lines 7–9 are mutually independent and can be efficiently performed in parallel.

The perturbation at line 7 of Algorithm 4.1 is implemented as described in [Bleyer *et al.*, 2011]. Namely, each candidate label $\alpha_{ij} = (a, b, c)$ is converted to the form of a disparity d and normal vector n. We then add a random disparity $\Delta'_d \in [-r_d, r_d]$ and a random unit vector Δ'_n to them, respectively, as $d' = d_p + \Delta'_d$ and $n' = n + r_n \Delta'_n$. Finally, d' and $n'/|n'|$ are converted to the plane representation $\alpha_{ij} \leftarrow (a', b', c')$ to obtain a perturbed candidate label. The values r_d and r_n define an allowed change of disparity planes. We initialize them by setting $r_d \leftarrow \text{dispmax}/2$ and $r_n \leftarrow 1$ at line 3 of Algorithm 4.2, and update them by $r_d \leftarrow r_d/2$ and $r_n \leftarrow r_n/2$ at line 12 of Algorithm 4.2 and line 9 of Algorithm 4.1.

Finally, after the whole process, we perform post-processing using left-right consistency check and weighted median filtering as described in [Bleyer *et al.*, 2011] for further

improving the results. This step is widely employed in recent methods [Bleyer *et al.*, 2011; Besse *et al.*, 2012; Lu *et al.*, 2013; Heise *et al.*, 2013].

Note that there are mainly two differences between this algorithm and our previous version [Taniai *et al.*, 2014]. For one thing, we have removed a per-pixel label refinement step of [Taniai *et al.*, 2014]. This step is required for updating a special label space structure named *locally shared labels* (LSL) used in [Taniai *et al.*, 2014], but can be removed in our new algorithm by using local α-expansions instead. For the other, the previous algorithm proceeds rather in a batch cycle; *i.e.*, it produces all local α-expansions with all grid structures at once and then applies them to the current solution f in one iteration. This way we can minimize the overhead of data transferring between GPU in unary cost computation, but results in slower convergence than our new algorithm that produces each local α-expansion always from the latest current solution f. Our new algorithm rather concedes the increased overhead of GPU because it is also intended for a fast CPU implementation as we describe below.

4.3.5 Fast Implementation

Our method has two major computation parts: the calculations of matching costs $\phi_p(f_p)$ of Eq. (4.10), and application of GC in Algorithm 4.1. In Section 4.3.3 and Section 4.3.4, we have shown that the latter part can be accelerated by performing disjoint local α expansions in parallel. In this section, we discuss the former part.

The calculations of the matching costs $\phi_p(f_p)$ are very expensive, since they require $O(|W|)$ of computation for each term, where $|W|$ is the size of the matching window. In our previous algorithm [Taniai *et al.*, 2014], we accelerate this part by using GPU. The use of GPU is reasonable for our method because $\phi_p(f_p)$ of all pixels can be individually computed in parallel during the inference, which contrasts to PMBP [Besse *et al.*, 2012] that can only sequentially process each pixel. Still, the computation complexity is $O(|W|)$ and it becomes very inefficient if we only use CPU [Taniai *et al.*, 2014].

In the following part, we show that we can approximately achieve $O(1)$ of complexity for computing each $\phi_p(f_p)$. The key observation here is that, we only need to compute this matching cost $\phi_p(f_p)$ during the α_{ij}-expansions in Algorithm 4.1, where $\phi_p(f_p)$ is computed as $\phi_p(\alpha_{ij})$ for all $p \in R_{ij}$. With this constant-label property, we can effectively incorporate the fast subregion cost-filtering technique used in [Lu *et al.*, 2013], by which $\phi_p(\alpha_{ij})$ for all $p \in R_{ij}$ are computed altogether.

To more specifically discuss it, we first separate the computation of $\phi_p(f_p)$ into two steps: the calculations of raw matching costs

$$\rho_{f_p}(s) = \rho(s|f_p) \tag{4.21}$$

for the support pixels $s \in W_p$, and the aggregation of the raw matching costs using an

edge-aware filter kernel ω_{ps}

$$\phi_{f_p}(p) = \sum_{s \in W_p} \omega_{ps}\, \rho_{f_p}(s). \tag{4.22}$$

We also define a filtering region M_{ij} as the joint matching windows in R_{ij} as

$$M_{ij} = \bigcup_{p \in R_{ij}} W_p. \tag{4.23}$$

As shown in Figure 4.7, this M_{ij} is typically a square region margining R_{ij} with r pixels of width around R_{ij}, where r is the radius of the matching windows.

In the raw matching part of Eq. (4.21), the calculations of the raw costs $\rho_{f_p}(s), {}^{\forall}s \in W_p$ for all $p \in R_{ij}$ generally require $O(|W||R_{ij}|)$ of total computation. However, with the constant-label property (*i.e.*, $f_p = \alpha_{ij}$ for all $p \in R_{ij}$), they can be computed at once in $O(|M_{ij}|)$ by computing $\rho_{\alpha_{ij}}(s)$ for ${}^{\forall}s \in M_{ij}$. Here, the computation complexity per each unary term is $O(|M_{ij}|/|R_{ij}|)$. Therefore, if $|M_{ij}| \simeq |R_{ij}|$, we can approximately achieve $O(|M_{ij}|/|R_{ij}|) \simeq O(1)$ [Lu *et al.*, 2013].

Similarly, the cost aggregation part of Eq. (4.22) can be done in approximately $O(1)$, if we apply a constant-time edge-aware filtering ω_{ps} to $\rho_{\alpha_{ij}}(s), {}^{\forall}s \in M_{ij}$ [Lu *et al.*, 2013]. Unfortunately, our weight function w_{ps} in Eq. (4.11) represents a variant of the joint bilateral filtering [Petschnigg *et al.*, 2004], which generally requires $O(|W|)$ of computation. However, there are several constant-time edge-aware filterings that work similarly to or even better than the bilateral filtering, such as the guided image filtering [He *et al.*, 2013] and the cross-based local multipoint filterings [Lu *et al.*, 2012]. Therefore, if we replace the adaptive window weight w_{ps} of Eq. (4.11) with those filter kernels, the overall computation complexity for each unary cost $\phi_p(f_p)$ becomes approximately $O(1)$. Formally, we use the following filter kernel of the guided image filtering [He *et al.*, 2013]:

$$\omega_{ps} = \frac{1}{|W'|^2} \sum_{k:(p,s) \in W'_k} \left(1 + (I_p - \mu_k)^T (\Sigma_k + e)^{-1} (I_s - \mu_k)\right) \tag{4.24}$$

Here, $I_p = I_L(p)/255$ is a normalized color vector, μ_k and Σ_k are the mean and covariance matrix of I_p in a local regression window W'_k, and ϵ is an identity matrix with a small positive coefficient for avoiding over-fitting. This filtering can be computed in $O(1)$ using integral image.

Note that as the constant-label property is the key to being able to use the filtering techniques, this scheme cannot be used in other PatchMatch based methods [Besse *et al.*, 2012; Bleyer *et al.*, 2011; Heise *et al.*, 2013] except for [Lu *et al.*, 2013] that uses superpixels as computation units.

4.4 Experiments

In the experiments, we first evaluate our method on the Middlebury benchmark. We further assess the effect of sizes of grid-cells and also the effectiveness of the proposed acceleration schemes in comparison to our previous method [Taniai *et al.*, 2014]. Our method is further compared with the PMBP [Besse *et al.*, 2012] and PMF methods [Lu *et al.*, 2013] that are closely related to our approach.

Settings

We use the following settings throughout the experiments. We use a desktop computer with a Core i7 CPU (3.5 GHz $\times$ 4 physical cores) and NVIDIA GeForce GTX Titan Black GPU. All methods are implemented using C++ and OpenCV.

The parameters of our data term are set as $\{\tau_{col}, \tau_{grad}, \gamma, \alpha\} = \{10, 2, 10, 0.9\}$ as specified in [Bleyer *et al.*, 2011]. The size of matching windows W_p is set to 41×41, which is the same setting with PMBP [Besse *et al.*, 2012] and [Bleyer *et al.*, 2011; Heise *et al.*, 2013]. For the smoothness term, we use $\{\lambda, \tau_{dis}, \epsilon\} = \{20, 1, 0.01\}$ and eight neighbors for $\mathcal{N}$. All of these model parameters are the same with the ones we used in [Taniai *et al.*, 2014].

For our algorithm, we use three grid structures with cell sizes of 5×5, 15×15, and 25×25 pixels. The iteration numbers $\{K_{prop}, K_{rand}\}$ in Algorithm 4.1 are set to $\{1, 7\}$ for the first grid structure, and $\{2, 0\}$ (only propagation step) for the other two. We iterate the main loop ten times. We use a GC implementation of [Boykov and Kolmogorov, 2004].

We use two variants of our method. **LE-BF** uses the bilateral filtering weight ω_{ps} of Eq. (4.11) for the adaptive windows. Therefore, the computation of matching costs is as slow as $O(|W|)$. In a GPU implementation, we accelerate this calculation by computing each unary term individually in parallel on GPU. In a CPU implementation, we compute them as filtering to raw matching costs. This way we can still accelerate the computation at the first step of cost filtering. **LE-GF** uses the guided image filtering [He *et al.*, 2013] of Eq. (4.24) for w_{ps}. The size of local regression windows W'_k is set to 21×21 so that the size of actual matching windows becomes 41×41. We use $e = 0.01^2$ as specified in [Lu *et al.*, 2013; Rhemann *et al.*, 2011], and $\lambda = 1$. We only use a CPU implementation for this method. For both LE-BF and LE-GF, we perform disjoint local α-expansions in parallel using four CPU cores.

4.4.1 Evaluation on the Middlebury Benchmark

We show in Table 4.1 selected rankings on the Middlebury stereo benchmark for 0.5-pixel accuracy. The proposed LE-GF method achieves the current best average rank (3.9) and

Table 4.1 Middlebury stereo benchmark results. In the top part, we compare the proposed method (LE-GF) with our previous method (GC+LSL) [Taniai *et al.*, 2014], PM-Huber [Heise *et al.*, 2013], PMF [Lu *et al.*, 2013], PMBP [Besse *et al.*, 2012], and PatchMatch [Bleyer *et al.*, 2011]. In the bottom part, we also compare results of our LE-GF and LE-BF with and without post-processing. Error rates using the threshold of 0.5-pixel are shown. Our method using the guided image filtering (GF) achieves the current best average rank 3.9. The rankings were evaluated on April 22nd, 2015. In *all*, results are evaluated for all pixels where the ground truth is given, while only for non-occluded pixels in *nonocc*, and around depth discontinuities in *disc*.

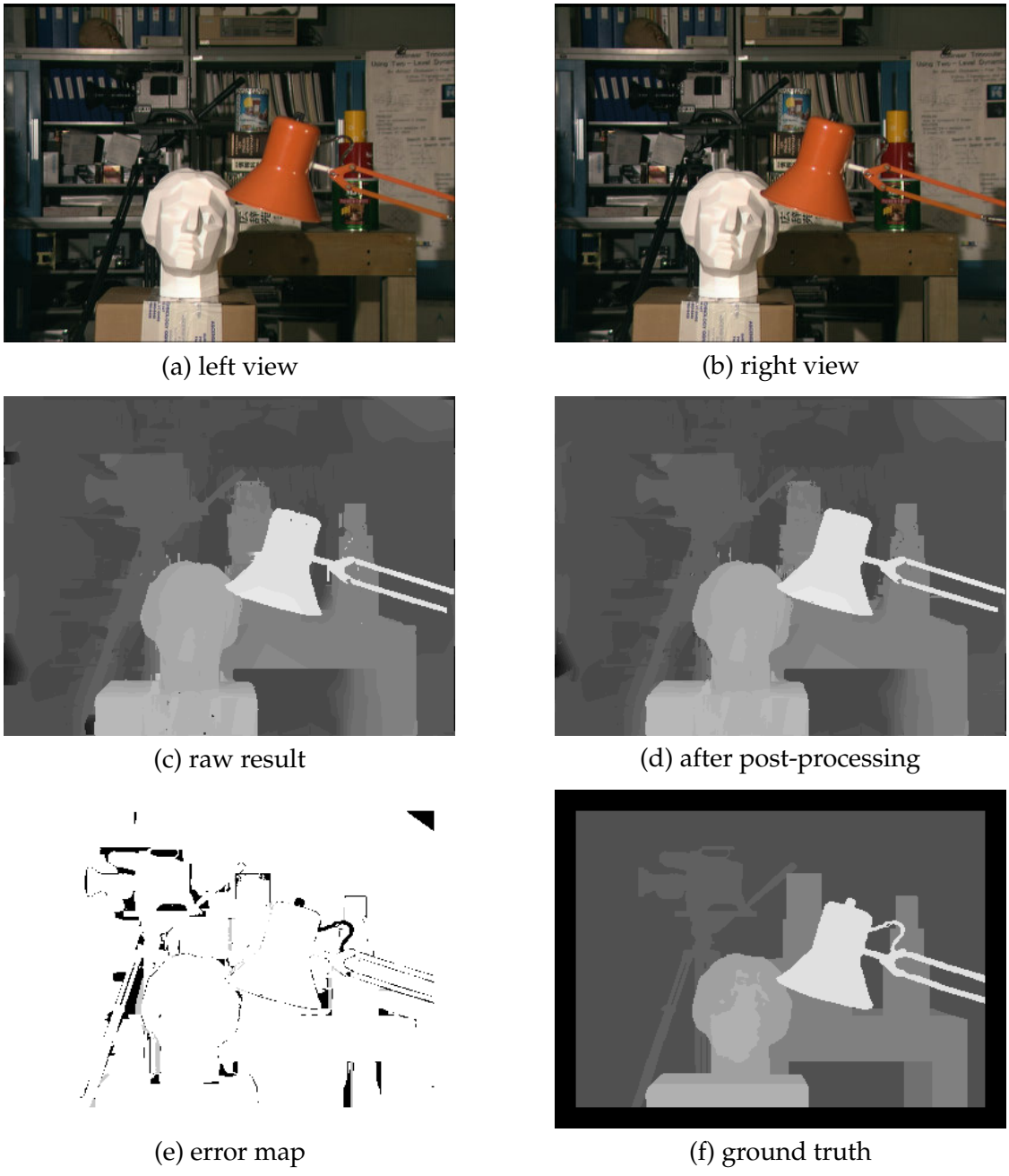
Algorithm	Avg. Rank	Tsukuba nonocc	Tsukuba all	Tsukuba disc	Venus nonocc	Venus all	Venus disc	Teddy nonocc	Teddy all	Teddy disc	Cones nonocc	Cones all	Cones disc	Average Percent Bad Pixels
1. PROPOSED (GF)	3.9	4.08 2	4.71 2	9.71 4	0.35 2	0.56 2	3.30 2	5.16 5	7.73 3	14.2 5	3.46 5	8.65 6	9.72 9	5.97
2. GC+LSL	6.2	5.04 3	5.56 3	14.0 13	0.66 6	0.88 6	5.82 8	**4.20** 1	7.12 2	12.9 3	3.77 8	9.16 9	10.4 13	6.63
3. PM-Huber	8.8	7.12 11	7.80 13	13.7 11	1.00 12	1.40 13	7.80 19	5.53 8	9.36 5	15.9 9	**2.70** 1	7.90 2	**7.77** 1	7.33
7. PMF	12.5	11.0 39	11.4 36	16.0 32	0.72 8	0.92 7	5.27 7	4.45 3	9.44 7	13.7 4	2.89 2	8.31 3	8.22 2	7.69
11. PMBP	19.7	11.9 52	12.3 48	17.8 60	0.85 10	1.10 8	6.45 11	5.60 9	12.0 12	15.5 6	3.48 6	8.88 8	9.41 6	8.77
14. PatchMatch	28.2	15.0 74	15.4 73	20.3 89	1.00 13	1.34 12	7.75 17	5.66 10	11.8 10	16.5 10	3.80 9	10.2 11	10.2 11	9.91
Reference Evaluations														
BF w/ post-proc	6.6	5.48 3	6.07 3	14.5 15	0.82 9	1.08 7	6.32 10	**4.05** 1	7.24 3	12.4 3	3.69 7	9.12 8	9.96 10	6.73
BF w/o post-proc	8.4	5.36 3	5.99 3	14.1 14	0.82 9	1.11 9	6.33 10	4.59 4	10.8 8	14.0 5	3.90 9	10.3 13	10.6 14	7.32
GF w/ post-proc	3.9	4.08 2	4.71 2	9.71 4	0.35 2	0.56 2	3.30 2	5.16 5	7.73 3	14.2 5	3.46 5	8.65 6	9.72 9	5.97
GF w/o post-proc	3.9	4.07 2	4.79 2	9.78 4	0.41 2	0.78 2	4.01 2	4.26 2	9.19 4	13.6 4	3.37 5	9.63 9	9.59 9	6.13

bad-pixel-rate (5.97%) amongst more than 150 stereo methods including our previous method (GC+LSL) [Taniai *et al.*, 2014]. Even without post-processing, our LE-GF method still outperforms the other methods in average rank, despite that methods [Bleyer *et al.*, 2011; Besse *et al.*, 2012; Lu *et al.*, 2013; Heise *et al.*, 2013; Taniai *et al.*, 2014] use the post-processing. On the other hand, the proposed LE-BF method achieves comparable accuracy with our previous algorithm [Taniai *et al.*, 2014], since both methods optimize the same energy function in a very similar way.

Compared with closely related approaches (PMBP [Besse *et al.*, 2012] and PatchMatch stereo [Bleyer *et al.*, 2011]), which are ranked eleventh and fourteenth in Table 4.1, although their results of Cones are slightly better than ours in some evaluations, our LE-GF and LE-BF methods consistently outperform the two methods in the other evaluations. We summarize the results of our LE-GF method in Figures 4.8–4.11.

4.4.2 Effect of Grid-Cell Sizes

To observe the effect of grid-cell sizes, we use the three sizes of grid-cells, 5×5 pixels (denoted as "S"mall), 15×15 pixels (denoted as "M"edium), 25×25 pixels (denoted as "L"arge), in different combinations and assess the performance using the following five different settings; (S, S, S): the small-size cells for all grid-structures; (M, M, M): the medium-size cells for all grid-structures; (L, L, L): the large-size cells for all grid-structures; (S, M, M): the small and medium-size cells for the first and the other two grid-structures, respectively; (S, M, L): the small, medium, and large-size cells for the first, second, and third grid-structures, respectively (the default setting for our method



(a) left view (b) right view

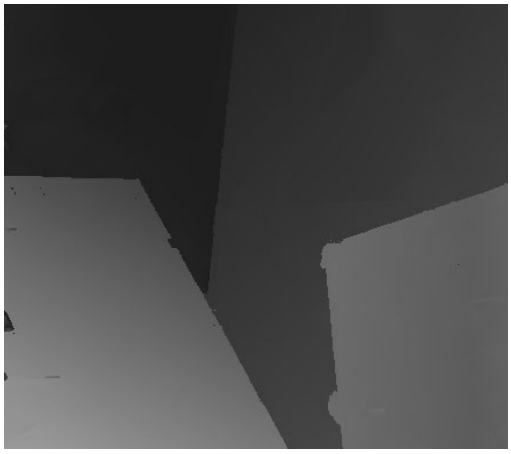
(c) raw result (d) after post-processing

(e) error map (f) ground truth

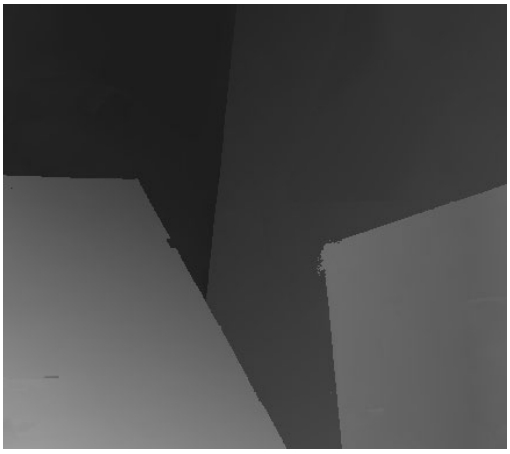
Figure 4.8 Results of the proposed LE-GF method on Tsukuba in Middlebury benchmark. From top-left to right-bottom, we show (a) left and (b) right views of input images, (c) our result before post-processing, (d) after post-processing, (e) the error map of the result after post-processing, and (f) the ground truth. In the error maps, white and black pixels indicate correct and incorrect disparities, while gray indicates incorrect but occluded pixels. The error threshold of 0.5-pixel is used. Note that Tsukuba may not be appropriate for accurate evaluations because its ground truth has only integer precision.

(a) left view

(b) right view

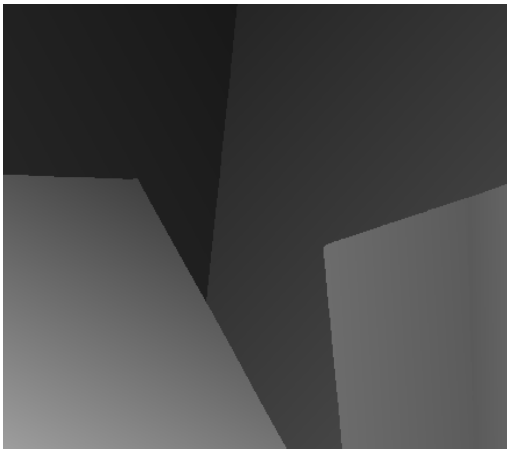


(c) raw result



(d) after post-processing

(e) error map

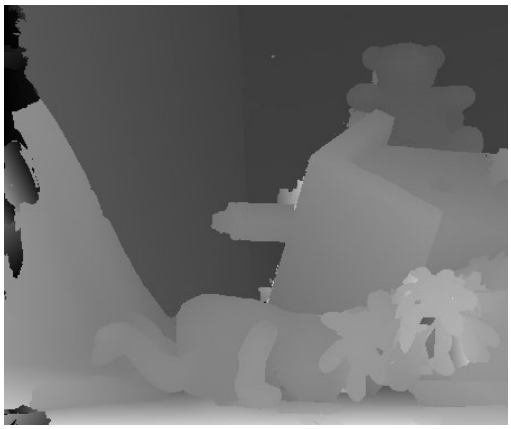


(f) ground truth

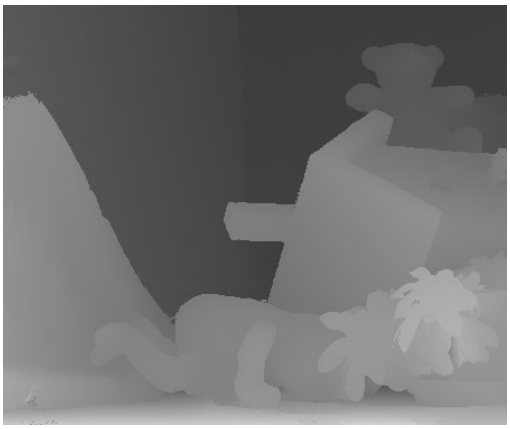
Figure 4.9 Results of the proposed LE-GF method on Venus in Middlebury benchmark. From top-left to right-bottom, we show (a) left and (b) right views of input images, (c) our result before post-processing, (d) after post-processing, (e) the error map of the result after post-processing, and (f) the ground truth. In the error maps, white and black pixels indicate correct and incorrect disparities, while gray indicates incorrect but occluded pixels. The error threshold of 0.5-pixel is used.

(a) left view

(b) right view

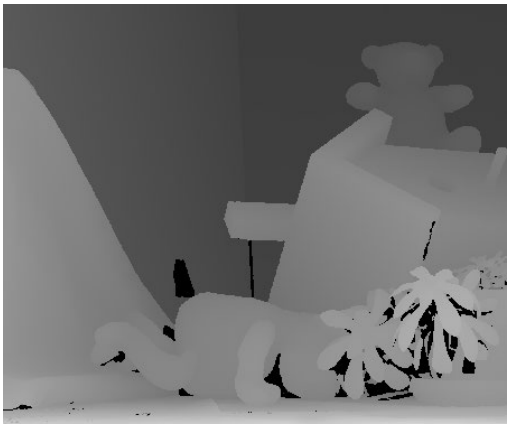


(c) raw result



(d) after post-processing

(e) error map

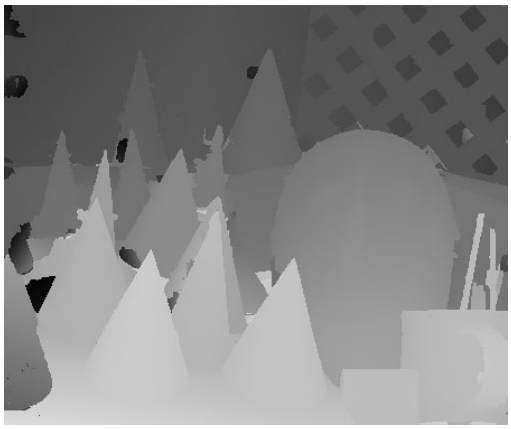


(f) ground truth

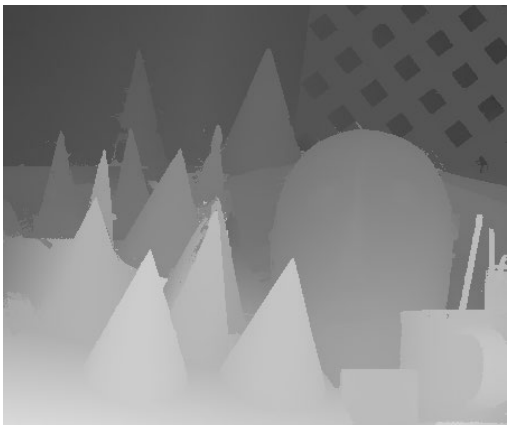
Figure 4.10 Results of the proposed LE-GF method on Teddy in Middlebury benchmark. From top-left to right-bottom, we show (a) left and (b) right views of input images, (c) our result before post-processing, (d) after post-processing, (e) the error map of the result after post-processing, and (f) the ground truth. In the error maps, white and black pixels indicate correct and incorrect disparities, while gray indicates incorrect but occluded pixels. The error threshold of 0.5-pixel is used.

(a) left view

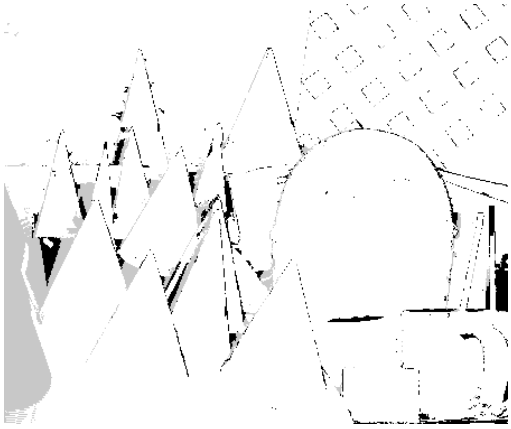
(b) right view



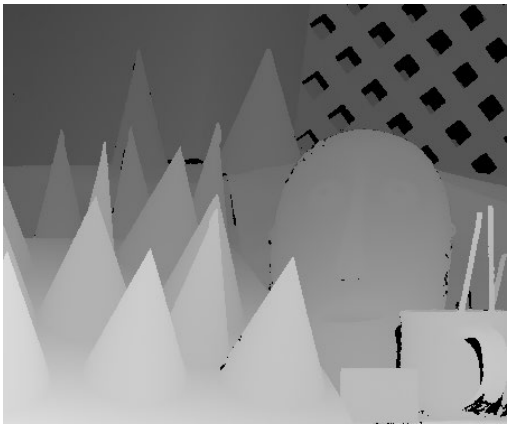
(c) raw result



(d) after post-processing



(e) error map



(f) ground truth

Figure 4.11 Results of the proposed LE-GF method on Cones in Middlebury benchmark. From top-left to right-bottom, we show (a) left and (b) right views of input images, (c) our result before post-processing, (d) after post-processing, (e) the error map of the result after post-processing, and (f) the ground truth. In the error maps, white and black pixels indicate correct and incorrect disparities, while gray indicates incorrect but occluded pixels. The error threshold of 0.5-pixel is used.

described above). Here, the iteration numbers for the first to third grid-structures are kept as default. We use the LE-GF method so as to access the effect on cost filtering acceleration as well. We use $\lambda = 0.5$ but keep the other parameters as default. Using these settings, we observe the performance variations by estimating the disparities of only the left image of the *Reindeer* dataset without post-processing.

The plots in Figure 4.12 (a) show the transitions of the energy function values over iterations. Here, energies are evaluated by the re-defined energy function using Eq. (4.24). The plots in Figure 4.12 (b) show the temporal transitions of error rates with subpixel accuracy. As shown, the joint use of multiple cell sizes improves the performance in both energy reduction and accuracy. Although (S, M, M) and (S, M, L) show almost the same energy transitions, the proposed combination (S, M, L) shows faster and better convergence in accuracy.

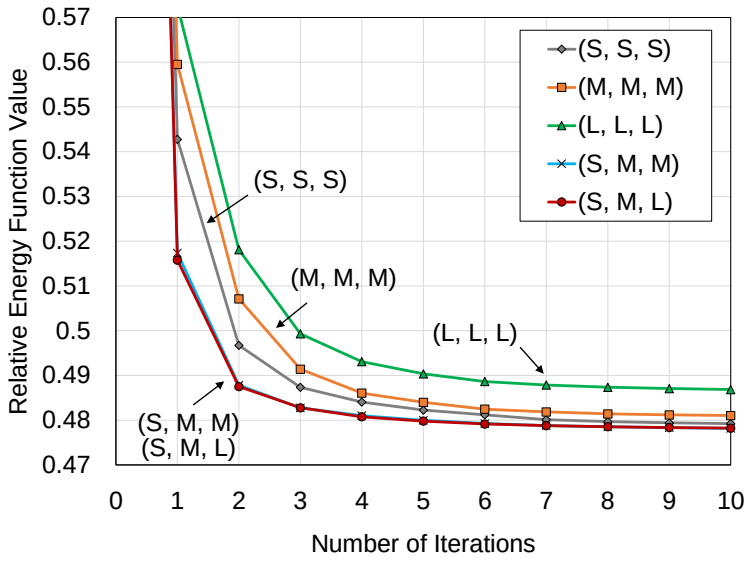
Figure 4.13 compares the results of the five settings with corresponding error maps. The use of larger grid-cells helps to obtain smoother disparities, and it is especially effective for occluded regions.

Comparing the running times in Figure 4.12 (b), the use of the small-size cells is inefficient due to the increased overhead in cost filtering, whereas the use of the medium and large-size cells achieves almost the same efficiency.

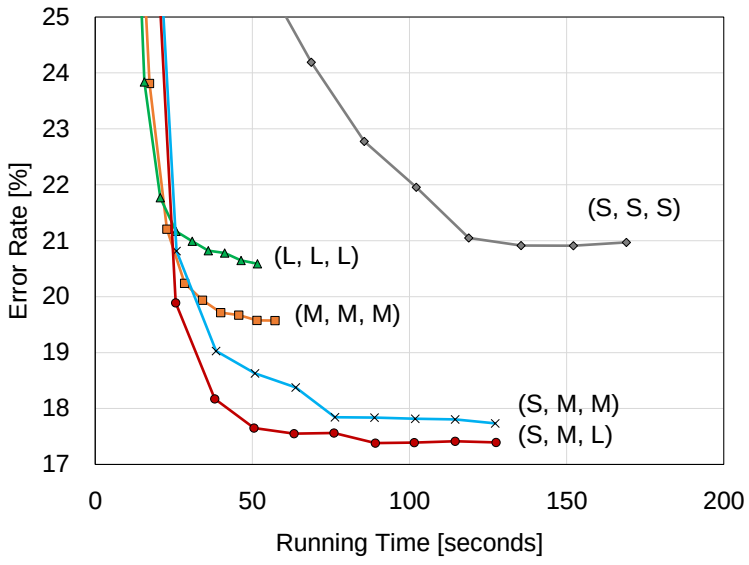
4.4.3 Efficiency Evaluation in Comparison to Our Previous Algorithm

In this section, we evaluate the effectiveness of the three acceleration techniques: 1) parallelization of disjoint α-expansions, and 2) acceleration of unary cost computation by GPU and 3) by cost filtering. For evaluation, we compare following six variants of our method: LE-BF and LE-GF using one or four CPU cores (denoted as CPUx1 and CPUx4), and LE-BF using GPU and one or four CPU cores (denoted as GPU+CPUx1 and GPU+CPUx4). Additionally, we compare with our previous algorithm of [Taniai *et al.*, 2014] with CPU and GPU implementations (denoted as LSL with CPUx1 and GPU+CPUx4). We use the *Rocks1* dataset by estimating the disparities of only the left image without post-processing. Figures 4.14 (a)–(c) show the temporal transitions of the energy values in the full and zoomed scales, and the subpixel error rates, respectively.

Parallelization of local α-expansions on CPU: Comparing CPUx1 and CPUx4, we observe about 3.5x of speed-up for both LE-BF and LE-GF. Note that the converged energy values of LE-GF are relatively higher than the others because it optimizes a different energy function. However, if we see Figure 4.14 (c), it actually finds better solutions in this example. On the other hand, speed-up for LE-BF from GPU+CPUx1 to GPU+CPUx4 is limited to about 1.7x. This is because the unary cost computation is already parallelized by GPU and this speed-up is accounted for the parallelization of other parts, *e.g.*, the computation of pairwise terms and min-cut.



(a) Energy function values w.r.t. iterations



(b) Temporal error-rate transitions

Figure 4.12 Effect of grid-cell sizes. We use LE-GF with different combinations of grid structures. The S, M, and L denote small, medium, and large grid-cells, respectively. The joint use of different sizes of grid-cells improves the performance. See also Figure 4.13 for visual comparison.

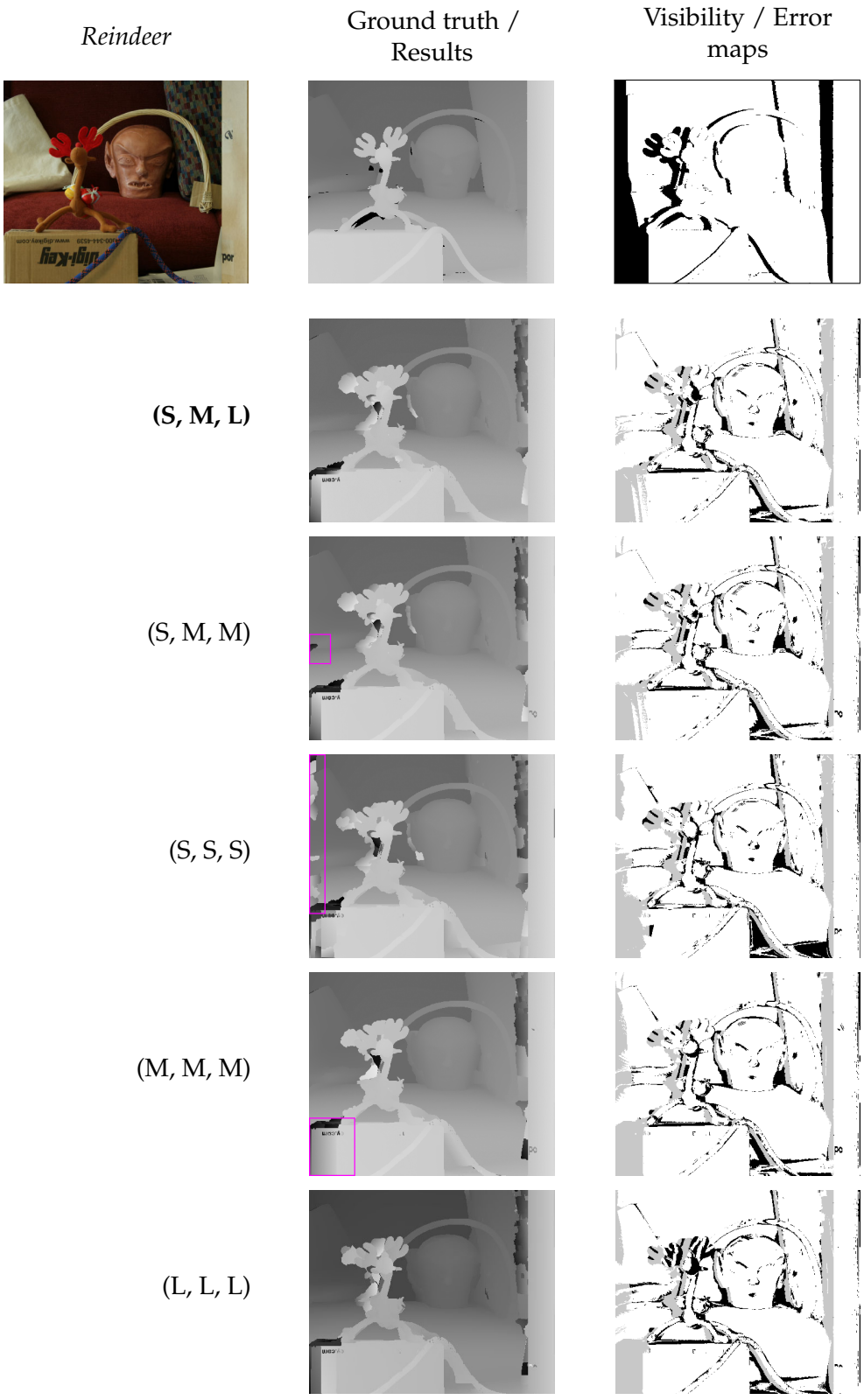


Figure 4.13 Visual effect of grid-cell sizes. The use of larger grid-cells leads to smoother solutions and effective for occluded regions. The proposed combination (S, M, L) well balances localization and spatial propagation and performs best. These are all raw results without post-processing.

Parallelization of unary cost computation on GPU: By comparing GPU+CPUx1 and CPUx1 of LE-BF, we obtain about 19x of speed-up. This is relatively smaller than 32x of speed-up in our previous LSL method, mainly due to the larger overhead of GPU data transferring.

Fast cost filtering: By comparing LE-GF and LE-BF methods, we observe about 5.3x speed-up for both CPUx1 and CPUx4 versions. It is also worth noting that even a CPU implementation of LE-GF (CPUx4) achieves almost comparable efficiency with a GPU implementation of LE-BF (GPU+CPUx1).

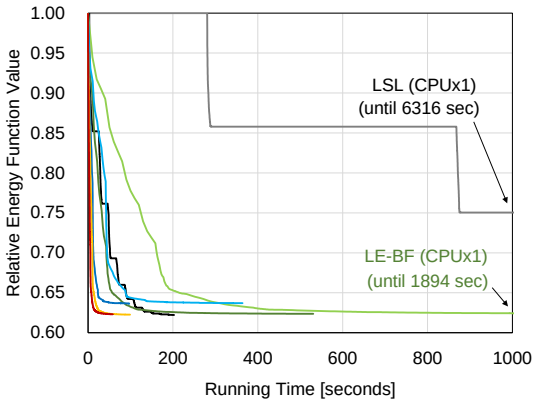
Comparison to our previous method [Taniai *et al.*, 2014]: In comparison to our previous LSL method [Taniai *et al.*, 2014], our LE-BF shows much faster convergence than LSL in the same single-core results (CPUx1). We consider there are mainly three factors contributing to this speed-up; First, the batch-cycle algorithm adopted in [Taniai *et al.*, 2014] makes its convergence slower; Second, we use cost filtering and compute its first step (*i.e.*, raw image matching) in $O(1)$, while [Taniai *et al.*, 2014] computes each unary term individually in $O(|W|)$; Finally, we have removed a per-pixel label refinement step of [Taniai *et al.*, 2014] which requires additional unary-cost computations. Similar speed-up can be observed from LSL (GPU+CPUx4) to LE-BF (GPU+CPUx4). Note that we obtain only a few percents of speed-up by CPU parallelization in [Taniai *et al.*, 2014], since it does not perform min-cut computation in parallel as disjoint local α-expansions.

4.4.4 Comparison with PMBP

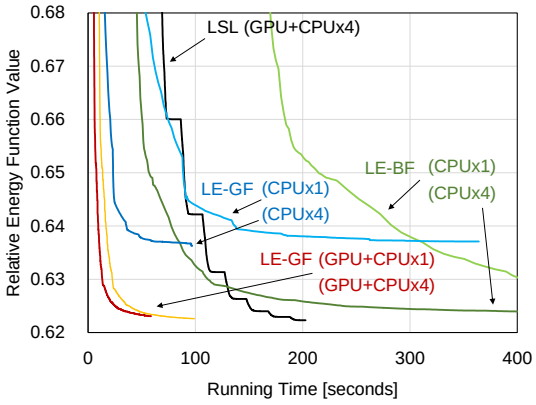
We compare our method with PMBP [Besse *et al.*, 2012] that is the closest method to ours. For a fair comparison, we use four neighbors for $\mathcal{N}$ in Eq. (4.9), which is the same setting with PMBP. For a comparable smoothness weight with the default setting (eight-neighbor $\mathcal{N}$), we use $\lambda = 40$ for LE-BF and $\lambda = 4$ for LE-GF, and keep the other parameters as default. For PMBP, we use the same model as ours; the only difference from the original PMBP is the smoothness term, which does not satisfy the submodularity of Eq. (4.2). In PMBP, it defines K candidate labels for each pixel, for which we set $K = 1$ and $K = 5$ (original paper uses $K = 5$). We show the comparison using the *Cones* dataset by estimating the disparities of only the left image without post-processing.

Figures 4.15 (a)–(c) show the temporal transitions of the energy values in the full and zoomed scales, and the subpixel error rates, respectively. We show the performance of our method using its GPU and CPU (one or four CPU cores) implementations. For PMBP, we also implemented the unary cost computation on GPU, but it became rather slow, due to the overhead of data transfer. Efficient GPU implementations for PMBP are not available in literature.[4] Therefore, the plots show PMBP results that use a single CPU

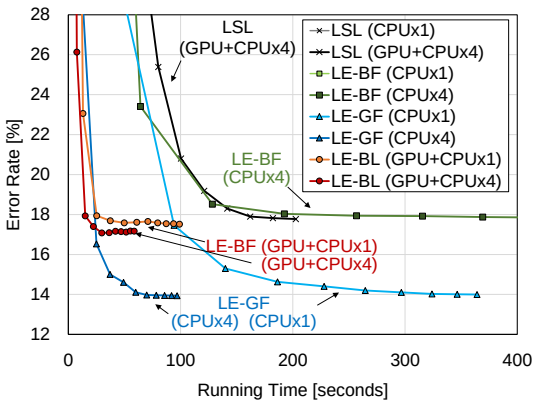
[4] GPU-parallelization schemes of BP are not directly applicable due to PMBP's unique settings. The "jump flooding" used in the original PatchMatch [Barnes *et al.*, 2009] reports 7x speed-ups by GPU. However, because it propagates candidate labels to distant pixels, it is not applicable to PMBP that must propagate



(a) Time-energy transitions (full)



(b) Time-energy transitions (zoomed)



(c) Error rate transistions

Figure 4.14 Efficiency evaluation in comparison to our previous algorithm (LSL) [Taniai *et al.*, 2014]. Accuracies are evaluated for all-regions at each iteration.

core. Figures 4.15 (a) and (b) show that, even with a single CPU-core implementation, our LE-BF and LE-GF show comparable or even faster convergence than PMBP. With CPU and GPU parallelization, our methods achieve much faster convergence than PMBP. Furthermore, our methods reach lower energies with greater accuracies than PMBP at the convergence.

In Figure 4.16 we compare the results of our LE-GF method and PMBP with $K = 5$. While PMBP yields noisy disparities, our method finds smoother and better disparities at around edges and occluded regions.

4.4.5 Comparison with PMF

We also compare our LE-GF method with PMF [Lu *et al.*, 2013] using the same data term as ours. For PMF, the number of superpixels K is set to 300, 500, and 700 as used in [Lu *et al.*, 2013] ($K = 500$ is the default in [Lu *et al.*, 2013]), and we sufficiently iterate 30 times. Both LE-GF and PMF are run using a single CPU core.

Figures 4.17 (a)–(c) show the temporal transitions of the energy values in the full and zoomed scales, and the subpixel error rates, respectively. Here, the energy values are evaluated by the re-defined weights using Eq. (4.24). As shown in Figures 4.17 (a) and (b), PMF converges at higher energies than ours, since it cannot explicitly optimize pairwise smoothness terms as being a local method. Furthermore, although energies are reduced almost monotonically in PMF, the transitions of accuracies are not stable and even degrade in same cases. This is also because of the lack of explicit smoothness regularizer, and PMF converges at a bad local minima. Figure 4.18 compares the results of our method and PMF with $K = 500$. Again, our methods find smoother and better disparities at around edges and occluded regions.

4.5 Summary

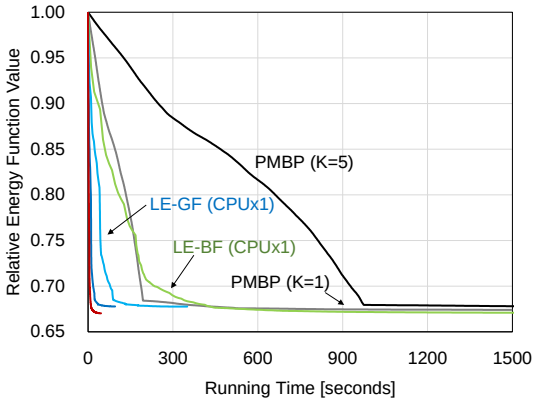
In this study, we have presented an accurate and efficient stereo matching method for continuous disparity estimation. Unlike previous approaches that use fusion moves [Lempitsky *et al.*, 2010; Olsson *et al.*, 2013; Woodford *et al.*, 2009], our method is subproblem optimal and only requires a randomized initial solution. By comparing with a recent continuous MRF stereo method, PMBP [Besse *et al.*, 2012], our method has shown an advantage in efficiency and comparable or greater accuracy. The use of a GC-based optimizer makes our method advantageous.

Furthermore, by using the subregion cost-filtering scheme developed in a local stereo method PMF [Lu *et al.*, 2013], we achieve a fast CPU implementation of our algorithm

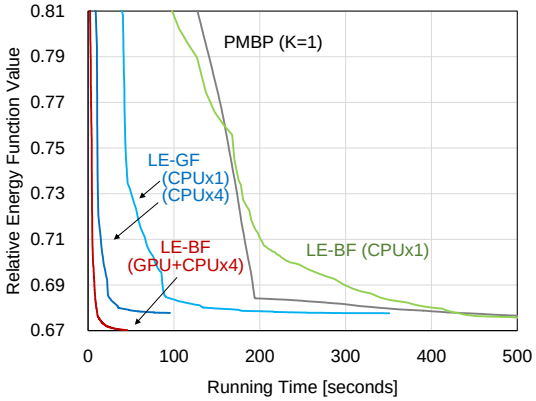
messages to *neighbors*, and is not as efficient as our 19x, either.

and greater accuracy than PMF. As shown in [Lu *et al.*, 2013], our method will be even faster by using a more light-weight constant-time filtering such as [Lu *et al.*, 2012].

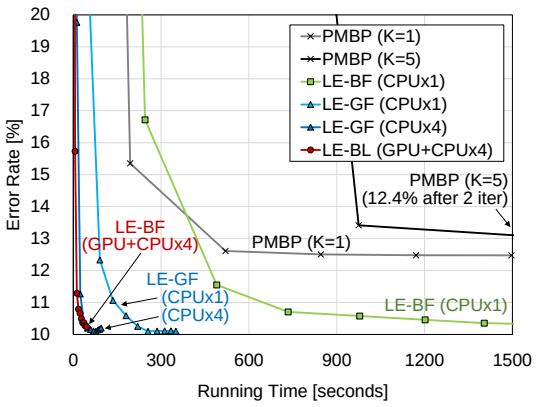
We believe that our optimization strategy can be applied for more general corresponding field estimation such as optical flow. We also believe that some occlusion handling schemes based on GC optimization [Kolmogorov and Zabih, 2001, 2002; Wei and Quan, 2005] can be incorporated into our framework, which may yield even greater accuracy.



(a) Time-energy transitions (full)

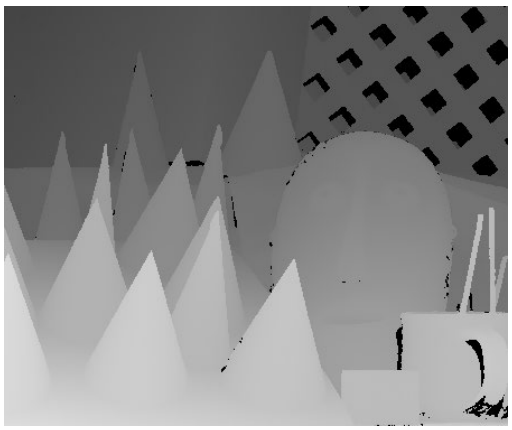


(b) Time-energy transitions (zoomed)



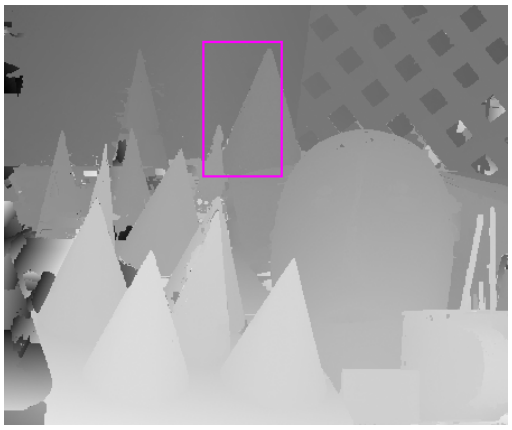
(c) Error rate transistions

Figure 4.15 Efficiency and accuracy comparison with PMBP [Besse *et al.*, 2012]. Our methods achieve much faster convergence, reaching lower energies and better accuracies at the convergence. Accuracies are evaluated for all-regions at each iteration. See also Figure 4.16 for visual comparison.

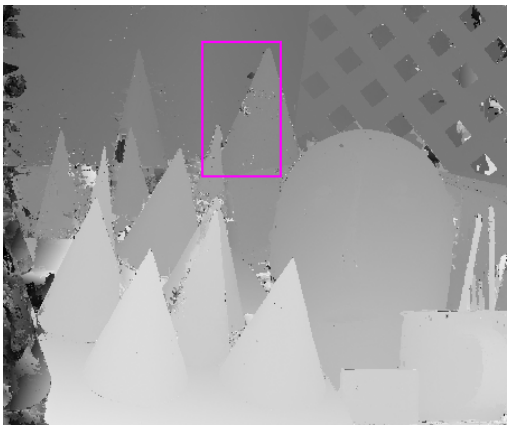


(a) Ground truth

(b) Visibility map



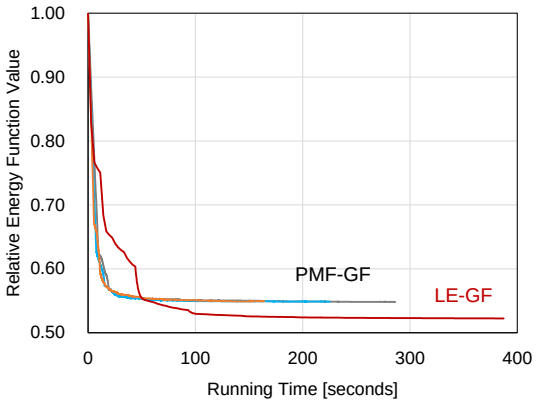
(c) Our LE-BF



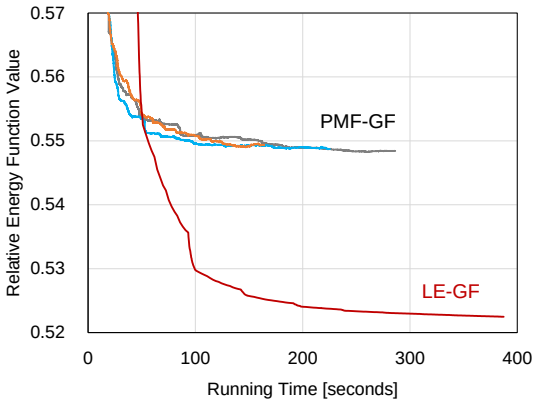
(d) PMBP

(e) Zoomed-in on the reference image, ground truth, results of ours and PMBP.

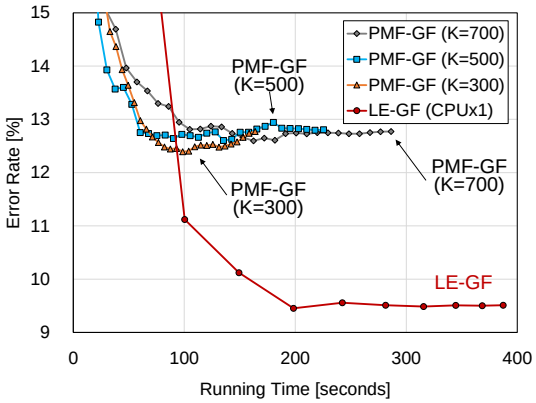
Figure 4.16 Visual comparison with PMBP [Besse *et al.*, 2012]. Our method finds smother and better disparities at around edges and occluded regions.



(a) Time-energy transitions (full)

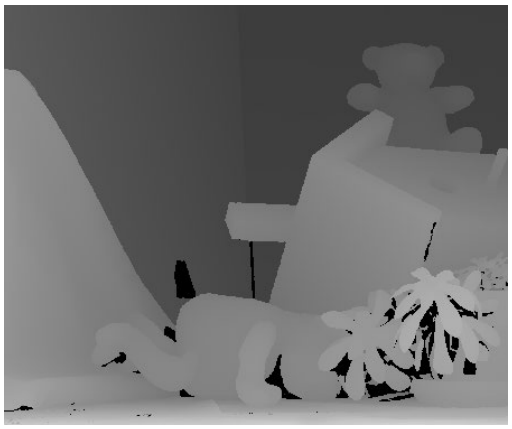


(b) Time-energy transitions (zoomed)



(c) Error rate transistions

Figure 4.17 Efficiency and accuracy comparison with PMF [Lu *et al.*, 2013]. Our method stably improves the solution and reaches a lower energy with greater accuracy at the convergence. Accuracies are evaluated for all-regions at each iteration. See also Figure 4.18 for visual comparison.

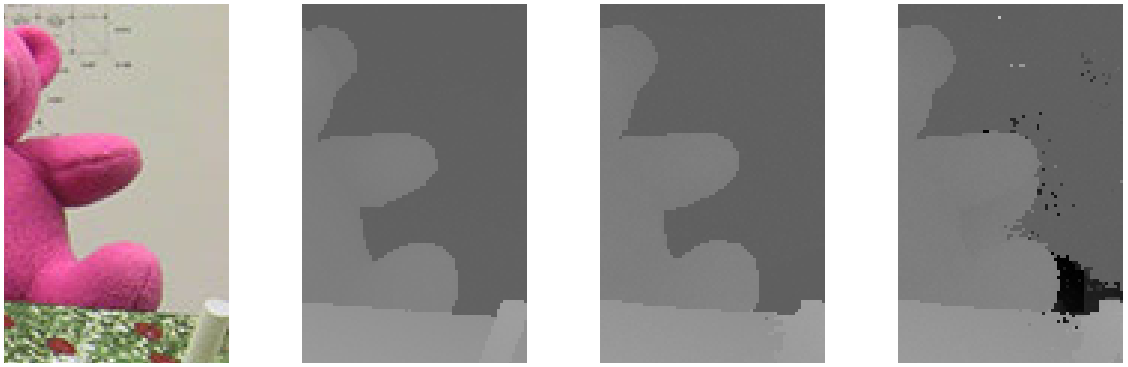


(a) Ground truth

(b) Visibility map

(c) Our LE-BF

(d) PMF



(e) Zoomed-in on the reference image, ground truth, results of ours and PMF.

Figure 4.18 Visual comparison with PMF [Lu *et al.*, 2013]. With explicit smoothness regularization, our method finds smoother and better disparities at around edges and occluded regions.

5

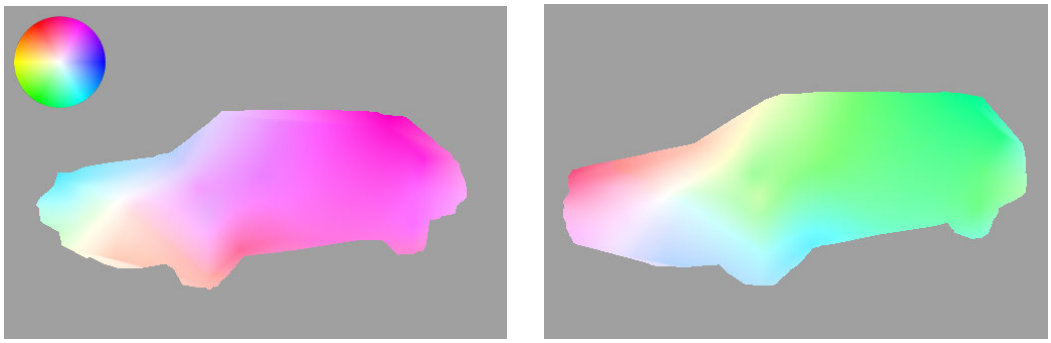
Joint Hierarchical MRF Inference for General Dense Correspondence and Cosegmentation

DENSE correspondence problems such as stereo and optical flow originally target on images showing the same scene. However, since the emergence of SIFT Flow [Liu *et al.*, 2011], more general or *semantic correspondence* estimation for images from different scenes has been gathering attentions. This generalized setting additionally brings a fundamental difficulty of robustly aligning regions of objects whose appearances may significantly differ. In this chapter, we also study a general dense correspondence problem but in a more proper form by jointly estimating valid correspondence regions or cosegmentation. We model the problem using joint labels of correspondence and cosegmentation on a MRF, and also use higher-order constraints for the robust correspondence estimation. We therefore also tackle inference challenges due to the high-dimensional label space and higher-order energies.

5.1 Introduction

Recovering dense per-pixel correspondence between image regions in two or more images is a central problem in computer vision. While correspondence estimation for images of the same scene (stereo, optical flow, etc.) is well studied, there has been growing interest in the case where the images portray semantically similar but different scenes or depict semantically related but different object instances [Liu *et al.*, 2011]. Due to the variability in appearance, shape and pose of distinct object instances, camera viewpoint, scene lighting and backgrounds in the images, the task is quite challenging in the unsupervised setting. Yet, correspondence estimation enables fine-grained image alignment crucial in tasks such as non-parametric scene parsing and label transfer [Liu *et al.*, 2011], 3D shape recovery [Vicente *et al.*, 2014], image editing [HaCohen *et al.*, 2011] and unsupervised visual object discovery [Chen *et al.*, 2014; Rubinstein *et al.*, 2013; Sivic

Input two images



Ground truth segmented flow maps

Ground truth aligned images

Estimated aligned images

Figure 5.1 Joint recovery of dense correspondence and cosegmentation where foregrounds are segmented and aligned. We show our results and corresponding ground truth from our new dataset.

et al., 2005; Tuytelaars *et al.*, 2010].

In parallel to advances in correspondence estimation, there has also been rapid progress in image cosegmentation [Faktor and Irani, 2013; Joulin *et al.*, 2014; Rother *et al.*, 2006; Tang *et al.*, 2014] methods that automatically segment similar "foreground" areas in two or more images. These methods often require the foregrounds depicting common objects to have similar region statistics. Most cosegmentation methods do not explicitly recover dense pixel correspondence and alignment in the region labeled foreground. On the other hand, correspondence estimation methods [Liu *et al.*, 2011; Kim *et al.*, 2013; Yang *et al.*, 2014; Hur *et al.*, 2015] align all the pixels without explicitly inferring which pixels in the two images actually have valid correspondence. Thus, recovering cosegmentation along with a dense alignment of the common foregrounds can be viewed as a holistic approach to solving both tasks.

In this study, we present insight into how image cosegmentation and correspondence (or flow) estimation can be tackled within a unified framework by framing it as a labeling problem (Figure 5). We show that jointly solving the two tasks in this way can improve performance on both of them. This study deals with the case where only two input images are given. The setting is unsupervised and we do not assume a priori information about the objects or the scene.

Our contributions are three folds. First, we propose a new hierarchical Markov random field (MRF) model for joint cosegmentation and correspondence recovery. The hierarchy is defined over nested image regions in the reference image and the nodes representing these regions take segmentation and flow labels. In our method, the hierarchy itself is inferred in conjunction with the labeling and is crucial for achieving robustness to dissimilar appearance of different object instances. Precomputed hierarchical structures [Kim *et al.*, 2013; Hur *et al.*, 2015; Kennedy and Taylor, 2015] are unsuitable for our task because pixels inferred as background must be excluded from matching.

Second, we propose a new optimization technique for the joint inference of the graph structure and labeling. Performing exact inference jointly on the whole hierarchical structure is intractable. In the proposed approach, layers of the hierarchy are incrementally estimated with the labeling in an energy minimization framework using iterated graph cuts [Boykov and Kolmogorov, 2004; Kolmogorov and Zabin, 2004] (alpha expansion moves).

Finally, we release a new dataset with 400 image pairs for which we provide ground truth cosegmentation masks and flow maps. The original images and some of the segmentation masks are taken from existing datasets [Lin *et al.*, 2014; Rubinstein *et al.*, 2013; Hariharan *et al.*, 2011]. The remaining segmentation masks were obtained using interactive image segmentation. The flow maps were obtained by selecting sparse keypoint correspondence with our interactive annotation tool and applying natural neighbor interpolation [Sibson, 1981] on the sparse data. Poor flow maps were discarded

by visually inspecting the flow-induced image warping result. The ground truth flow maps makes it possible to directly evaluate dense image alignment. Even SIFT Flow [Liu *et al.*, 2011] and other correspondence estimation methods [Kim *et al.*, 2013; Zhang *et al.*, 2015] are evaluated indirectly on tasks such as segmentation transfer and scene parsing using datasets that lack ground truth pixel correspondence.

The rest of this chapter is organized as follows. We describe related work in Section 5.2 and our proposed model in Section 5.3. Section 5.4 presents our optimization method whereas implementation details and the images features used are described in Section 5.5. Finally, in Section 5.6, we report experimental evaluation and comparisons with existing approaches.

5.2 Related Work

We are not aware of any existing method that explicitly solves both cosegmentation and dense correspondence recovery together. However, the motivation behind our work is similar to that behind some recent cosegmentation methods [Dai *et al.*, 2013; Faktor and Irani, 2013; Rubio *et al.*, 2012]. We review those and other broadly related works on cosegmentation and correspondence estimation.

Cosegmentation

Rubio *et al.* [2012] formulate cosegmentation in terms of region matching. However, the matches are computed independently using graph matching [Duchenne *et al.*, 2011] and then exploited in their cosegmentation algorithm. Faktor and Irani [2013] describe a model where common foregrounds in multiple images can be composed from interchangeable image regions. Although region matching is a key element of their method, it is primarily used to estimate unary potentials (foreground/background likelihoods) for a standard image segmentation method. While, Dai *et al.* [2013] propose to cosegment images by matching foregrounds through a codebook of deformable shape templates, it involves learning a codebook requiring external background images. While a notion of correspondence implicitly exists in all these works, none of them explicitly compute dense correspondence maps between the cosegmented regions, which is an important distinction to our work.

Cosegmentation methods originally proposed by Rother *et al.* [2006] have been applied in broader settings [Batra *et al.*, 2010; Joulin *et al.*, 2010, 2012; Kowdle *et al.*, 2012; Kim *et al.*, 2011; Vicente *et al.*, 2010] and also on large sets of Internet images [Rubinstein *et al.*, 2013; Chen *et al.*, 2014]. Interesting convex formulations have also been proposed for a variant of cosegmentation – the object co-localization task [Joulin *et al.*, 2014], which aims to find a bounding box around related objects in multiple images.

Correspondence Estimation

SIFT Flow [Liu *et al.*, 2011] generalizes optical flow to images of different scenes and estimates complete flow maps with 2D translations at every pixel. Their energy function uses local matching costs based on dense SIFT features, and smoothness terms encoding standard pairwise potentials. SIFT Flow uses loopy belief propagation (BP) [Felzenszwalb and Huttenlocher, 2006] for inference in a coarse-to-fine pipeline but other inference techniques [HaCohen *et al.*, 2011; Yang *et al.*, 2014] have also been explored. HaCohen et al. [HaCohen *et al.*, 2011] propose an extension of PatchMatch [Barnes *et al.*, 2009, 2010] that handles images of identical scenes with large motions. However, their method is often unable to handle different scenes as it lacks regularization on correspondence fields. As another extension, DAISY filter flow (DFF) [Yang *et al.*, 2014] proposes to use efficient cost-volume filtering [Lu *et al.*, 2013] for enforcing smoothness, instead of adding explicit regularization. Deformable spatial pyramid (DSP) matching [Kim *et al.*, 2013] and its generalization [Hur *et al.*, 2015] propose hierarchical regularization using a regular grid-cell pyramid for flow estimation. Correspondence maps are parameterized using similarity transformations in [Hur *et al.*, 2015] similarly to our work. Images with scale differences are handled by [Qiu *et al.*, 2014; Hassner *et al.*, 2012; Hur *et al.*, 2015]. Cosegmentation has been used to guide sparse feature correspondence recovery [Cech *et al.*, 2010]. However, such methods do not aim to accurately segment common regions.

Hierarchical Models

To exploit multi-scale image cues or to add flexible regularization, hierarchical conditional random fields (CRFs) have been proposed for single image segmentation [He *et al.*, 2004], image matching [Todorovic and Ahuja, 2008], stereo correspondence [Lei *et al.*, 2006], and much recently for optical flow [Lei and Yang, 2009; Kennedy and Taylor, 2015] and more general correspondence estimation [Kim *et al.*, 2013; Hur *et al.*, 2015]. These methods use precomputed hierarchical structures such obtained by an external hierarchical oversegmentation method [Arbelaez *et al.*, 2011], or spatial pyramids as used in DSP [Kim *et al.*, 2013; Hur *et al.*, 2015].

Optimization Techniques

Discrete optimization is commonplace in stereo but often problematic in general dense correspondence estimation because of the large label spaces involved. For this issue, SIFT Flow [Liu *et al.*, 2011] performs hierarchical BP [Felzenszwalb and Huttenlocher, 2006] on the image pyramid from coarse to fine levels using limited translation ranges. Recently, inspired by randomization search and label propagation schemes of PatchMatch [Barnes *et al.*, 2009, 2010; Bleyer *et al.*, 2011], optimization methods using BP [Besse *et al.*, 2014] or graph cuts [Taniai *et al.*, 2016a, 2014] (Chapter 4) have been proposed for efficient

inference in pairwise MRFs with large label spaces. However, they are not directly applicable to our hierarchical model. We extend graph cut techniques [Taniai *et al.*, 2016a, 2014] for our inference task where we recover both the graph structure as well as the labeling.

5.3 Proposed Model

Given two images $\mathbf{I}$ and $\mathbf{I}'$ our goal is to find dense correspondence and cosegmentation of a common object shown in the two images. The reference image $\mathbf{I}$ is represented by a set of superpixel nodes $i \in V$ where $\Omega_i \subseteq \Omega$ denotes a superpixel region in the image domain $\Omega \subset \mathbb{Z}^2$.

In the reference image, we seek a labeling involving a geometric transformation $\mathbf{T}_i \in \mathcal{T}$ and a foreground alpha-matte value $\alpha_i \in [0, 1]$ for each superpixel $i \in V$. We formulate this as a mapping function $f_i = f(i) : V \to \{\mathcal{T} \times [0, 1]\}$ that assigns each node a pair of labels $f_i = (\mathbf{T}_i, \alpha_i)$. Here, α_i is continuous during inference and binarized at the final step[1]. $\mathbf{T}_i$ denotes a similarity transform parameterized using a quadruplet (t_u, t_v, s, r). Slightly abusing the notation, we express the warped pixel location of $\mathbf{p}'$ in the other image as follows.

$$\mathbf{p}' = \mathbf{T}_i(\mathbf{p}) = sR_r(\mathbf{p} - \mathbf{c}_i) + \mathbf{c}_i + \mathbf{t}. \tag{5.1}$$

Here, $\mathbf{c}_i$ is the centroid of pixels in region Ω_i, and centering at this point, $\mathbf{p}$ is rotated by the 2D rotation matrix R_r of angle r and scaled by s, and then translated by $\mathbf{t} = (t_u, t_v)$.

In following sections we present the proposed model, by first defining a standard 2D MRF model in Section 5.3.1 and later generalizing it to a hierarchical model in Section 5.3.2. We discuss the allowed hierarchical structure in Section 5.3.3.

5.3.1 Single Layer Model

Let $L = (V, E)$ be a graphical representation of the image $\mathbf{I}$, where nodes $i \in V$ and edges $(i, j) \in E$ represent superpixels and spatial neighbors, respectively. Given this graph, our single layer model is defined as a standard 2D MRF model:

$$\mathcal{E}_{\text{mrf}}(f|L) = \lambda_{\text{flo}} \sum_{i \in V} \mathcal{E}^i_{\text{flo}}(f_i) + \lambda_{\text{seg}} \sum_{i \in V} \mathcal{E}^i_{\text{seg}}(f_i) + \sum_{(s,t) \in E} w_{st}\, \mathcal{E}^{st}_{\text{reg}}(f_s, f_t), \tag{5.2}$$

which consists of the flow data term, cosegmentation data term and the spatial regularization term described below.

[1] To avoid degenerate flow solutions, we set α_i always larger than 0.1 during inference. See Appendix C.1 for a detailed explanation.

Flow Data Term

$\mathcal{E}^i_{\text{flo}}$ measures similarity between corresponding regions in the image pair. We define it as

$$\mathcal{E}^i_{\text{flo}}(f_i) = \sum_{\mathbf{p}\in\Omega_i} \Big[\alpha_i \rho(\mathbf{p}, \mathbf{p}') + \bar{\alpha}_i \lambda_{\text{occ}}\Big], \tag{5.3}$$

where $\bar{\alpha}_i = 1 - \alpha_i$ and λ_{occ} is a constant penalty for background pixels to avoid trivial solutions where all pixels are labeled background. The $\rho(\mathbf{p}, \mathbf{p}')$ robustly measures visual dissimilarity between $\mathbf{p}$ and its correspondence $\mathbf{p}'$ as

$$\rho(\mathbf{p}, \mathbf{p}') = \min\{\|\mathbf{D}(\mathbf{p}) - \mathbf{D}'(\mathbf{p}'))\|_2^2, \tau_{\text{D}}\}, \tag{5.4}$$

where truncation using the threshold τ_{D} adds robustness. $\mathbf{D}(\mathbf{p})$ is a local feature descriptor extracted at $\mathbf{p}$ in the image $\mathbf{I}$, and $\mathbf{D}'(\mathbf{p}')$ is extracted in $\mathbf{I}'$[2]. We use a variant of the HOG descriptor [Dalal and Triggs, 2005]. See Section 5.5.1 for the details.

Cosegmentation Data Term

The foreground and background likelihoods for each node are defined as follows.

$$\mathcal{E}^i_{\text{seg}}(f_i) = -\sum_{\mathbf{p}\in\Omega_i} \Big[\alpha_i \ln P(\mathbf{I}_\mathbf{p}|\boldsymbol{\theta}^F) + \bar{\alpha}_i \ln P(\mathbf{I}_\mathbf{p}|\boldsymbol{\theta}^B)\Big]. \tag{5.5}$$

Here, $P(\cdot|\boldsymbol{\theta})$ is likelihood given a foreground or background color model $\{\boldsymbol{\theta}^F, \boldsymbol{\theta}^B\}$ of the image $\mathbf{I}$, which is implemented as 64^3 bins of RGB color histograms. The color models are estimated during initialization (Section 5.5.3).

Spatial Regularization Term

The term $\mathcal{E}^{st}_{\text{reg}}$ encourages flow and alpha values of neighboring nodes to be similar.

$$\mathcal{E}^{st}_{\text{reg}}(f_s, f_t) = \lambda_{\text{st1}} \min\{\alpha_s, \alpha_t\} \sum_{\mathbf{p}\in B_{st}} \psi^{st}(\mathbf{p})/|B_{st}| + \lambda_{\text{st2}}|\alpha_s - \alpha_t|. \tag{5.6}$$

Here, $B_{st} = \partial\Omega_s \cap \partial\Omega_t$ is the set of pixels on the boundary of two adjoining regions Ω_s and Ω_t, and $\psi^{st}(\mathbf{p})$ penalizes flow discontinuities at these pixels. It is defined as

$$\psi^{st}(\mathbf{p}) = \min\{\|\mathbf{T}_s(\mathbf{p}) - \mathbf{T}_t(\mathbf{p})\|_2, \tau_{\text{st}}\}. \tag{5.7}$$

If α were binary, then the first term in Eq. (5.6) would enforce flow smoothness when two adjoining regions are labeled foreground ($\alpha_s = \alpha_t = 1$), and the second term would

[2] As suggested in [Hur *et al.*, 2015; Yang *et al.*, 2014], $\mathbf{D}'(\mathbf{p}')$ can be more accurately computed by using the scale s and rotation r of the similarity transformation $\mathbf{T}_i$.

give a constant penalty λ_{st2} only when $\alpha_s \neq \alpha_t$. However, Eq. (5.6) generalizes this idea to continuous valued α.

5.3.2 Hierarchical Model

Now we introduce the notion of a layered graph and generalize the single layer model to a full hierarchical model. As illustrated in Figure 5.2, our hierarchical graph $G = (V, E)$ consists of multiple layered subgraphs $\{L_0, L_1, \cdots, L_H\}$. Each layer $L_l = (V_l, E_l)$ represents a superpixel graph of the image. In addition to spatial edges within each layer E_l, our hierarchy G contains parent-child edges $(p, c) \in E_l^{\text{pc}}$ that connect parent nodes $p \in V_l$ to their children nodes $c \in V_{l-1}$ (green edges in Figure 5.2).

Using a layered graph G and the model $\mathcal{E}_{\text{mrf}}(f|L)$ defined in Eq. (5.2), we define our hierarchical model as

$$\mathcal{E}(f, G) = \sum_{l=0}^{H} \Big[\mathcal{E}_{\text{mrf}}(f|L_l) + \mathcal{E}_{\text{reg}}^{l}(f|G) + \mathcal{E}_{\text{gra}}^{l}(V_l) \Big]. \tag{5.8}$$

Here, we treat the hierarchical graph G as a variable that is dynamically estimated together with f. Our construction is fundamentally different from prior work [Hur *et al.*, 2015; Kennedy and Taylor, 2015; Kim *et al.*, 2013], where the hierarchical structure is computed before flow inference.

Multi-layer Regularization Term

Similar to the spatial regularization term in Eq. (5.6), the term $\mathcal{E}_{\text{reg}}^{l}$ enforces smoothness between parent child pairs of V_l and V_{l-1} as

$$\mathcal{E}_{\text{reg}}^{l}(f|G) = \sum_{(p,c) \in E_l^{\text{pc}}} w_{pc}\, \mathcal{E}_{\text{reg}}^{pc}(f_p, f_c), \tag{5.9}$$

where $\mathcal{E}_{\text{reg}}^{pc}$ is defined using Eq. (5.7) and c's centroid $\mathbf{c}_c$ as

$$\mathcal{E}_{\text{reg}}^{pc}(f_p, f_c) = \lambda_{\text{pc1}} \min\{\alpha_p, \alpha_c\} \psi^{pc}(\mathbf{c}_c) + \lambda_{\text{pc2}} |\alpha_p - \alpha_c|. \tag{5.10}$$

Graph Validity Term

The term $\mathcal{E}_{\text{gra}}^{l}$ measures validity of the layer structure V_l as

$$\mathcal{E}_{\text{gra}}^{l}(V_l) = \lambda_{\text{nod}} \beta^l |V_l| - \lambda_{\text{col}} \sum_{i \in V_l} \sum_{\mathbf{p} \in \Omega_i} \ln P(\mathbf{I}_{\mathbf{p}} | \boldsymbol{\theta}^i). \tag{5.11}$$

The first term reduces nodes in the higher layers. We set $\beta = 2$ to reduce the node count approximately by half at each layer. The second term enforces color consistencies within

each region Ω_i. $\boldsymbol{\theta}^i$ represents the RGB color histogram of the region Ω_i. Our definition of Eq. (5.11) is motivated by work in multi-region segmentation [Delong *et al.*, 2012].

5.3.3 Hierarchical Structure

Here we describe the form of hierarchical graphs allowed in our method. The nodes $i \in V_l$ in each layer divide the image domain Ω into $|V_l|$ connected regions $\Omega_i \subseteq \Omega$. Our hierarchical superpixels have a nested (or tree) structure, *i.e.*, a superpixel (parent) in a layer V_l consists of the union of superpixels (children) in its sublayer V_{l-1}. The lowest layer V_0 named the *pixel layer* is special because each node $i \in V_0$ represents a pixel $\mathbf{p}_i \in \Omega$. The finest region layer V_1 has about 500 nodes which are set to SLIC superpixels [Achanta *et al.*, 2012].

For parent-child edges $(p, c) \in E_l^{\mathrm{pc}}$ $(l = 1, \cdots, H)$, the edge weights w_{pc} are assigned to the area of child regions

$$w_{pc} = |\Omega_c|. \tag{5.12}$$

At the two lowest layers $(l = 0, 1)$, edges $(s, t) \in E_l$ between adjoining nodes are assigned edge weights w_{st} as

$$w_{st} = e^{-\|\mathbf{I}_s - \mathbf{I}_t\|_2^2/\kappa}, \tag{5.13}$$

where $\mathbf{I}_i \in \mathbb{R}^3$ is the mean color of the region Ω_i. Following [Rother *et al.*, 2004], we set κ to the expected value of $2\|\mathbf{I}_s - \mathbf{I}_t\|_2^2$ over $(s, t) \in E_l$. For the upper layers $(l \geq 2)$, the edge weights w_{st} are set to the sum of the children's edge weights as

$$w_{st} = \sum w_{s't'}, \tag{5.14}$$

where $(s', t') \in E_{l-1}$ are children of s and t, respectively.

5.4 Optimization

Optimizing $\mathcal{E}(f, G)$ in Eq. (5.8) has two main difficulties. 1) The joint inference of f and G is intractable due to the dependency of f on G. 2) The label space of f resides in a 5-dimensional continuous domain and the number of candidate labels is essentially infinite. To practically address these issues, we propose two-pass bottom-up and top-down optimizing procedures that approximately optimize the energy. In the bottom-up phase, we construct a hierarchical structure G by incrementally adding layers from lower to higher levels, while simultaneously estimating the labeling f. In the top-down phase, we refine the labeling f while keeping the structure G fixed. The optimization procedure is summarized in Algorithm 5.1. Next we discuss the details.

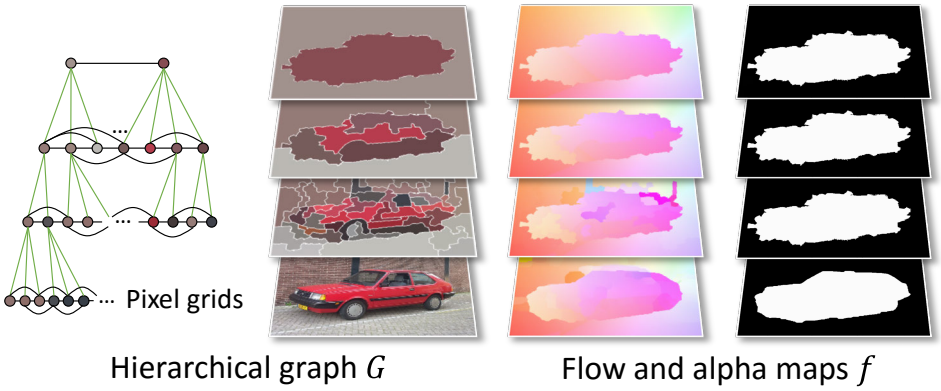


Figure 5.2 Hierarchical model. Each layer (2D MRF) estimates a dense flow and alpha map f, which is regularized by higher-level estimates and the final estimates are obtained at the bottom layer.

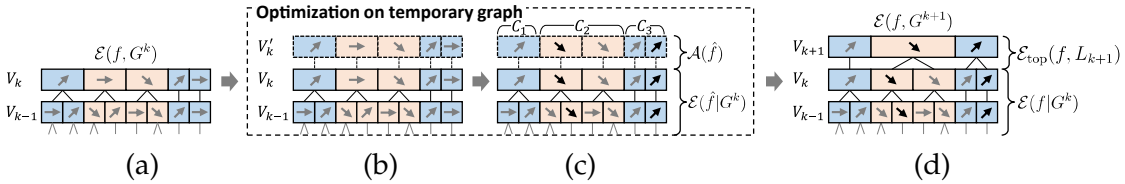


Figure 5.3 Bottom-up Graph Construction (one step). Each rectangular cell in the illustration represents a node $i \in V_l$ and a set of contiguous cells represents a graph layer V_l. The arrows and colors denote flow and alpha labels f_i (red: foreground, blue: background). (a) Graph G^k and its labeling f. (b) By duplicating the top layer V_k of G^k, we create a temporary graph $\hat{G}^{k+1}$ as an approximation of G^{k+1}. (c) We optimize the labeling $\hat{f}$ on $\hat{G}^{k+1}$. The number of unique labels in V'_k is reduced by *label costs* [Delong *et al.*, 2012] to induce region merging. (d) V'_k is converted into a new layer V_{k+1}, by merging nodes of V'_k assigned the same label that form connected components in $\hat{G}^{k+1}$.

5.4.1 Bottom-Up Hierarchy Construction

To formally describe our bottom-up procedure, we denote $G^k = (V^k, E^k)$ as a hierarchy consisting of $k+1$ layered subgraphs $\{L_0, \cdots, L_k\}$ where $L_l = (V_l, E_l)$. We also define it sequentially, *i.e.*, G^k and G^{k+1} share the same structure for the bottom $k+1$ layers.

At a high level, our bottom-up procedure is presented as a sequence of subtasks, where given a current solution $\{f, G^k\}$ we estimate $\{f, G^{k+1}\}$ as illustrated in Figures 5.3 (a) and (d), respectively. We estimate $\{f, G^{k+1}\}$ as approximate minimizers of $\mathcal{E}(f, G^{k+1})$ in Eq. (5.8). Here, $\mathcal{E}(f, G^{k+1})$ given G^k can be separated into two parts

$$\mathcal{E}(f, G^{k+1}) = \mathcal{E}(f|G^k) + \mathcal{E}_{\text{top}}(f, L_{k+1}), \tag{5.15}$$

where $\mathcal{E}(f|G^k)$ is energy involved in the *known* graph G^k while $\mathcal{E}_{\text{top}}(f, L_{k+1})$ refers to the *unknown* top layer L_{k+1}.

$$\mathcal{E}_{\text{top}}(f, L_{k+1}) = \mathcal{E}_{\text{mrf}}(f|L_{k+1}) + \mathcal{E}^{k+1}_{\text{reg}}(f|G^{k+1}) + \mathcal{E}^{k+1}_{\text{gra}}(V_{k+1}). \tag{5.16}$$

Jointly inferring G^{k+1} and its labeling f is difficult. Therefore, we assume a known *temporary graph* $\hat{G}^{k+1}$ for unknown G^{k+1}, and we replace this joint problem by a simpler labeling problem $\hat{f}$ on $\hat{G}^{k+1}$.

$$\hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1}) = \mathcal{E}(\hat{f}|G^k) + \mathcal{A}(\hat{f}). \tag{5.17}$$

Here, $\mathcal{E}(\hat{f}|G^k)$ is equivalent to $\mathcal{E}(f|G^k)$ in Eq. (5.15), and $\mathcal{A}$ is an approximation of the top layer energy $\mathcal{E}_{\text{top}}$.

In following three sections, we detail lines 4–10 of Algorithm 5.1 and explain how we derive $\hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1})$, optimize it, and obtain the desired solution $\{f, G^{k+1}\}$ from $\{\hat{f}, \hat{G}^{k+1}\}$.

Energy Approximation using Temporary Graphs

We now briefly explain the conversion from $\mathcal{E}(f, G^{k+1})$ in Eq. (5.15) to $\hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1})$ in Eq. (5.17). For detailed derivations, please refer to Appendix C.2.

To relax the joint inference of $\mathcal{E}(f, G^{k+1})$, we create a temporary graph $\hat{G}^{k+1}$ as an approximation of G^{k+1}, by duplicating the top layer of G^k as $L'_k = (V'_k, E'_k) \leftarrow (V_k, E_k)$ (line 4 of Algorithm 5.1). We illustrate G^k and $\hat{G}^{k+1}$ in Figures 5.3 (a) and (b), respectively. Here, a labeling $\hat{f}$ on $\hat{G}^{k+1}$ (or V'_k) can equivalently express all possible f on G^{k+1} (or V_{k+1}), because V'_k is the finest form of any possible V_{k+1}. The labeling $\hat{f}$ is copied from f at line 5.

Substituting $G^{k+1} \leftarrow \hat{G}^{k+1}$ into $\mathcal{E}(f, G^{k+1})$, we derive its approximation $\hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1})$

in Eq. (5.17) with following $\mathcal{A}$.

$$\mathcal{A}(\hat{f}) = \mathcal{E}_{\text{mrf}}(\hat{f}|L'_k) + \mathcal{E}^{k+1}_{\text{reg}}(\hat{f}|\hat{G}^{k+1}) + \mathcal{E}^{k+1}_{\text{gra}}(\hat{f}|V'_k). \tag{5.18}$$

The conversion from $\mathcal{E}_{\text{top}}$ in Eq. (5.16) to this $\mathcal{A}$ is provably exact except for only terms $\mathcal{E}^{k+1}_{\text{gra}}$ and $\mathcal{E}^{st}_{\text{reg}}$ in $\mathcal{E}_{\text{mrf}}$ of Eq. (5.2). Here, conversion of $\mathcal{E}^{k+1}_{\text{gra}}$ is tricky because we need to convert variables from V_{k+1} to a labeling $\hat{f}$ on V'_k. We observe that the region of each node $i \in V_{k+1}$ should optimally 1) be a connected component, 2) assigned a single label unique from neighbors, and 3) be the union of regions in V'_k. Thus, we can treat nodes $i \in V_{k+1}$ as connected components C_i of nodes in V'_k assigned the same label, *i.e.*,

$$V_{k+1} \equiv \left\{ C_i \middle| \begin{array}{l} \text{nodes } {}^{\forall}c \in C_i \text{ in } V'_k \text{ are assigned} \\ \text{the same label } \hat{f}_c \text{ and connected.} \end{array} \right\}. \tag{5.19}$$

This property allows us to rewrite $\mathcal{E}^{k+1}_{\text{gra}}(V_{k+1})$ in Eq. (5.11) as a function of $\hat{f}$. To further make inference tractable, we relax the connectivity of $|V_{k+1}|$ and treat $|V_{k+1}|$ as *label costs* [Delong *et al.*, 2012] of $\hat{f}$, *i.e.*, the number of unique labels $\hat{f}_i$ in V'_k without considering their spatial connections. In this manner, the formulation of Eq. (5.11) becomes the same as that of multi-region segmentation [Delong *et al.*, 2012]. Following their model fitting approach based on alpha expansion moves [Boykov *et al.*, 2001], we treat the label costs and the likelihood terms of Eq. (5.11) as pairwise submodular terms and unary terms, respectively. See more discussions in Appendix C.1.

Optimization of Approximation Energy

In Figure 5.3 (c) and at line 6 of Algorithm 5.1, we minimize the approximation energy $\hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1})$ of Eq. (5.17) with known $\hat{G}^{k+1}$. To efficiently infer the continuous 5dof labels in $\hat{f}$, we use the local expansion move method of [Taniai *et al.*, 2016a, 2014] (Chapter 4).

In its general form, the local expansion move algorithm repeatedly solves the following binary labeling problem for each target node $i \in V$ visited in sequence.

$$f^{(t+1)} = \operatorname{argmin} \mathcal{E}(f|f_j \in \{f^{(t)}_j, \boldsymbol{\ell}\} \text{ for } j \in R_i). \tag{5.20}$$

Here, $R_i \subset V$ is a set of local nodes around the target node i (named *expansion region*), and this operation tries to improve the labels of the local nodes $j \in R_i$ by assigning them either their current label $f^{(t)}_j$ or a candidate label $\boldsymbol{\ell}$. We use graph cuts [Boykov and Kolmogorov, 2004] to solve this binary problem.

Our version of local expansion moves is summarized in Algorithm 5.2. During the bottom-up process, we randomly visit all nodes in the top layer V'_k (*i.e.*, target layer V_T) at line 1, and update the labeling of local nodes. In order to apply the local expansion move algorithm for our hierarchical MRF model, we extended it in two ways. First,

Algorithm 5.1: TWO-PASS OPTIMIZATION PROCESS

input : Two images $\mathbf{I}, \mathbf{I}'$
output: Hierarchical graph G and flow-alpha map f

Initialize the graph: $G \leftarrow G^1$
Initialize the labeling f and color models $\boldsymbol{\theta}^F, \boldsymbol{\theta}^B$ (Section 5.5.3)
for $k = 1, 2, \cdots$ **do bottom-up graph construction**
  Create temporary $\hat{G}^{k+1}$ by duplicating V_k of G^k ;
  Initialize temporary $\hat{f}$ by copying labels from f ;
  Perform local expansion moves ($\hat{\mathcal{E}}$,$\hat{f}$, $\hat{G}^{k+1}$, V'_k)
   $\hat{f} \leftarrow \mathrm{argmin}\, \hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1})$;
  Create G^{k+1} by merging nodes of V'_k in $\hat{G}^{k+1}$;
  if *rejection criterion is met* **then** break;
  Update solution $\{f, G\} \leftarrow \{\hat{f}, G^{k+1}\}$;
  if *any stopping criteria is met* **then** break;
end
for $k = H, \cdots, 1$ **do top-down label refinement**
  Perform local expansion moves ($\mathcal{E}$, f, G, V_k)
   $f \leftarrow \mathrm{argmin}\, \mathcal{E}(f|G)$ with f_i fixed for $\forall i \in V_{l>k}$;
end

Algorithm 5.2: LOCAL EXPANSION MOVES [Taniai *et al.*, 2016a, 2014]

argments: (model $\mathcal{E}$, labeling f, graph G, target layer V_T)

foreach *target node* $i \in V_T$ **do**
  Make neighborhood: $N_i \leftarrow \{i\text{'s neighbors}\} \cup \{i\}$;
  Make expansion region: $R_i \leftarrow \{N_i\text{'s descendants}\} \cup N_i$;
  foreach *candidate proposer* **do**
    Generate a candidate label $\boldsymbol{\ell} = (\mathbf{T}, \alpha)$;
    Apply a local expansion move using min-cut:
     $f \leftarrow \mathrm{argmin}\, \mathcal{E}(f|G)$ with $f_j \in \{f_j, \boldsymbol{\ell}\}$ for $j \in R_i$;
  end
end
return f ;

the expansion region R_i is extended from i's neighbors (N_i at line 2) to include all their descendants (line 3). Second, when generating candidate labels $\boldsymbol{\ell}$ for the target node i (line 5), we use four types of candidate proposers listed below.

- *Expansion proposer* generates a label by copying the current label as $\boldsymbol{\ell} \leftarrow f_i$. This tries to propagate the current label f_i to nearby nodes in R_i as explained in [Taniai *et al.*, 2016a].
- *Cross-view proposer* refers to the current labeling f' of the other image, and uses a label $f'_{i'}$ that gives warping to the target node region Ω_i as a candidate $\boldsymbol{\ell}$, using inverse warp of $f'_{i'}$. This is similar to *view propagation* in [Bleyer *et al.*, 2011; Besse *et al.*, 2014].
- *Merging proposer* generates labels $\boldsymbol{\ell} \leftarrow w_i f_i + w_j f_j$ as weighted sums of i's current label f_i and its neighbors' labels f_j, $j \in N_i$. The weights $w_i, w_j \in [0, 1]$ are proportional to their region sizes $|\Omega_i|, |\Omega_j|$. This is a new extension for promoting better region merging.
- *Perturbation proposer* generates labels $\boldsymbol{\ell} \leftarrow f_i + \Delta$ by randomly perturbing the current label f_i. Similarly to [Taniai *et al.*, 2016a; Bleyer *et al.*, 2011], we iterate between lines 5 and 6 several times while reducing the perturbation size $|\Delta|$ by half.

Incremental Layer Construction

In Figure 5.3 (d) and at line 7 of Algorithm 5.1, we create a new graph G^{k+1} by merging nodes of V'_k in $\hat{G}^{k+1}$. Here, V_{k+1} is created from V'_k and $\hat{f}$ using the variable conversion of Eq. (5.19). After merging regions, we check the number of new foreground regions at the top layer. If it is zero (line 8), we reject the new solution and stop the graph construction process. Otherwise, we adopt the new solution $\{\hat{f}, G^{k+1}\}$ as $\{f, G\}$ (line 9). Later we check the foreground count again and if it is one or not reduced from the previous iteration (line 10), we stop the graph construction process.

5.4.2 Top-Down Labeling Refinement

After the bottom-up phase, we further refine the labeling f during the top-down phase shown at lines 12–14 of Algorithm 5.1. Since G is held fixed during this step, we can directly optimize $\mathcal{E}(f|G)$ using local expansion moves without requiring the energy conversion described in Section 5.4.1. During this phase, we visit layers V_k in G in top-down order (from $k = H$ to $k = 1$) and apply local expansion moves with V_k as a target layer V_T. Here, the labeling f for the higher layers V_l ($l > k$) does not change, because the expansion regions R_i only contain nodes in layers V_k and below.

5.5 Implementation Details

We now discuss initialization steps and features used in our method.

5.5.1 Local HOG Features

The images are first resized so that their larger dimension becomes 512 pixels. A Gaussian pyramid is then built for each image (we use 1 octave and 1 sub-octave). From each pyramid layer, we densely extract local histogram of gradient (HOG) feature descriptors [Dalal and Triggs, 2005]. These features are extracted at every pixel on the image grid from patches of size 27×27 pixels. Our HOG descriptors are 96-dimensional. We use a 3×3 cell grid for each patch and 16 equally spaced bins for the oriented gradient histograms. Each gradient histogram thus has 16 bins for signed gradients and 8 bins for unsigned gradients. The histograms for each contiguous 2×2 block of the 3×3 cell grid are aggregated to form a 24-dimensional vector. These are then L2-normalized followed by element-wise truncation (using a threshold of 0.5). Four such vectors are concatenated to form the final 96-dimensional HOG descriptor. These HOG features are used to compute the flow data terms $\mathcal{E}^i_{\text{flo}}$ described earlier. They are also used to construct bag of visual words (BoW) histogram features required during the initialization stage.

5.5.2 BoW Histogram Features

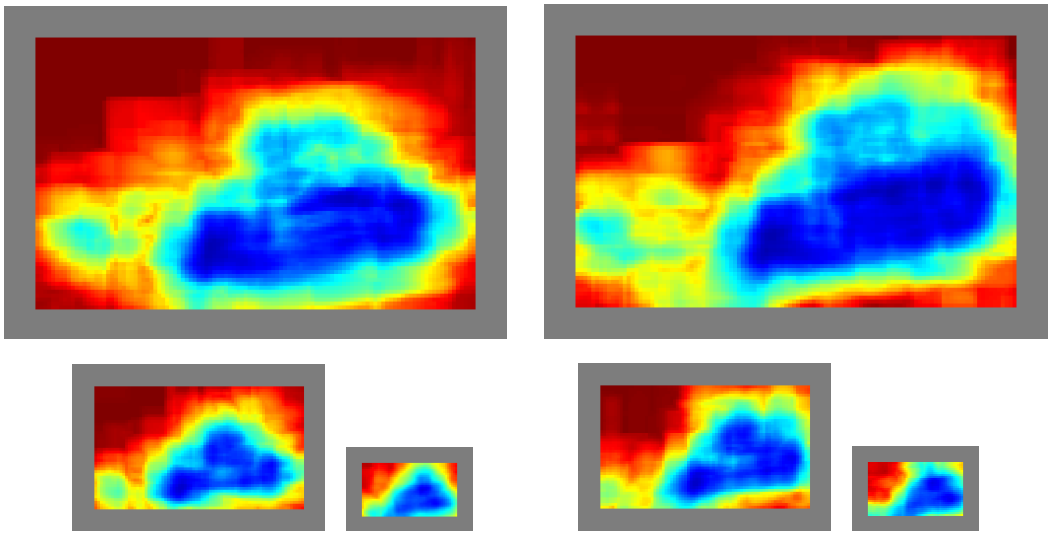
Each HOG descriptor is vector-quantized using a K-means codebook of size 256. Next, BoW histograms are computed from several overlapping image patches of size 64×64 pixels. These patches are sampled every 4 pixels (both horizontally and vertically) in the image. We use integral images (one per visual word) to speed up the BoW histogram computation. All the visual words are aggregated into a histogram. This is repeated for 2×2 sub-regions. The five BoW histograms are then L2-normalized followed by element-wise square root normalization[3]. The 256-dimensional histograms are concatenated to form 1280-dimensional BoW histogram features.

5.5.3 Initialization

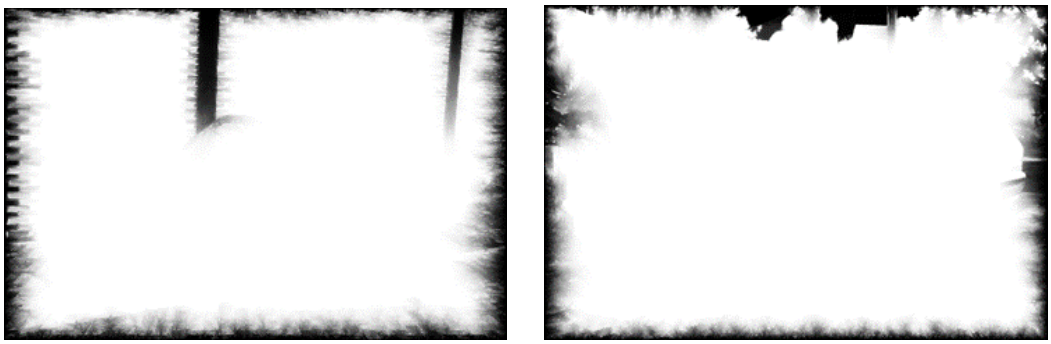
During initialization, initial flow candidates and foreground/background color models for each image are computed as follows. First, dense matching is done using the BoW features at three levels in the image pyramid. The Euclidean distances between each pixel feature in the first image and features for all pixels within a search window in the second image are computed. Fortunately this is quite fast due to the sparsity of the

[3]This is equivalent to using a Hellinger kernel instead of the Euclidean distance to measure the similarity of two feature vectors.

Input two images



Segmentation heat maps on three levels of pyramids.



Geodesic distance maps.

Figure 5.4 Initial segmentation cues. For an input image pair (top), minimum/maximum ratios of BoW feature matching distances in local windows (middle) are computed as cues of initial segmentation. Low ratios (blue) are likely to suggest foreground. Geodesic distances from the image boundary (bottom) are also used to add background clues (black regions).

BoW features. The best match is stored as a flow candidate. The ratio of the Euclidean distances of the best and worst match is computed. We use this heuristic to predict the probability of a true match, motivated by the ratio test [Lowe, 2004] (see Figure 5.4).

Areas with high and low match probabilities are likely to be the "foreground" and "background" respectively. By thresholding the ratio values, we create foreground/background soft seeds and initial segmentations as input to GrabCut [Rother *et al.*, 2004] and learn color models $\{\boldsymbol{\theta}^F, \boldsymbol{\theta}^B\}$ for each image. Geodesic distance from the image boundary is used as an additional unary background likelihood term (Figure 5.4, right). See Appendix C.3 for further implementation details.

5.5.4 Efficient Implementation

Three key ideas allow our optimization method to be efficiently implemented. First, unary terms $\mathcal{E}^i_{\text{flo}}(f_i)$ and $\mathcal{E}^i_{\text{seg}}(f_i)$ in Eq. (5.2) can be efficiently computed using the tree structure of G. Specifically, the unary cost $\mathcal{E}^p(\boldsymbol{\ell})$ of a node $p \in V_l$ is computed as the sum $\sum_c \mathcal{E}^c(\boldsymbol{\ell})$ over its children $c \in V_{l-1}$, if their labels $\boldsymbol{\ell}$ are the same. This constant-label property is satisfied during local expansion moves because the candidate label $\boldsymbol{\ell}$ is the same for all nodes in an expansion region R_i. Thus, at line 6 of Algorithm 5.2, we compute the unary costs $\mathcal{E}^j(\boldsymbol{\ell})$ for $j \in R_i$ by sequentially summing them up from bottom to top layer nodes. Second, we exclude the pixel layer L_0 / V_0 from the graph G during the main iterations. We add it to G just before the last refinement step in the top-down phase ($k = 1$ at line 13 of Algorithm 5.1). Finally, we use efficient graph cuts [Boykov and Kolmogorov, 2004] at line 6 of Algorithm 5.2, instead of QPBO [Kolmogorov and Rother, 2007]. This is possible because our energy is submodular under (local) expansion moves [Taniai *et al.*, 2016a, 2014]. The proofs are in Appendix C.4.

5.6 Experiments

We evaluate our method for flow and segmentation accuracy and compare it to existing methods on our new dataset.

Dataset

Our dataset comprises of 400 image pairs divided into three groups – **FG3DCar** contains 195 image pairs of vehicles from [Lin *et al.*, 2014]. **JODS** contains 81 image pairs of airplanes, horses, and cars from [Rubinstein *et al.*, 2013]. **PASCAL** contains 124 image pairs of bicycles, motorbikes, buses, cars, trains from [Hariharan *et al.*, 2011]. The object categories of JODS and PASCAL are summarized in Figure 5.5. We also show some example image pairs from each group in Figure 5.6. Notice that JODS and PASCAL contain some horizontally flipped image pairs, *i.e.*, one image requires a mirror reflection

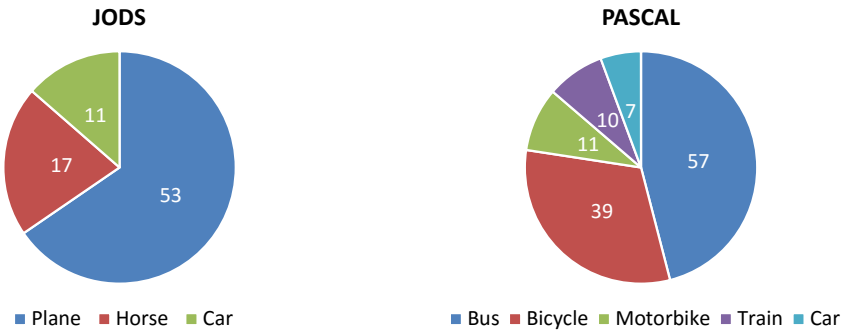


Figure 5.5 Object categories of JODS and PASCAL.

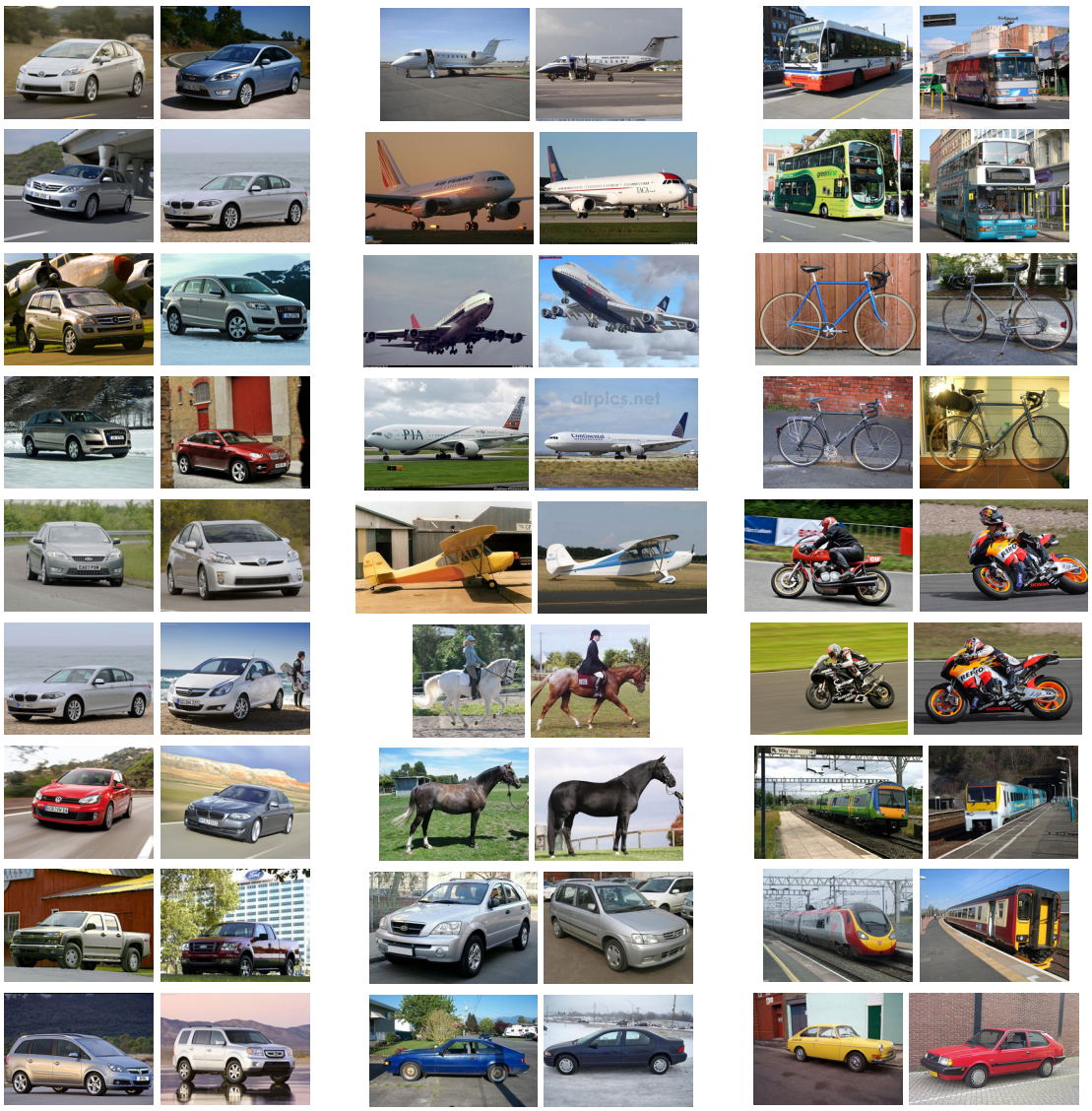


Figure 5.6 Example image pairs of our dataset. **FG3DCar** contains 195 image pairs of vehicles from [Lin *et al.*, 2014]. **JODS** contains 81 image pairs from [Rubinstein *et al.*, 2013]. **PASCAL** contains 124 image pairs from [Hariharan *et al.*, 2011].

prior to alignment. The numbers of such flipped image pairs included in each group are follows. FG3DCar: 2 pairs (1 %). JODS: 9 pairs (11 %). PASCAL: 48 pairs (39 %).

Flow Accuracy

We evaluate flow accuracy by the percentage of pixels in the true foreground region that have an error measure below a certain threshold. Here, we compute the absolute flow endpoint error (*i.e.*, the Euclidean distance between estimated and true flow vectors) in a normalized scale where the larger dimensions of images are 100 pixels.

Segmentation Accuracy

We use the standard *intersection-over-union* ratio metric for segmentation accuracy. As existing flow estimation methods do not recover common foreground regions, we compute them by post-processing the estimated flow maps. Specifically, given the two flow maps, we do a left-right consistency check with a suitable threshold and treat pixels that pass this test as foreground.

Settings

We strictly fixed all the parameters throughout the experiments as follows. For the data and graph term parameters, we set $\{\lambda_{\text{flo}}, \lambda_{\text{occ}}, \tau_{\text{D}}, \lambda_{\text{seg}}, \lambda_{\text{nod}}, \lambda_{\text{col}}\} \leftarrow \{0.25, 2.4, 6.5, 0.8, 125, 1\}$. For regularization parameters $\{\lambda_{\text{st1}}, \lambda_{\text{st2}}, \tau_{\text{st}}, \lambda_{\text{pc1}}, \lambda_{\text{pc2}}, \tau_{\text{pc}}\}$ associated with the pixel layer (edges E_0 and E_1^{pc}) we use $\{0.5, 20, 20, 0.005, 10, 200\}$, and for the other edges we use $\{0.1, 4, 20, 0.04, 8, 200\}$. See Appendix C.5 for our strategy of tuning parameters. Our method is implemented using C++ and run by a single thread on a Core i7 CPU of 3.5 GHz.

5.6.1 Comparison with Existing Approaches

For correspondence, we compare our method with SIFT Flow [Liu *et al.*, 2011], DSP [Kim *et al.*, 2013] and DFF [Yang *et al.*, 2014]. We also evaluate our method using only the single layer model without hierarchy, which can be done by skipping the bottom-up construction step in Algorithm 5.1. This single layer method can be seen as a variant of [Taniai *et al.*, 2016a]. Here, we omit results of HaCohen *et al.* [2011] for its low performance on our dataset. It could not find any correspondence for many image pairs.

For cosegmentation, we compare our method with Joulin *et al.* [2010] and Faktor and Irani [2013] based only on segmentation accuracies. For Joulin *et al.* [2010] that cannot identify the "foreground" label from $\{0, 1\}$, we refer to ground truth and choose for each image pair either 0 or 1 to maximize the scores. Results of their extension method [Joulin *et al.*, 2012] are omitted since we could not observe improvemnts over [Joulin *et al.*, 2010]

in our settings. Also, we omit results of Dai *et al.* [2013] as it did not work for many image pairs. The method seems to fail in finding matches with learned templates.

We summarize average accuracy scores for each subset in the upper part of Table 5.1, where flow accuracy is evaluated using a threshold of 5 pixels. The plots in Figure 5.7 show average flow accuracies with varying thresholds. As shown here, our method achieves the best performance on all three groups at all thresholds. Our average flow accuracies for FG3DCar, JODS and PASCAL, respectively, are up to 45%, 19% and 34% higher than SIFT Flow (best existing method). Superior results to our single layer method shows the effectiveness of our hierarchical model and inference. DFF [Yang *et al.*, 2014] cannot handle large appearance differences of objects due to lack of explicit regularization. We show qualitative comparisons with SIFT Flow [Liu *et al.*, 2011] and DSP [Kim *et al.*, 2013] in Figures 5.9–5.11.

We report average segmentation scores in the lower part of Table 5.1. Figures 5.12–5.14 show qualitative comparisons with Faktor and Irani [2013] and Joulin *et al*. [Joulin *et al.*, 2010]. Although our model for segmentation is quite simple compared to other methods, our method is competitive or has higher accuracy due to joint inference of foreground correspondence.

Since our method and others do not explicitly handle flipped image pairs that are included in our dataset, they fail to find correspondence for them. Therefore, we also evaluate accuracy scores similar to Table 5.1 and Figure 5.7 but excluding flipped image pairs from the evaluation. We show the average scores for the three groups in Table 5.2, and the plots of average flow accuracies with varying thresholds in Figure 5.8. We observe similar trends between scores with and without flipped image pairs.

Running time of our method is about 7 minutes for obtaining a pair of flow-alpha maps of 512×384 pixels, including 1 minute for the feature extraction and initialization, 3 minutes for the final refinement step with the pixel layer.

5.7 Summary

We have presented a joint method for cosegmentation and dense correspondence estimation in two images. Our method uses a hierarchical MRF model and jointly infers the hierarchy as well as segmentation and correspondence using iterated graph cuts. Our method outperforms a number of methods designed specifically either for correspondence recovery [Liu *et al.*, 2011; Kim *et al.*, 2013; HaCohen *et al.*, 2011; Yang *et al.*, 2014] or cosegmentation [Joulin *et al.*, 2010, 2012; Faktor and Irani, 2013; Dai *et al.*, 2013]. We provide a new dataset for quantitative evaluation. Enforcing left-right consistencies on flow and segmentation maps for two images, or by using multiple images [Cho *et al.*, 2015; Zhou *et al.*, 2015; Joulin *et al.*, 2012] are promising avenues for future work.

Table 5.1 Correspondence and cosegmentation benchmark results. FAcc is flow accuracy rate for an error threshold of 5 pixels in a normalized scale where the larger dimensions of two images are 100 pixels. SAcc is segmentation accuracy by intersection-over-union ratios. SAcc scores $(\cdots)$ of optical flow mothods are computed by post-processing using left right consistency check. Note that our single layer method refers to the proposed algorithm without the hierarchy construction, whose optimization strategy is similar to [Taniai *et al.*, 2016a].

Optical flow / cosegmentation methods	FG3DCar		JODS		PASCAL	
	FAcc	SAcc	FAcc	SAcc	FAcc	SAcc
Ours	**0.830**	**0.744**	**0.595**	0.495	**0.483**	**0.624**
Our single layer ([Taniai *et al.*, 2016a])	0.728	**0.746**	0.473	0.500	0.414	0.616
SIFT Flow [Liu *et al.*, 2011]	0.634	(0.405)	0.522	(0.242)	0.453	(0.392)
DSP [Kim *et al.*, 2013]	0.487	(0.279)	0.465	(0.224)	0.382	(0.329)
DFF [Yang *et al.*, 2014]	0.495	(0.312)	0.304	(0.210)	0.224	(0.195)
Faktor and Irani [2013]	–	0.678	–	**0.539**	–	0.492
Joulin *et al.* [2010]	–	0.450	–	0.318	–	0.389

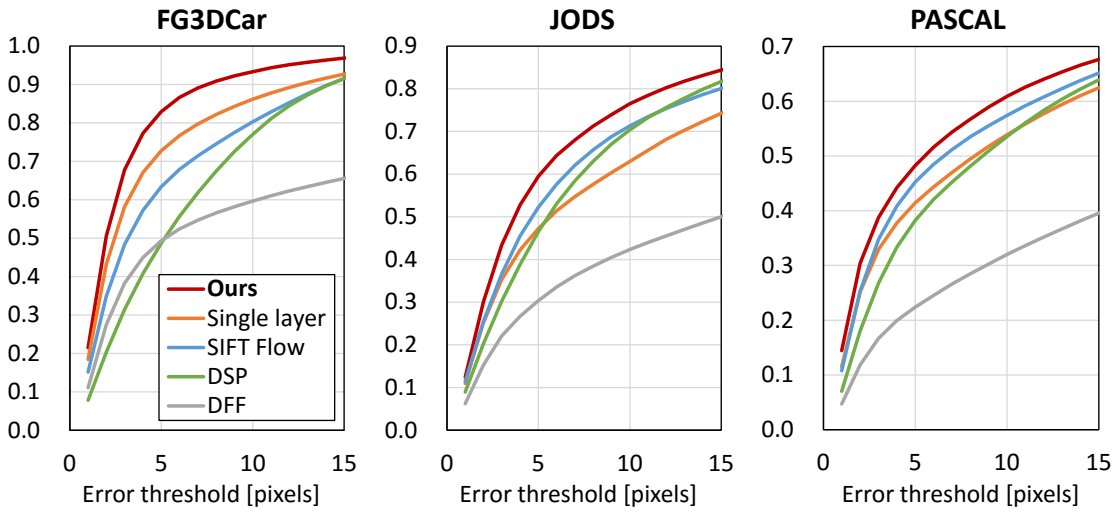


Figure 5.7 Average flow accuracies with varying error thresholds. Here, the endpoint errors are evaluated in a normalized scale in a normalized scale where the larger dimensions of two images are 100 pixels. Our method always shows best scores. DFF [Yang *et al.*, 2014] is not robust due to lack of explicit regularization. By comparing with our single layer method, the proposed hierarchical model and inference have shown to be effective.

Table 5.2 Correspondence and cosegmentation benchmark results (**without flipped images**). FAcc is flow accuracy rate for an error threshold of 5 pixels in a normalized scale where the larger dimensions of two images are 100 pixels. SAcc is segmentation accuracy by intersection-over-union ratios. SAcc scores $(\cdots)$ of optical flow mothods are computed by post-processing using left right consistency check. Note that our single layer method refers to the proposed algorithm without the hierarchy construction, whose optimization strategy is similar to [Taniai *et al.*, 2016a].

Optical flow / cosegmentation methods	FG3DCar		JODS		PASCAL	
	FAcc	SAcc	FAcc	SAcc	FAcc	SAcc
Ours	**0.838**	**0.744**	**0.665**	0.512	**0.755**	**0.634**
Our single layer ([Taniai *et al.*, 2016a])	0.735	**0.745**	0.525	0.516	0.649	0.625
SIFT Flow [Liu *et al.*, 2011]	0.640	(0.405)	0.582	(0.258)	0.704	(0.458)
DSP [Kim *et al.*, 2013]	0.492	(0.279)	0.517	(0.232)	0.598	(0.361)
DFF [Yang *et al.*, 2014]	0.499	(0.314)	0.331	(0.217)	0.338	(0.237)
Faktor and Irani [2013]	–	0.676	–	**0.545**	–	0.478
Joulin *et al.* [2010]	–	0.450	–	0.331	–	0.400

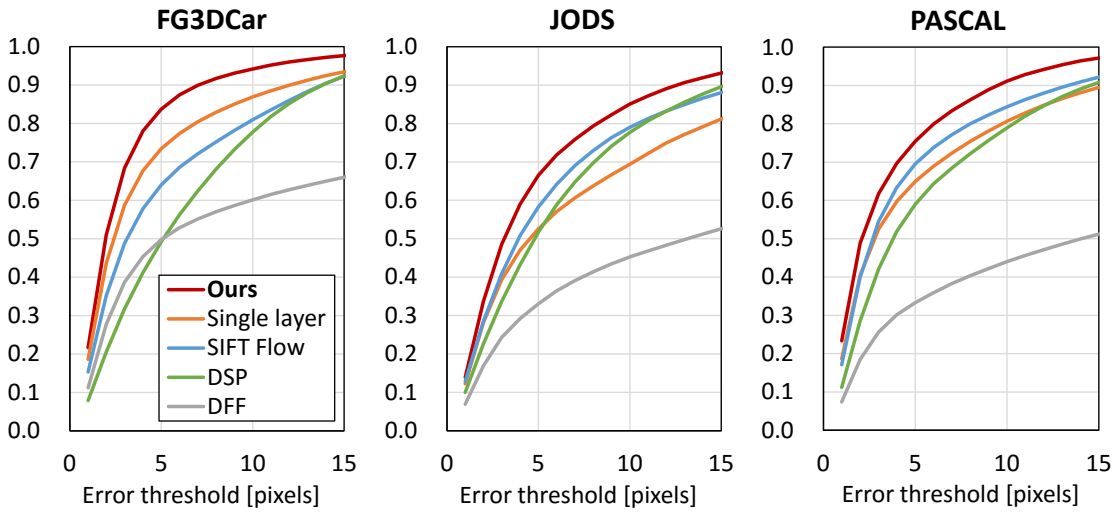
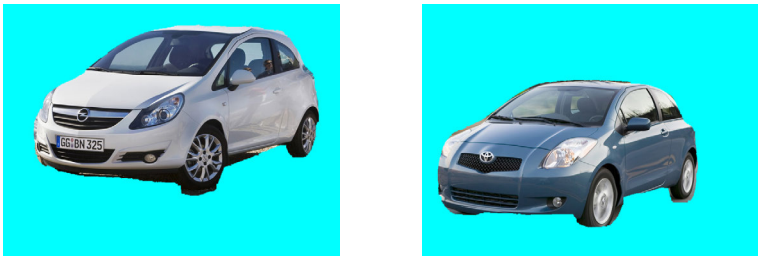


Figure 5.8 Average flow accuracies with varying error thresholds (**without flipped images**). Similarly to Figure 5.7, our method shows always best scores.

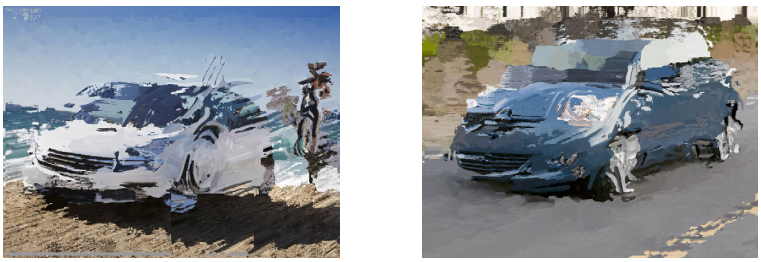
Input two images



Ground truth warped images

Estimated aligned images with segmentation by our method

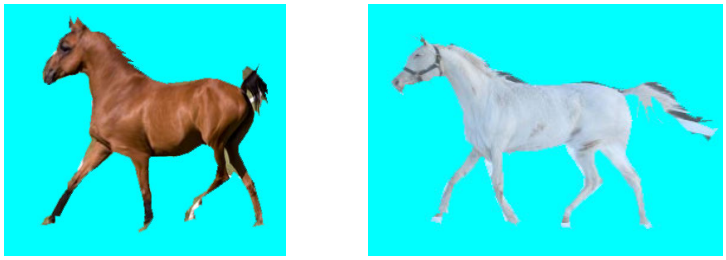
Estimated aligned images by SIFTFlow [Liu *et al.*, 2011]



Estimated aligned images by DSP [Kim *et al.*, 2013]

Figure 5.9 Correspondence results (FG3Dcar).

Input two images



Ground truth warped images

Estimated aligned images with segmentation by our method

Estimated aligned images by SIFTFlow [Liu *et al.*, 2011]

Estimated aligned images by DSP [Kim *et al.*, 2013]

Figure 5.10 Correspondence results (JODS).

Input two images

Ground truth warped images

Estimated aligned images with segmentation by our method

Estimated aligned images by SIFTFlow [Liu *et al.*, 2011]

Estimated aligned images by DSP [Kim *et al.*, 2013]

Figure 5.11 Correspondence results (PASCAL).

Input two images

Ground truth cosegmentation

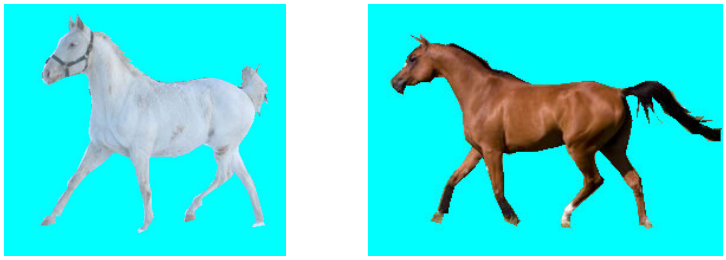
Estimated cosegmentation by our method

Estimated cosegmentation by Faktor and Irani [2013]

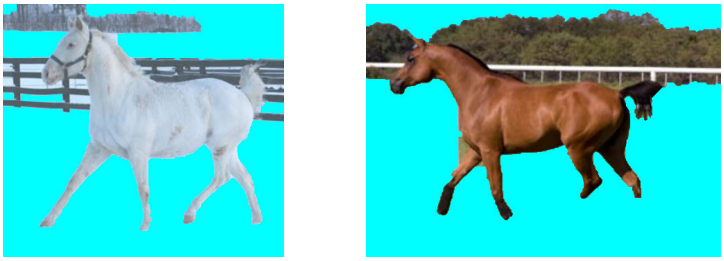
Estimated cosegmentation by Joulin *et al.* [2010]

Figure 5.12 Cosegmentation results (FG3Dcar).

Input two images



Ground truth cosegmentation



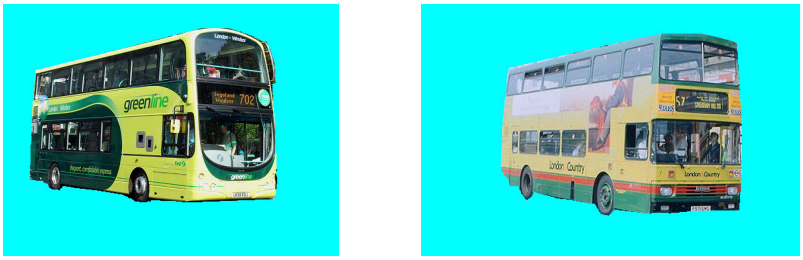
Estimated cosegmentation by our method

Estimated cosegmentation by Faktor and Irani [2013]

Estimated cosegmentation by Joulin *et al.* [2010]

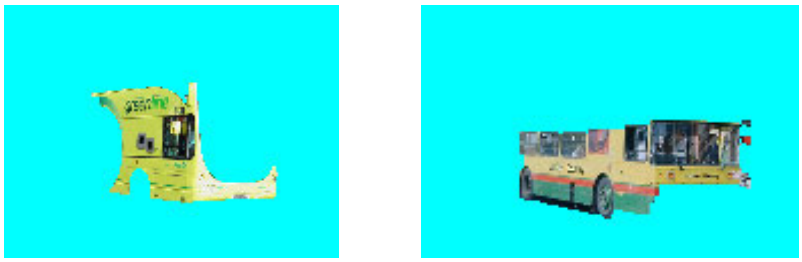
Figure 5.13 Cosegmentation results (JODS).

Input two images

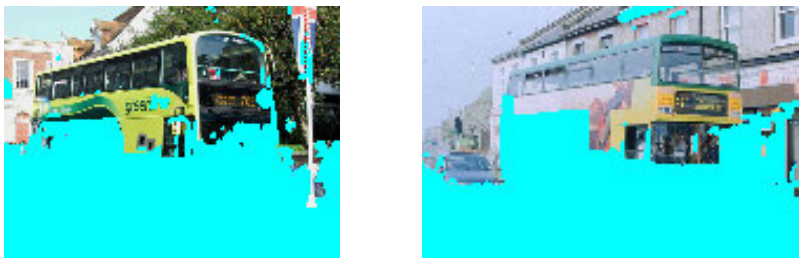


Ground truth cosegmentation

Estimated cosegmentation by our method



Estimated cosegmentation by Faktor and Irani [2013]



Estimated cosegmentation by Joulin *et al.* [2010]

Figure 5.14 Cosegmentation results (PASCAL).

6

Joint MRF Inference for Stereo Scene Flow and Motion Segmentation

STEREO scene flow is a task of simultaneously recovering disparity and optical flow maps given a sequence of stereo image pairs. While stereo disparities can be efficiently estimated even in real-time, optical flow estimation requires much more expensive computations due to the large 2D label space, making it a bottleneck in scene flow estimation. However, if the scene is rigid, explicit flow estimation can be avoided, because it can be efficiently recovered by computing projections of 3D points given by the disparity map. This only requires additional information of the 6-DOF camera ego-motion. In this chapter, we study fast inference of scene flow by fully exploiting this property. In a general dynamic scene, the presence of moving objects prevents direct application of this approach. Hence, we also estimate motion segmentation to identify moving object regions, for which optical flow is explicitly computed. This leads to a new unified framework of stereo, visual odometry, optical flow and motion segmentation.

6.1 Introduction

Scene flow refers to 3D flow or equivalently the dense 3D motion field of a scene [Vedula *et al.*, 1999]. It can be estimated from video acquired with synchronized cameras from multiple viewpoints [Lv *et al.*, 2016; Mayer *et al.*, 2016; Menze and Geiger, 2015; Vogel *et al.*, 2015] or with RGB-D sensors [Hornacek *et al.*, 2014; Jaimez *et al.*, 2015b; Herbst *et al.*, 2013; Quiroga *et al.*, 2014] and has applications in video analysis and editing, 3D mapping, autonomous driving [Menze and Geiger, 2015] and mobile robotics.

Scene flow estimation builds upon two tasks central to computer vision – stereo matching and optical flow estimation. Even though many existing methods can already solve these two tasks independently [Kolmogorov and Zabih, 2001; Hirschmuller, 2008; Taniai *et al.*, 2014; Lucas and Kanade, 1981; Horn and Schunck, 1981; Xu *et al.*, 2012; Chen and Koltun, 2016], a naive combination of stereo and optical flow methods for

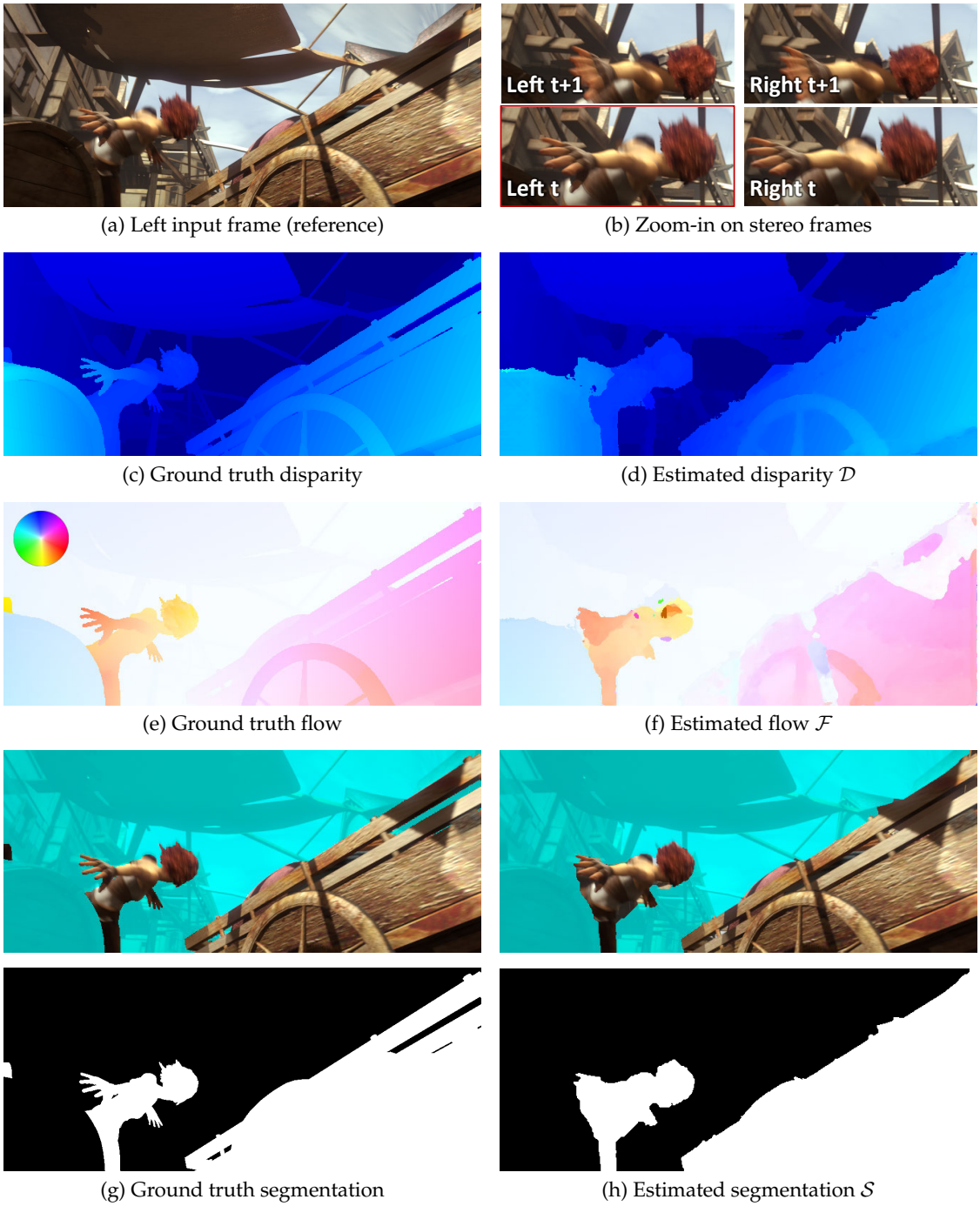
computing scene flow is unable to exploit inherent redundancies in the two tasks or leverage additional scene information which may be available. Specifically, it is well known that the optical flow between consecutive image pairs for stationary (rigid) 3D points are constrained by their depths and the associated 6-DOF motion of the camera rig. However, this idea has not been fully exploited by existing scene flow methods. Perhaps, this is due to the additional complexity involved in simultaneously estimating camera motion and detecting moving objects in the scene.

Recent renewed interest in stereoscopic scene flow estimation has led to improved accuracy on challenging benchmarks, which stems from better representations, priors, optimization objectives as well as the use of better optimization methods [Huguet and Devernay, 2007; Wedel *et al.*, 2011; Čech *et al.*, 2011; Menze and Geiger, 2015; Vogel *et al.*, 2015; Lv *et al.*, 2016]. However, those state of the art methods are computationally expensive which limits their practical usage. In addition, other than a few exceptions [Vogel *et al.*, 2014], most existing scene flow methods process every two consecutive frames independently and cannot efficiently propagate information across long sequences.

In this study, we propose a new technique to estimate scene flow from a multi-frame sequence acquired by a calibrated stereo camera on a moving rig. We simultaneously compute dense disparity and optical flow maps on every frame. In addition, the 6-DOF relative camera pose between consecutive frames is estimated along with a per-pixel binary mask that indicates which pixels correspond to either rigid or non-rigid independently moving objects (see Figure 6.1). Our sequential algorithm uses information only from the past and present, thus useful for real-time systems.

We exploit the fact that even in dynamic scenes, many observed pixels often correspond to static rigid surfaces. Given disparity maps estimated from stereo images, we robustly compute the 6-DOF camera motion using visual odometry robust to outliers (moving objects in the scene). Given the ego-motion estimate, we improve the depth estimates at occluded pixels via epipolar stereo matching. Then, we identify image regions inconsistent with the camera motion and compute an explicit optical flow proposal for these regions. Finally, this flow proposal is fused with the camera motion-based flow proposal using fusion moves to obtain the final flow map and motion segmentation.

While these four tasks – stereo, optical flow, visual odometry and motion segmentation has been extensively studied, most of the existing methods solve these tasks independently. In contrast, we present a single unified framework where the solution to one task benefits the other tasks. This is the primary contribution of our work. Fortunately, this also increases the computational efficiency. Our method is significantly faster than top six methods on KITTI. Our method takes about 2–3 seconds per frame (on the CPU) whereas state-of-the-art methods take 1–50 minutes per-frame [Vogel *et al.*, 2015; Menze and Geiger, 2015; Lv *et al.*, 2016; Vogel *et al.*, 2013]. Not only is our method faster but it also explicitly recovers the camera motion and motion segmentation. We



(a) Left input frame (reference)

(b) Zoom-in on stereo frames

(c) Ground truth disparity

(d) Estimated disparity $\mathcal{D}$

(e) Ground truth flow

(f) Estimated flow $\mathcal{F}$

(g) Ground truth segmentation

(h) Estimated segmentation $\mathcal{S}$

Figure 6.1 Proposed unified framework. Our method estimates dense disparity and optical flow from stereo pairs, which is equivalent to stereoscopic scene flow estimation. The camera motion is simultaneously recovered and allows moving objects to be explicitly segmented in our approach.

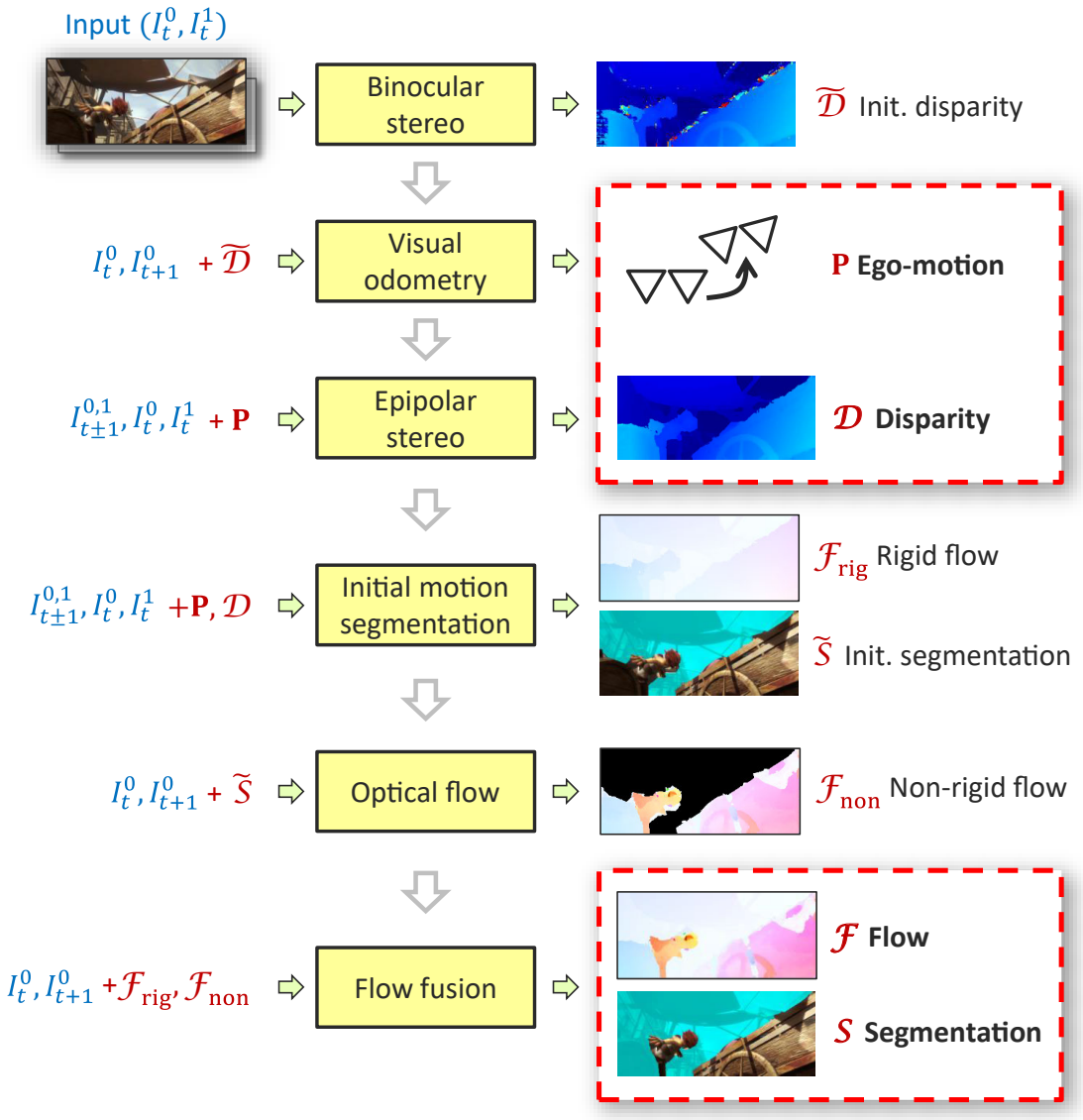


Figure 6.2 Overview of the proposed method. In the first three steps, we estimate the disparity $\mathcal{D}$ and camera motion $\mathbf{P}$ using stereo matching and visual odometry techniques. We then detect moving object regions by using the rigid flow $\mathcal{F}_{\text{rig}}$ computed from $\mathcal{D}$ and $\mathbf{P}$. Optical flow is performed only for the detected regions, and the resulting non-rigid flow $\mathcal{F}_{\text{non}}$ is fused with $\mathcal{F}_{\text{rig}}$ to obtain final flow $\mathcal{F}$ and segmentation $\mathcal{S}$.

now discuss how our unified framework benefits each of the four individual tasks.

Optical Flow. Given known depth and camera motion, the 2D flow for rigid 3D points which we refer to as *rigid flow* in this study, can be recovered more efficiently and accurately compared to generic *non-rigid flow*. We still need to compute non-rigid flow but only at pixels associated with moving objects. This reduces redundant computation. Furthermore, this representation is effective for occlusion. Even when corresponding points are invisible in consecutive frames, the rigid flow can be correctly computed as long as the depth and camera motion estimates are correct.

Stereo. For rigid surfaces in the scene, our method can recover more accurate disparities at pixels with left-right stereo occlusions. This is because computing camera motions over consecutive frames makes it possible to use multi-view stereo matching on temporally adjacent stereo frames in addition to the current frame pair.

Visual Odometry. Explicit motion segmentation makes camera motion recovery more robust. In our method, the binary mask from the previous frame is used to predict which pixels in the current frame are likely to be outliers and must be downweighted during visual odometry estimation.

Motion Segmentation. This task is essentially solved for free in our method. Since the final optimization performed on each frame fuses rigid and non-rigid optical flow proposals (using MRF fusion moves) the resulting binary labeling indicates which pixels belong to non-rigid objects.

6.2 Related Work

Starting with the seminal work by Vedula *et al.* [1999, 2005], the task of estimating scene flow from multiview image sequences has often been formulated as a variational problem [Pons *et al.*, 2003, 2007; Basha *et al.*, 2012; Wedel *et al.*, 2011]. These problems were solved using different optimization methods – Pons *et al.* [2003, 2007] proposed a solution based on level-sets for volumetric representations whereas Basha *et al.* [2012] proposed view-centric representations suiltable for occlusion reasoning and large motions. Previously, Zhang and Kambhamettu [2001] studied how image segmentation cues can help recover accurate motion and depth discontinuities in multi-view scene flow.

Subsequently, the problem was studied in the binocular stereo setting [Li and Sclaroff, 2008; Huguet and Devernay, 2007; Wedel *et al.*, 2011]. Huguet and Devernay [2007] proposed a variational method suitable for the two-view case and Li and Sclaroff [Li and Sclaroff, 2008] proposed a multiscale approach that incorporated uncertainty during coarse to fine processing. Wedel *et al.* [2011] proposed an efficient variational method suitable for GPUs where scene flow recovery was decoupled into two subtasks – disparity

and optical flow estimation. Valgaerts *et al.* [2010] proposed a variational method that dealt with stereo cameras with unknown extrinsics.

Earlier works on scene flow were evaluated on sequences from static cameras or cameras moving in relatively simple scenes (see [Menze and Geiger, 2015] for a detailed discussion). Čech *et al.* [2011]proposed a seed-growing method for sterescopic scene flow which could handle realistic scenes with many moving objects captured by a moving stereo camera. The advent of the KITTI benchmark led to further improvements in scene flow estimation. Vogel *et al.* [2011, 2013, 2014, 2015] recently explored a type of 3D regularization – they proposed a model of dense depth and 3D motion vector fields in [Vogel *et al.*, 2011] and later proposed a piecewise rigid scene model (PRSM) in two [Vogel *et al.*, 2013] and multi-frame settings [Vogel *et al.*, 2014, 2015] that treats scenes as a collection of planar segments undergoing rigid motions. While PRSM [Vogel *et al.*, 2015] is the current top method on KITTI, its joint estimation of 3D geometries, rigid motions and superpixel segmentation using discrete-continuous optimization is fairly complex and computationally expensive. Lv *et al.* [2016] recently proposed a simplified approach to PRSM using continuous optimization and fixed superpixels (named CSF), which is faster than [Vogel *et al.*, 2015] but is still too slow for practical use.

As a closely related approach to ours, object scene flow (OSF) [Menze and Geiger, 2015] segments scenes into multiple rigidly-moving objects based on fixed superpixels, where each object is modeled as a set of planar segments. This model is more rigidly regularized than PRSM. The inference by max-product particle belief propagation is also very computationally expensive taking 50 minutes per frame. A faster setting of their code takes 2 minutes but has lower accuracy.

A different line of work explored scene flow estimation from RGB-D sequences [Herbst *et al.*, 2013; Quiroga *et al.*, 2014; Hornacek *et al.*, 2014; Jaimez *et al.*, 2015b,a; Wang *et al.*, 2016]. Meanwhile, deep convolutional neural network (CNN) based supervised learning methods have shown promise [Mayer *et al.*, 2016] but it is still unclear to what extent it can generalize to new scenes.

6.3 Notations and Preliminaries

Before describing our method in details, we define notations and review basic concepts used in this study.

We denote relative camera motion between two images using matrices $\mathbf{P} = [\mathbf{R}|\mathbf{t}] \in \mathbb{R}^{3\times4}$, which transform homogeneous 3D points $\hat{\mathbf{x}} = (x, y, z, 1)^T$ in camera coordinates of the source image to 3D points $\mathbf{x}' = \mathbf{P}\hat{\mathbf{x}}$ in camera coordinates of the target image. For simplicity, we assume a rectified calibrated stereo system. Therefore, the two cameras have the same known camera intrinsics matrix $\mathbf{K} \in \mathbb{R}^{3\times3}$ and the left-to-right camera pose $\mathbf{P}^{01} = [I| - B\mathbf{e}_x]$ is also known. Here, I is the identity rotation, $\mathbf{e}_x = (1, 0, 0)^T$, and

B is the baseline between the left and right cameras.

We assume the input stereo image pairs have the same size of image domains $\Omega \in \mathbb{Z}^2$ where $\mathbf{p} = (u, v)^T \in \Omega$ is a pixel coordinate. Disparity $\mathcal{D}$, flow $\mathcal{F}$ and segmentation $\mathcal{S}$ are defined as mappings on the image domain Ω, *e.g.*, $\mathcal{D}(\mathbf{p}) : \Omega \to \mathbb{R}^+$, $\mathcal{F}(\mathbf{p}) : \Omega \to \mathbb{R}^2$ and $\mathcal{S}(\mathbf{p}) : \Omega \to \{0, 1\}$.

Given relative camera motion $\mathbf{P}$ and a disparity map $\mathcal{D}$ of the source image, pixels $\mathbf{p}$ of stationary surfaces in the source image are warped to points $\mathbf{p}' = w(\mathbf{p}; \mathcal{D}, \mathbf{P})$ in the target image by the rigid transformation [Hartley and Zisserman, 2004] as

$$w(\mathbf{p}; \mathcal{D}, \mathbf{P}) = \pi \left(\mathbf{KP} \begin{bmatrix} \mathbf{K}^{-1} & \mathbf{0} \\ \mathbf{0}^T & (fB)^{-1} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{p}} \\ \mathcal{D}(\mathbf{p}) \end{bmatrix} \right). \tag{6.1}$$

Here, $\hat{\mathbf{p}} = (u, v, 1)^T$ is the 2D homogeneous coordinate of $\mathbf{p}$, the function $\pi(u, v, w) = (u/w, v/w)^T$ returns 2D non-homogeneous coordinates, and f is the focal length of the cameras. This warping is also used to find which pixels $\mathbf{p}$ in the source image are visible in the target image using z-buffering based visibility test and whether $\mathbf{p}' \in \Omega$.

6.4 Proposed Method

Let I_t^0 and I_t^1, $t \in \{1, 2, \cdots, N+1\}$ be the input image sequences captured by the left and right cameras of a calibrated stereo system, respectively. We sequentially process the first to N-th frames and estimate their disparity maps $\mathcal{D}_t$, flow maps $\mathcal{F}_t$, camera motions $\mathbf{P}_t$ and motion segmentation masks $\mathcal{S}_t$ for the left (reference) images. We call moving and stationary objects as foreground and background, respectively. Below we focus on processing the t-th frame and omit the subscript t when it is not needed.

At a high level, our method is designed to implicitly minimize image residuals

$$E(\mathbf{\Theta}) = \sum_{\mathbf{p}} \| I_t^0(\mathbf{p}) - I_{t+1}^0(w(\mathbf{p}; \mathbf{\Theta})) \| \tag{6.2}$$

by estimating the parameters $\mathbf{\Theta}$ of the warping function w

$$\mathbf{\Theta} = \{\mathcal{D}, \mathbf{P}, \mathcal{S}, \mathcal{F}_{\text{non}}\}. \tag{6.3}$$

The warping function is defined, in the form of the flow map $w(\mathbf{p}; \mathbf{\Theta}) = \mathbf{p} + \mathcal{F}(\mathbf{p})$, using the binary segmentation $\mathcal{S}$ on the reference image I_t^0 as follows.

$$\mathcal{F}(\mathbf{p}) = \begin{cases} \mathcal{F}_{\text{rig}}(\mathbf{p}) & \text{if } \mathcal{S}(\mathbf{p}) = \textit{background} \\ \mathcal{F}_{\text{non}}(\mathbf{p}) & \text{if } \mathcal{S}(\mathbf{p}) = \textit{foreground} \end{cases} \tag{6.4}$$

Here, $\mathcal{F}_{\text{rig}}(\mathbf{p})$ is the rigid flow computed from the disparity map $\mathcal{D}$ and the camera

motion $\mathbf{P}$ using Eq. (6.1), and $\mathcal{F}_{\text{non}}(\mathbf{p})$ is the non-rigid flow defined non-parametrically. Directly estimating this full model is computationally expensive. Instead, we start with a simpler rigid motion model computed from the reduced model parameters $\mathbf{\Theta} = \{\mathcal{D}, \mathbf{P}\}$ (Eq. (6.1)), and then increase the complexity of the motion model by adding non-rigid motion regions $\mathcal{S}$ and their flow $\mathcal{F}_{\text{non}}$. Instead of directly comparing pixel intensities, at various steps of our method, we robustly evaluate the image residuals $\|I(\mathbf{p}) - I'(\mathbf{p}'))\|$ by truncated normalized cross-correlation

$$\text{TNCC}_\tau(\mathbf{p}, \mathbf{p}') = \min\{1 - \text{NCC}(\mathbf{p}, \mathbf{p}'), \tau\}. \tag{6.5}$$

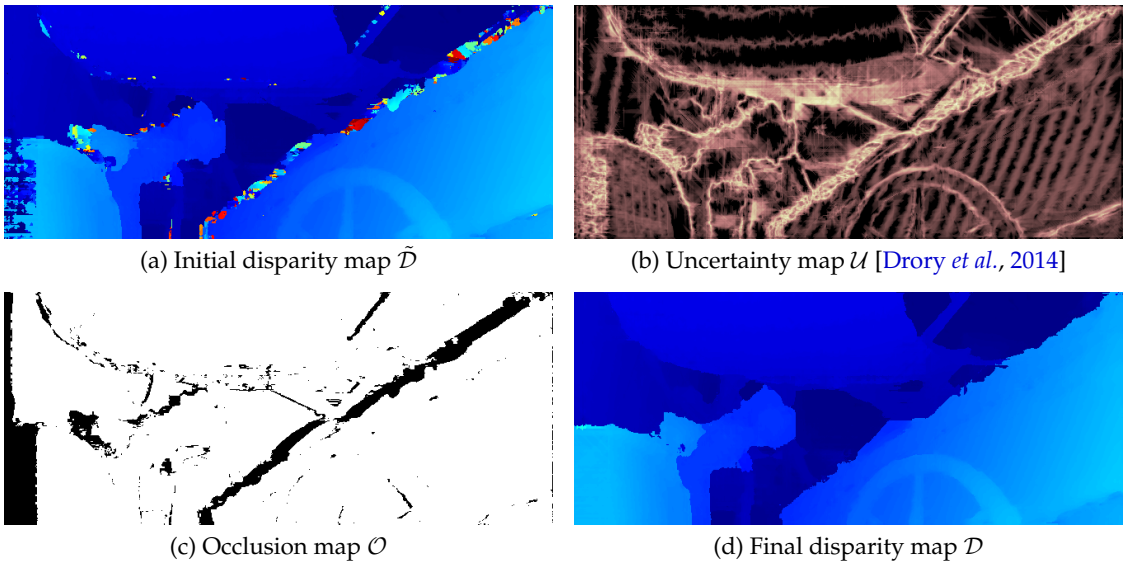
Here, NCC is normalized cross-correlation computed for 5×5 grayscale image patches centered at $I(\mathbf{p})$ and $I'(\mathbf{p}')$, respectively. The thresholding value τ is set to 1.

In the following sections, we describe the proposed pipeline of our method. We first estimate an initial disparity map $\tilde{\mathcal{D}}$ (Section 6.4.1). The disparity map $\tilde{\mathcal{D}}$ is then used to estimate the camera motion $\mathbf{P}$ using visual odometry recovery (Section 6.4.2). This motion estimate $\mathbf{P}$ is used in the epipolar stereo matching stage, where we improve the initial disparity to get the final disparity map $\mathcal{D}$ (Section 6.4.3). The $\mathcal{D}$ and $\mathbf{P}$ estimates are used to compute a rigid flow proposal $\mathcal{F}_{\text{rig}}$ and recover an initial segmentation $\tilde{\mathcal{S}}$ (Section 6.4.4). We then estimate non-rigid flow proposal $\mathcal{F}_{\text{non}}$ for only the moving object regions of $\tilde{\mathcal{S}}$ (Section 6.4.5). Finally we fuse the rigid and non-rigid flow proposals $\{\mathcal{F}_{\text{rig}}, \mathcal{F}_{\text{non}}\}$ and obtain the final flow map $\mathcal{F}$ and segmentation $\mathcal{S}$ (Section 6.4.6). All the steps of the proposed method are summarized in Figure 6.2.

6.4.1 Binocular Stereo

Given left and right images I^0 and I^1, we first estimate an initial disparity map $\tilde{\mathcal{D}}$ of the left image and also its occlusion map $\mathcal{O}$ and uncertainty map $\mathcal{U}$ [Drory *et al.*, 2014]. We visualize example estimates in Figures 6.3 (a)–(c).

As a defacto standard method, we estimate disparity maps by using semi-global matching (SGM) [Hirschmuller, 2008] with a fixed disparity range of $[0, 1, \cdots, D_{\text{max}}]$. Our implementation of SGM uses 8 cardinal directions and NCC-based matching costs of Eq. (6.5) for the data term. The occlusion map $\mathcal{O}$ is obtained by left-right consistency check. The uncertainty map $\mathcal{U}$ is computed during SGM as described in [Drory *et al.*, 2014] without any computational overhead. We also define a fixed confidence threshold τ_{u} for $\mathcal{U}$, *i.e.*, $\tilde{\mathcal{D}}(\mathbf{p})$ is considered unreliable if $\mathcal{U}(\mathbf{p}) > \tau_{\text{u}}$. More details are provided in Appendix D.1.



(a) Initial disparity map $\tilde{\mathcal{D}}$ (b) Uncertainty map $\mathcal{U}$ [Drory *et al.*, 2014]

(c) Occlusion map $\mathcal{O}$ (d) Final disparity map $\mathcal{D}$

Figure 6.3 Binocular and epipolar stereo. (a) Initial disparity map. (c) Uncertainity map [Drory *et al.*, 2014] (darker pixels are more confident). (b) Occlusion map (black pixels are invisible in the right image). (d) Final disparity estimate by epipolar stereo.

6.4.2 Stereo Visual Odometry

Given the current and next image I_t^0 and I_{t+1}^0 and the initial disparity map $\tilde{\mathcal{D}}_t$ of I_t^0, we estimate the relative camera motion $\mathbf{P}$ between the current and next frame. Our method extends an existing stereo visual odometry method [Alismail and Browning , 2014]. This is a direct method, *i.e.*, it estimates the 6-DOF camera motion $\mathbf{P}$ by directly minimizing image intensity residuals

$$E_{\text{vo}}(\mathbf{P}) = \sum_{\mathbf{p}\in T} \omega_{\mathbf{p}}^{\text{vo}} \rho \left(|I_t^0(\mathbf{p}) - I_{t+1}^0(w(\mathbf{p}; \tilde{\mathcal{D}}_t, \mathbf{P}))| \right) \tag{6.6}$$

for some target pixels $\mathbf{p} \in T$, using the rigid warping w of Eq. (6.1). To achieve robustness to outliers (*e.g.*, by moving objects, occlusion, incorrect disparity), the residuals are scored using the Tukey's bi-weight [Beaton and Tukey, 1974] function denoted by ρ. The energy E_{vo} is minimized by iteratively re-weighted least squares in the inverse compositional framework [Baker and Matthews, 2004].

We have modified this method as follows. First, to exploit motion segmentation available in our method, we adjust the weights $\omega_{\mathbf{p}}^{\text{vo}}$ differently. They are set to either 0 or 1 based on the occlusion map $\mathcal{O}(\mathbf{p})$ but later downweighted by $1/8$, if $\mathbf{p}$ is predicted as a moving object point by the previous mask $\mathcal{S}_{t-1}$ and flow $\mathcal{F}_{t-1}$. Second, to reduce sensitivity of direct methods to initialization, we generate multiple diverse initializations for the optimizer and obtain multiple candidate solutions. We then choose the final estimate $\mathbf{P}$ such that best minimizes weighted NCC-based residuals

$E = \sum_{\mathbf{p}\in\Omega} \omega_{\mathbf{p}}^{\text{vo}} \text{TNCC}_\tau(\mathbf{p}, w(\mathbf{p}; \tilde{\mathcal{D}}_t, \mathbf{P}))$. For diverse initializations, we use (a) the identity motion, (b) the previous motion $\mathbf{P}_{t-1}$, (c) a motion estimate by feature-based correspondences using [Lepetit *et al.*, 2009], and (d) various forward translation motions (about 16 candidates, used only for driving scenes).

6.4.3 Epipolar Stereo Refinement

As shown in Figure 6.3 (a), the initial disparity map $\tilde{\mathcal{D}}$ computed from the current stereo pair $\{I_t^0, I_t^1\}$ can have errors at pixels occluded in right image. To address this issue, we use the multi-view epipolar stereo technique on temporarily adjacent six images $\{I_{t-1}^0, I_{t-1}^1, I_t^0, I_t^1, I_{t+1}^0, I_{t+1}^1\}$ and obtain the final disparity map $\mathcal{D}$ shown in Figure 6.1 (d).

From the binocular stereo stage, we already have computed a matching cost volume of I_t^0 for I_t^1, which we denote as $C_{\mathbf{p}}(d)$, with some disparity range $d \in [0, D_{\text{max}}]$. The goal here is to get a better cost volume $C_{\mathbf{p}}^{\text{epi}}(d)$ as input to SGM, by blending $C_{\mathbf{p}}(d)$ with matching costs for each of the four target images $I' \in \{I_{t-1}^0, I_{t-1}^1, I_{t+1}^0, I_{t+1}^1\}$. Since the relative camera poses of the current to next frame $\mathbf{P}_t$ and previous to current frame $\mathbf{P}_{t-1}$ are already estimated by the visual odometry in Section 6.4.2, the relative poses from I_t^0 to each target image can be estimated as $\mathbf{P}' \in \{\mathbf{P}_{t-1}^{-1}, \mathbf{P}^{01}\mathbf{P}_{t-1}^{-1}, \mathbf{P}_t, \mathbf{P}^{01}\mathbf{P}_t\}$, respectively. Recall $\mathbf{P}^{01}$ is the known left-to-right camera pose. Then, for each target image I', we compute matching costs $C'_{\mathbf{p}}(d)$ by projecting points $(\mathbf{p}, d)^T$ in I_t^0 to its corresponding points in I' using the pose $\mathbf{P}'$ and the rigid transformation of Eq. (6.1). Since $C'_{\mathbf{p}}(d)$ may be unreliable due to moving objects, we here lower the thresholding value τ of NCC in Eq. (6.5) to $1/4$ for higher robustness. The four cost volumes are averaged to obtain $C_{\mathbf{p}}^{\text{avr}}(d)$. We also truncate the left-right matching costs $C_{\mathbf{p}}(d)$ at $\tau = 1/4$ at occluded pixels known by $\mathcal{O}(\mathbf{p})$.

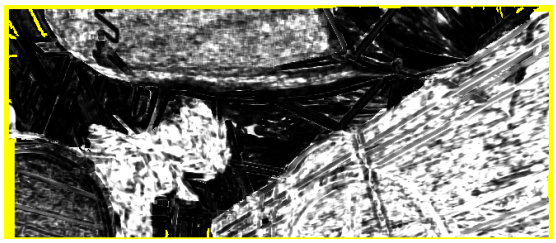
Finally, we compute the improved cost volume $C_{\mathbf{p}}^{\text{epi}}(d)$ by linearly blending $C_{\mathbf{p}}(d)$ with $C_{\mathbf{p}}^{\text{avr}}(d)$ as

$$C_{\mathbf{p}}^{\text{epi}}(d) = (1 - \alpha_{\mathbf{p}})C_{\mathbf{p}}(d) + \alpha_{\mathbf{p}} C_{\mathbf{p}}^{\text{avr}}(d), \tag{6.7}$$

and run SGM with $C_{\mathbf{p}}^{\text{epi}}(d)$ to get the final disparity map $\mathcal{D}$. The blending weights $\alpha_{\mathbf{p}} \in [0, 1]$ are computed from the uncertainty map $\mathcal{U}(\mathbf{p})$ (from Section 6.4.1) normalized as $u_{\mathbf{p}} = \min\{\mathcal{U}(\mathbf{p})/\tau_u, 1\}$ and then converted as follows.

$$\alpha_{\mathbf{p}}(u_{\mathbf{p}}) = \max\{u_{\mathbf{p}} - \tau_{\text{c}}, 0\}/(1 - \tau_{\text{c}}). \tag{6.8}$$

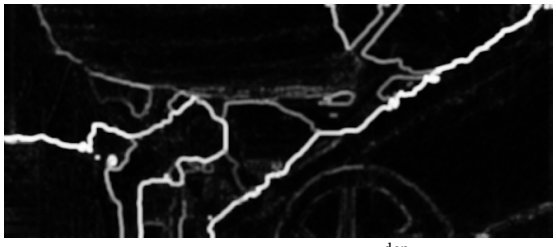
Here, τ_{c} is a confidence threshold. If $u_{\mathbf{p}} \leq \tau_{\text{c}}$, we get $\alpha_{\mathbf{p}} = 0$ and thus $C_{\mathbf{p}}^{\text{epi}} = C_{\mathbf{p}}$. When $u_{\mathbf{p}}$ increases from τ_{c} to 1, $\alpha_{\mathbf{p}}$ linearly increases from 0 to 1. Therefore, we only need to compute $C_{\mathbf{p}}^{\text{avr}}(d)$ at $\mathbf{p}$ where $u_{\mathbf{p}} > \tau_{\text{c}}$, which saves computation. We use $\tau_c = 0.1$.



(a) NCC-based residual map

(b) Patch-intensity variance $w_{\mathbf{p}}^{\text{var}}$

(c) Prior flow $\mathcal{F}_{\text{pri}}$ [Farnebäck, 2003]



(d) Depth edge map $w_{\mathbf{pq}}^{\text{dep}}$

(e) Image edge map $w_{\mathbf{pq}}^{\text{str}}$ [Dollár and Zitnick, 2015]

(f) Initial segmentation $\tilde{S}$

Figure 6.4 Initial segmentation. We detect moving object regions using clues from (a) image residuals weighted by (b) patch-intensity variance and (c) prior flow. We also use (d) depth edge and (e) image edge information to obtain (f) initial segmentation.

6.4.4 Initial Segmentation

During the initial segmentation step, the goal is to find a binary segmentation $\tilde{\mathcal{S}}$ in the reference image I_t^0, which shows where the rigid flow proposal $\mathcal{F}_{\text{rig}}$ is inaccurate and hence optical flow must be recomputed. Recall that $\mathcal{F}_{\text{rig}}$ is obtained from the estimated disparity map $\mathcal{D}$ and camera motion $\mathbf{P}$ using Eq. (6.1). An example of $\tilde{\mathcal{S}}$ is shown in Figure 6.4 (f). We now present the details.

First, we define binary variables $s_{\mathbf{p}} \in \{0, 1\}$ as proxy of $\tilde{\mathcal{S}}(\mathbf{p})$ where 1 and 0 correspond to foreground (moving objects) and background, respectively. Our segmentation energy $E_{\text{seg}}(\mathbf{s})$ is defined as

$$E_{\text{seg}} = \sum_{\mathbf{p}\in\Omega} \left[C_{\mathbf{p}}^{\text{ncc}} + C_{\mathbf{p}}^{\text{flo}} + C_{\mathbf{p}}^{\text{col}} + C_{\mathbf{p}}^{\text{pri}}\right] \bar{s}_{\mathbf{p}} + E_{\text{potts}}(\mathbf{s}). \tag{6.9}$$

Here, $\bar{s}_{\mathbf{p}} = 1 - s_{\mathbf{p}}$. The bracketed terms $[\,\cdot\,]$ are data terms that encode the likelihoods for mask $\tilde{\mathcal{S}}$, *i.e.*, positive values bias $s_{\mathbf{p}}$ toward 1 (moving foreground). E_{potts} is the pairwise smoothness term. We explain each term below.

Appearance Term $C_{\mathbf{p}}^{\text{ncc}}$

This term finds moving objects by checking image residuals of rigidly aligned images. We compute NCC-based matching costs between $I = I_t^0$ and $I' = I_{t+1}^0$ as

$$C'^{\text{ncc}}_{\mathbf{p}}(I, I') = \text{TNCC}_\tau(\mathbf{p}, \mathbf{p}'; I, I') - \tau_{\text{ncc}} \tag{6.10}$$

where $\mathbf{p}' = \mathbf{p} + \mathcal{F}_{\text{rig}}(\mathbf{p})$ and $\tau_{\text{ncc}} \in (0, \tau)$ is a threshold. However, TNCC values are unreliable at texture-less regions (see the high-residual tarp in Figure 6.4 (a)). Furthermore, if $\mathbf{p}'$ is out of field-of-view, $C'^{\text{ncc}}_{\mathbf{p}}$ is not determined (yellow pixels in Figure 6.4 (a)). Thus, similarly to epipolar stereo, we match I_t^0 with $I' \in \{I_{t-1}^0, I_{t-1}^1, I_{t+1}^0, I_{t+1}^1\}$ and compute the average of valid matching costs

$$C_{\mathbf{p}}^{\text{ncc}} = \lambda_{\text{ncc}} w_{\mathbf{p}}^{\text{var}} \, \text{Average}_{I'}\left[C'^{\text{ncc}}_{\mathbf{p}}(I, I')\right]. \tag{6.11}$$

Matching with many images increases the recall for detecting moving objects. To improve matching reliability, $C_{\mathbf{p}}^{\text{ncc}}$ is weighted by $w_{\mathbf{p}}^{\text{var}} = \min(\text{StdDev}(I), \tau_w)/\tau_w$, the truncated standard deviation of the 5×5 patch centered at $I(\mathbf{p})$. The weight map $w_{\mathbf{p}}^{\text{var}}$ is visualized in Figure 6.4 (b). We also truncate $C'^{\text{ncc}}_{\mathbf{p}}(I, I')$ at 0, if $\mathbf{p}'$ is expected to be occluded in I' by visibility test. We use $(\lambda_{\text{ncc}}, \tau_{\text{ncc}}, \tau_w) = (4, 0.5, 0.005)$.

Flow Term $C_{\mathbf{p}}^{\text{flo}}$

This term evaluates flow residuals $r_{\mathbf{p}} = \|\mathcal{F}_{\text{rig}}(\mathbf{p}) - \mathcal{F}_{\text{pri}}(\mathbf{p})\|$ between the rigid flow $\mathcal{F}$ and (non-rigid) prior flow $\mathcal{F}_{\text{pri}}$ computed by [Farnebäck, 2003] (see Figure 6.4 (c)). Using a threshold $\tau_{\mathbf{p}}^{\text{flo}}$ and the patch-variance weight $w_{\mathbf{p}}^{\text{var}}$, we define $C_{\mathbf{p}}^{\text{flo}}$ as

$$C_{\mathbf{p}}^{\text{flo}} = \lambda_{\text{flo}} w_{\mathbf{p}}^{\text{var}} \left[\min(r_{\mathbf{p}}, 2\tau_{\mathbf{p}}^{\text{flo}}) - \tau_{\mathbf{p}}^{\text{flo}}\right]/\tau_{\mathbf{p}}^{\text{flo}}. \tag{6.12}$$

The part after $w_{\mathbf{p}}^{\text{var}}$ normalizes $(r_{\mathbf{p}} - \tau_{\mathbf{p}}^{\text{flo}})$ to lie within $[-1, 1]$. The threshold $\tau_{\mathbf{p}}^{\text{flo}}$ is computed at each pixel $\mathbf{p}$ by

$$\tau_{\mathbf{p}}^{\text{flo}} = \max(\tau^{\text{flo}}, \gamma\|\mathcal{F}_{\text{rig}}(\mathbf{p})\|). \tag{6.13}$$

This way the threshold is relaxed if the rigid motion $\mathcal{F}_{\text{rig}}(\mathbf{p})$ is large. If prior flow $\mathcal{F}_{\text{pri}}(\mathbf{p})$ is invalidated by bi-directional consistency check (black holes in Figure 6.4 (c)), $C_{\mathbf{p}}^{\text{flo}}$ is set to 0. We use $(\lambda_{\text{flo}}, \tau^{\text{flo}}, \gamma) = (4, 0.75, 0.3)$.

Prior Term $C_{\mathbf{p}}^{\text{pri}}$

This term encodes segmentation priors based on results from previous frames or on scene context via ground plane detection. Section 6.4.7 for the details.

Color Term $\mathbf{C_p^{col}}$

This is a standard color-likelihood term [Boykov *et al.*, 2001] for RGB color vectors $\mathbf{I_p}$ of pixels in the reference image $I_t^0(\mathbf{p})$:

$$C_{\mathbf{p}}^{\text{col}} = \lambda_{\text{col}} \Big[\log \theta_1(\mathbf{I_p}) - \log \theta_0(\mathbf{I_p}) \Big]. \tag{6.14}$$

We use $\lambda_{\text{col}} = 0.5$ and 64^3 bins of histograms for the color models $\{\theta_0, \theta_1\}$.

Smoothness Term $\mathbf{E_{potts}}$

This term is based on the Potts model defined for all pairs of neighboring pixels $(\mathbf{p}, \mathbf{q}) \in \mathcal{N}$ on the 8-connected pixel grid.

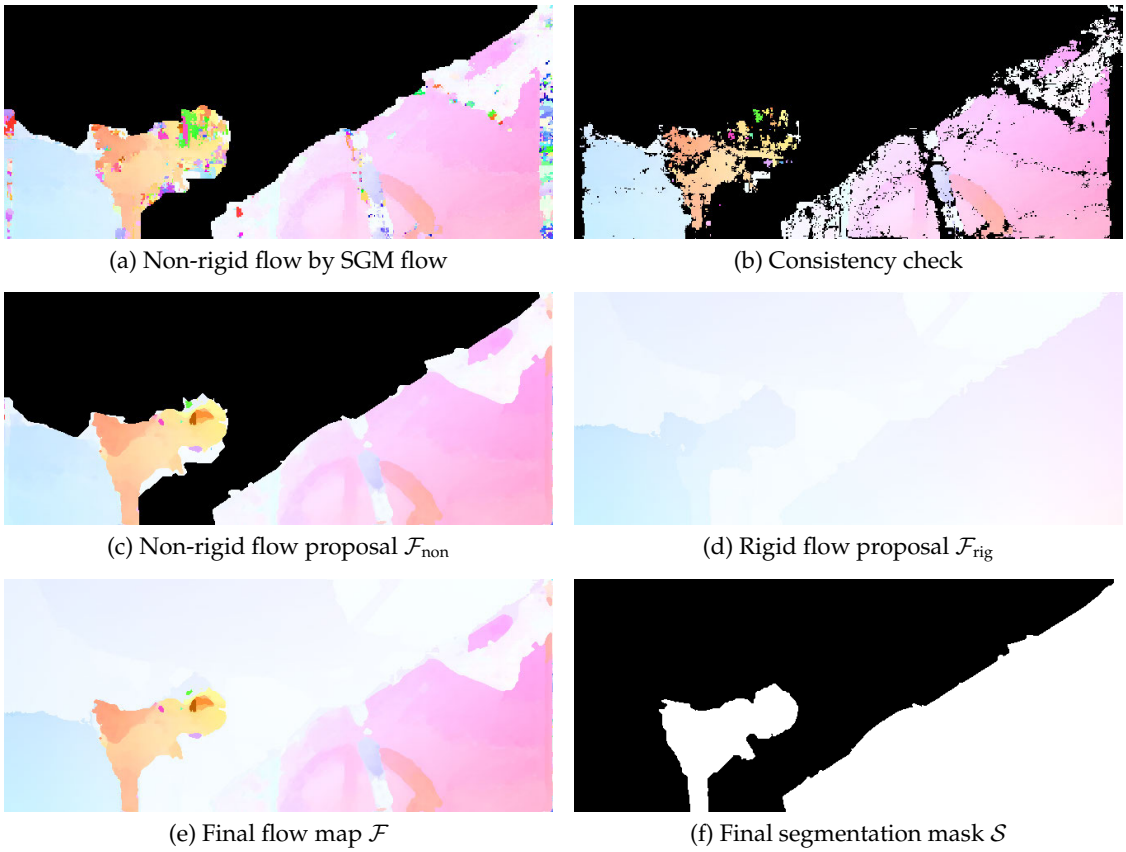
$$E_{\text{potts}}(\mathbf{s}) = \lambda_{\text{potts}} \sum_{(\mathbf{p},\mathbf{q}) \in \mathcal{N}} (\omega_{\mathbf{pq}}^{\text{col}} + \omega_{\mathbf{pq}}^{\text{dep}} + \omega_{\mathbf{pq}}^{\text{str}}) |s_{\mathbf{p}} - s_{\mathbf{q}}|. \tag{6.15}$$

We use three types of edge weights. The color-based weight $\omega_{\mathbf{pq}}^{\text{col}}$ is computed as $\omega_{\mathbf{pq}}^{\text{col}} = e^{-\|\mathbf{I_p} - \mathbf{I_q}\|_2^2 / \kappa_1}$ where κ_1 is estimated as the expected value of $2\|\mathbf{I_p} - \mathbf{I_q}\|_2^2$ over $(\mathbf{p}, \mathbf{q}) \in N$ [Rother *et al.*, 2004]. The depth-based weight $\omega_{\mathbf{pq}}^{\text{dep}}$ is computed as $\omega_{\mathbf{pq}}^{\text{dep}} = e^{-|L_{\mathbf{p}} + L_{\mathbf{q}}| / \kappa_2}$ where $L_{\mathbf{p}} = |\Delta D(\mathbf{p})|$ is the absolute Laplacian of the disparity map $\mathcal{D}$. The κ_2 is estimated similarly to κ_1. The edge-based weight $\omega_{\mathbf{pq}}^{\text{str}}$ uses an edge map $e_{\mathbf{p}} \in [0, 1]$ obtained by a fast edge detector [Dollár and Zitnick, 2015] and is computed as $\omega_{\mathbf{pq}}^{\text{str}} = e^{-|e_{\mathbf{p}} + e_{\mathbf{q}}| / \kappa_3}$. Edge maps of $\omega_{\mathbf{pq}}^{\text{dep}}$ and $\omega_{\mathbf{pq}}^{\text{str}}$ (in the form of $1 - w_{\mathbf{pq}}$) are visualized in Figures 6.4 (d) and (e). We use $(\lambda_{\text{potts}}, \kappa_3) = (10, 0.2)$.

The minimization of $E_{\text{seg}}(\mathbf{s})$ is similar to the GrabCut [Rother *et al.*, 2004] algorithm, *i.e.*, we alternate between minimizing $E_{\text{seg}}(\mathbf{s})$ using graph cuts [Boykov and Kolmogorov, 2004] and updating the color models $\{\theta_1, \theta_0\}$ of $C_{\mathbf{p}}^{\text{col}}$ from segmentation $\mathbf{s}$. We run up to five iterations until convergence using dynamic max-flow [Kohli and Torr, 2007].

6.4.5 Optical Flow

Next, we estimate the non-rigid flow proposal $\mathcal{F}_{\text{non}}$ for the moving foreground regions estimated as the initial segmentation $\tilde{\mathcal{S}}$. Similar to Full Flow [Chen and Koltun, 2016], we pose optical flow as a discrete labeling problem where the labels represent 2D translational shifts with in a 2D search range (see Section 6.4.7 for range estimation). Instead of TRW-S [Kolmogorov, 2006] as used in [Chen and Koltun, 2016], we apply the SGM algorithm as a discrete optimizer. After obtaining a flow map from SGM as shown in Figure 6.5 (a), we filter it further by 1) doing bi-directional consistency check (see Figure 6.5 (b)), and 2) filing holes by weighted median filtering to get the non-rigid flow proposal $\mathcal{F}_{\text{non}}$. The flow consistency map $\mathcal{O}^{\text{flo}}(\mathbf{p})$ is passed to the next stage. Our



(a) Non-rigid flow by SGM flow

(b) Consistency check

(c) Non-rigid flow proposal $\mathcal{F}_{\text{non}}$

(d) Rigid flow proposal $\mathcal{F}_{\text{rig}}$

(e) Final flow map $\mathcal{F}$

(f) Final segmentation mask $\mathcal{S}$

Figure 6.5 Optical flow and flow fusion. We obtain non-rigid flow proposal by (a) performing SGM followed by (b) consistency filtering and (c) hole filing using weighted median filtering. This flow proposal is fused with (d) the rigid flow proposal to obtain (e) the final flow estimate and (f) motion segmentation.

extension of SGM is straightforward and is detailed in Appendix D.2. The details of the refinement scheme is also provide in Appendix D.3.

6.4.6 Flow Fusion and Final Segmentation

Given the rigid and non-rigid flow proposals $\mathcal{F}_{\text{rig}}$ and $\mathcal{F}_{\text{non}}$, we fuse them to obtain the final flow estimate $\mathcal{F}$. This fusion step also produces the final segmentation $\mathcal{S}$. These inputs and outputs are illustrated in Figures 6.5 (c)–(f).

The fusion process is similar to the initial segmentation. The binary variables $s_{\mathbf{p}} \in \{0, 1\}$ indicating the final segmentation $\mathcal{S}$, now also indicate which of the two flow proposals $\{\mathcal{F}_{\text{rig}}(\mathbf{p}), \mathcal{F}_{\text{non}}(\mathbf{p})\}$ is selected as the final flow estimate $\mathcal{F}(\mathbf{p})$. To this end, the energy E_{seg} of Eq. (6.9) is modified as follows. First, $C_{\mathbf{p}}^{\text{ncc}}$ is replaced by

$$C_{\mathbf{p}}^{\text{ncc}} = \lambda_{\text{ncc}} w_{\mathbf{p}}^{\text{var}} [\text{TNCC}_{\tau}(\mathbf{p}, \mathbf{p}'_{\text{rig}}) - \text{TNCC}_{\tau}(\mathbf{p}, \mathbf{p}'_{\text{non}})], \tag{6.16}$$

where $\mathbf{p}'_{\text{rig}} = \mathbf{p} + \mathcal{F}_{\text{rig}}(\mathbf{p})$ and $\mathbf{p}'_{\text{non}} = \mathbf{p} + \mathcal{F}_{\text{non}}(\mathbf{p})$. Second, the prior flow $\mathcal{F}_{\text{pri}}(\mathbf{p})$ in $C_{\mathbf{p}}^{\text{flo}}$

is replaced by $\mathcal{F}_{\text{non}}(\mathbf{p})$. When $\mathbf{p}'_{\text{rig}}$ is out of view or $\mathcal{F}_{\text{non}}(\mathbf{p})$ is invalidated by the flow occlusion map $\mathcal{O}^{\text{flo}}(\mathbf{p})$, we set $C^{\text{ncc}}_{\mathbf{p}}$ and $C^{\text{flo}}_{\mathbf{p}}$ to 0.

This fusion step only infers $s_{\mathbf{p}}$ for pixels labeled foreground in the initial segmentation $\tilde{\mathcal{S}}$, since the background labels are fixed. The graph cut optimization for fusion is typically very efficient, since the pixels labeled foreground in $\tilde{\mathcal{S}}$ is often a small fraction of all the pixels.

6.4.7 Implementation Details

Disparity Range Reduction

For improving the efficiency of epipolar stereo, the disparity range $[0, D_{\text{max}}]$ is reduced by estimating D_{max} from the initially estimated $\tilde{\mathcal{D}}(\mathbf{p})$. We compute D_{max} robustly by making histograms of non-occluded disparities of $\tilde{\mathcal{D}}(\mathbf{p})$ and ignoring bins whose frequency is less than 0.5%. D_{max} is then chosen as the max bin from remaining valid non-zero bins.

Prior Flow

$\mathcal{F}_{\text{pri}}$ is obtained as follows. 1) Run SGM flow for low-resolution images (width of 100 pixels) with a 2D flow range of $[-16, 16]^2$. 2) Do bi-directional consistency check and fill the holes by EpicFlow interpolation [Revaud *et al.*, 2015]. 3) Upscale the flow map and use it as initialization of [Farnebäck, 2003]. We use OpenCV's implementations of EpicFlow and [Farnebäck, 2003].

Flow Range Estimation

The 2D search range $R = ([u_{\text{min}}, u_{\text{max}}] \times [v_{\text{min}}, v_{\text{max}}])$ for SGM flow is estimated as follows. For the target region $\tilde{\mathcal{S}}$, we compute three such ranges from feature-based sparse correspondences, the prior flow and rigid flow. For the latter two, we robustly compute ranges by making 2D histograms of flow vectors and ignoring bins whose frequency is less than one-tenth of the max frequency. Then, the final range R is the range that covers all three. To make R more compact, we repeat the range estimation and subsequent SGM for individual connected components in $\tilde{\mathcal{S}}$.

Cost-Map Smoothing

Since NCC and flow-based cost maps $C^{\text{ncc}}_{\mathbf{p}}$ and $C^{\text{flo}}_{\mathbf{p}}$ used in the segmentation and fusion steps are noisy, we smooth them by averaging the values within superpixels. We use superpixelization of approximately 850 segments produced by [Van den Bergh *et al.*, 2012] in OpenCV.

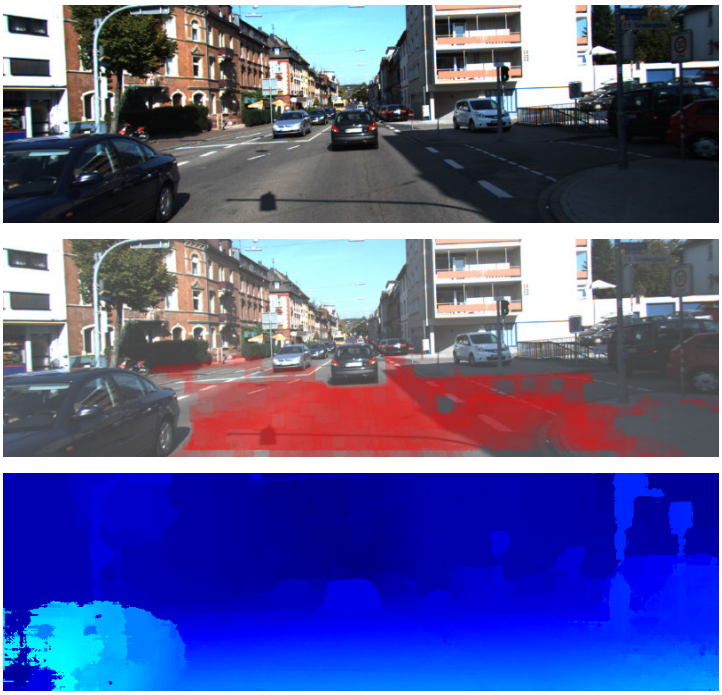


Figure 6.6 Segmentation ground prior. For road scenes (top), we compute the ground prior (middle) from the disparity map (bottom). This prior is used only for the sequences from the KITTI dataset. See also Appendix D.4 for the implementation details.

Segmentation Priors

We define $C_{\mathbf{p}}^{\text{pri}}$ of Eq. (6.9) as $C_{\mathbf{p}}^{\text{pri}} = \lambda_{\text{mask}} C_{\mathbf{p}}^{\text{mask}} + C_{\mathbf{p}}^{\text{pcol}}$. Here, $C_{\mathbf{p}}^{\text{mask}} \in [-0.1, 1]$ is a signed soft mask predicted by previous mask $\mathcal{S}_{t-1}$ and flow $\mathcal{F}_{t-1}$. Negative background regions are downweighted by 0.1 for better detection of new emerging objects. We use $\lambda_{\text{mask}} = 2$. $C_{\mathbf{p}}^{\text{pcol}}$ is a color term similar to Eq. (6.14) with the same λ_{col} but uses color models updated online as the average of past color models. For road scenes, we additionally use the ground prior such as shown in Figure 6.6 as a cue for the background. It is derived by the ground plane detected using RANSAC. See Appendix D.4 for more details.

Others

We run our algorithm on images downscaled by a factor of 0.4 for optical flow and 0.65 for the other steps (each image in KITTI is 1242×375 pixels). We do a subpixel refinement of the SGM disparity and flow maps via standard local quadratic curve fitting [Hirschmuller, 2008].

6.5 Experiments

We evaluate our method on the KITTI 2015 scene flow benchmark [Menze and Geiger, 2015] and further extensively evaluate on the challenging Sintel (stereo) datasets [Butler *et al.*, 2012]. On Sintel we compare with the top two state of the art methods – PRSM [Vogel *et al.*, 2015] and OSF [Menze and Geiger, 2015]. PRSM is a multi-frame method like ours. Although OSF does not explicitly distinguish moving objects from static background in segmentation, the dominant rigid motion bodies are assigned the first object index, which we regarded as background in evaluations. Our method was implemented in C++ and running times were measured on a computer with a quadcore 3.5GHz CPU. All parameter settings were determined using KITTI training data for validation. Only two parameters were re-tuned for Sintel. See also Appendix D.5 for our strategy of tuning parameters.

6.5.1 KITTI 2015 Scene Flow Benchmark

We show a selected ranking of KITTI benchmark results in Table 6.1, where our method is ranked third. Our method is much faster than all the top methods and more accurate than the fast methods [Derome *et al.*, 2016; Čech *et al.*, 2011]. See Figure 6.7 for the per-stage running times. The timings for most stages of our method are small and constant, while for optical flow they vary depending on the size of the moving objects. Results of our method on four sequences are shown in Figures 6.8–6.11. Motion segmentation results are visually quite accurate. As shown in Table 6.2, epipolar stereo refinement using temporarily adjacent stereo frames improves disparity accuracy even for non-occluded pixels. By visual inspection of successive images aligned via the camera motion and depth, we verified that there was never any failure in ego-motion estimation.

6.5.2 Evaluation on Sintel Dataset

Unlike previous scene flow methods, we also evaluated our method on Sintel and compared it with OSF [Menze and Geiger, 2015] and PRSM [Vogel *et al.*, 2015] (see Table 6.3 – best viewed in color). We also show qualitative comparisons on *ambush_5*, *cave_4*, *mountain_1* and *temple_2* in Figures 6.12–6.15. Recall, PRSM does not perform motion segmentation. Although OSF and PRSM are more accurate on KITTI, our method outperforms OSF on Sintel on all metrics. Also, unlike OSF, our method is multi-frame. Sintel scenes have fast, unpredictable camera motion, drastic non-rigid object motion and deformation unlike KITTI where vehicles are the only type of moving objects. While OSF and PRSM need strong rigid regularization, we employ per-pixel inference without requiring piecewise planar assumption. Therefore, our method generalizes more easily to Sintel. Only two parameters had to be modified as follows. $(\lambda_{\text{col}}, \tau_{\text{ncc}}) = (1.5, 0.25)$.

Limitations and Discussions.

The visual odometry step may fail when the scene is far away (see *mountain_1* in Figure 6.14) due to subtle disparity. It may also fail when the moving objects dominate the field of view. Our motion segmentation results are often accurate but in the future we will improve temporal consistency to produce more coherent motion segmentation.

6.6 Summary

We proposed an efficient scene flow method that unifies dense stereo, optical flow, visual odometry, and motion segmentation estimation. Even though simple optimization methods were used in our technique, the unified framework led to higher overall accuracy and efficiency. Our method is currently ranked third on the KITTI 2015 scene flow benchmark after PRSM [Vogel *et al.*, 2015] and OSF [Menze and Geiger, 2015] but is 1–3 orders of magnitude faster than the top six methods. On challenging Sintel sequences, our method outperforms OSF [Menze and Geiger, 2015] and is close to PRSM [Vogel *et al.*, 2015] in terms of accuracy. Our efficient method could be used to initialize PRSM [Vogel *et al.*, 2015] to improve its convergence speed. We hope it will enable new, practical applications of scene flow.

Table 6.1 KITTI 2015 scene flow benchmark results [Menze and Geiger, 2015]. We show the error rates (%) for the disparity on the reference frame (D1) and second frame (D2), the optical flow (Fl) and the scene flow (SF) at background (bg), foreground (fg) and all pixels. Disparity or flow is considered correctly estimated if the end-point error is < 3px or $< 5\%$. Scene flow is considered correct if D1, D2 and Fl are correct.

Rank	Method	D1-bg	D1-fg	D1-all	D2-bg	D2-fg	D2-all
1	PRSM [Vogel *et al.*, 2015]	**3.02**	**10.52**	**4.27**	**5.13**	**15.11**	**6.79**
2	OSF [Menze and Geiger, 2015]	4.54	12.03	5.79	5.45	19.41	7.77
3	**FSF+MS** (ours)	5.72	11.84	6.74	7.57	21.28	9.85
4	CSF [Lv *et al.*, 2016]	4.57	13.04	5.98	7.92	20.76	10.06
5	PR-Sceneflow [Vogel *et al.*, 2013]	4.74	13.74	6.24	11.14	20.47	12.69
8	PCOF+ACTF [Derome *et al.*, 2016]	6.31	19.24	8.46	19.15	36.27	22.00
12	GCSF [Čech *et al.*, 2011]	11.64	27.11	14.21	32.94	35.77	33.41

Method	Fl-bg	Fl-fg	Fl-all	SF-bg	SF-fg	SF-all	Time
PRSM	**5.33**	**17.02**	**7.28**	**6.61**	**23.60**	**9.44**	300 s
OSF	5.62	22.17	8.37	7.01	28.76	10.63	50 min
FSF+MS (ours)	8.48	29.62	12.00	11.17	37.40	15.54	2.7 s
CSF	10.40	30.33	13.71	12.21	36.97	16.33	80 s
PR-Sceneflow	11.73	27.73	14.39	13.49	33.72	16.85	150 s
PCOF+ACTF	14.89	62.42	22.80	25.77	69.35	33.02	0.08 s (GPU)
GCSF	47.38	45.08	47.00	52.92	59.11	53.95	2.4 s

Table 6.2 Disparity improvements by epipolar stereo.

	all pixels			non-occluded pixels		
	D1-bg	D1-fg	D1-all	D1-bg	D1-fg	D1-all
Binocular stereo ($\tilde{\mathcal{D}}$)	7.96	12.61	8.68	7.09	10.57	7.61
Epipolar stereo ($\mathcal{D}$)	**5.82**	**10.34**	**6.51**	**5.57**	**8.84**	**6.06**

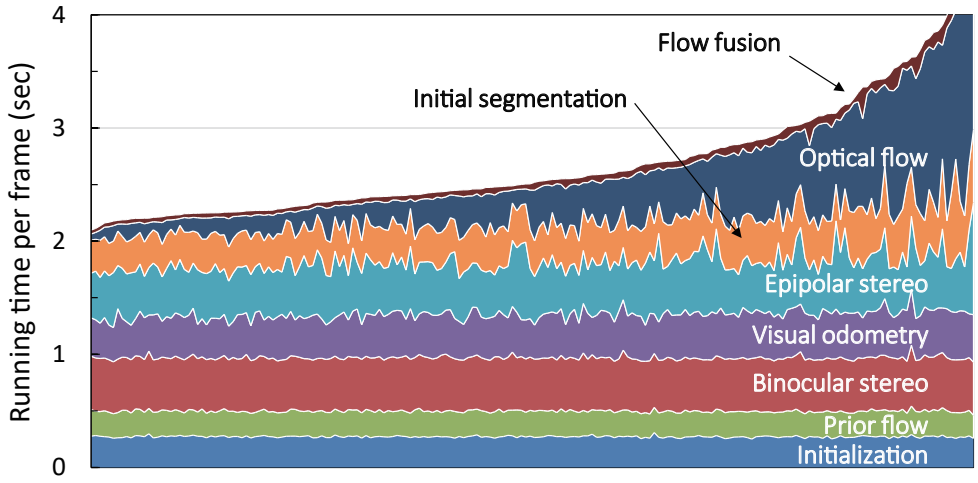
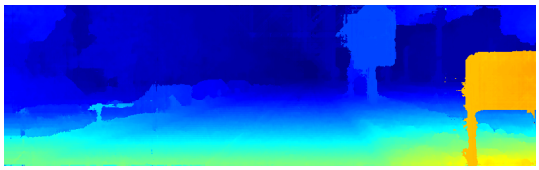


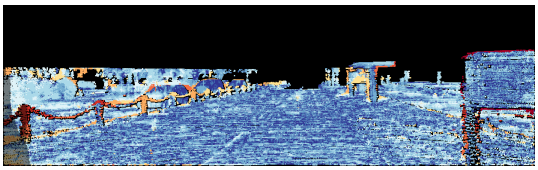
Figure 6.7 Running times on 200 sequences from KITTI. The average running time per-frame was 2.7 sec. Initialization includes edge extraction [Dollár and Zitnick, 2015], superpixelization [Van den Bergh *et al.*, 2012] and feature tracking.

Reference image

Estimated motion segmentation

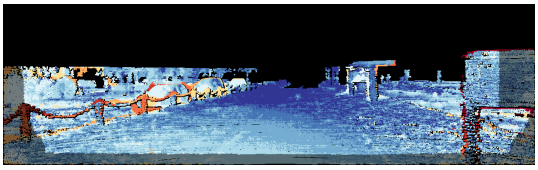


Estimated disparity map



Disparity error map

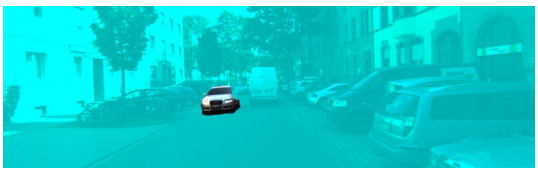
Estimated flow map



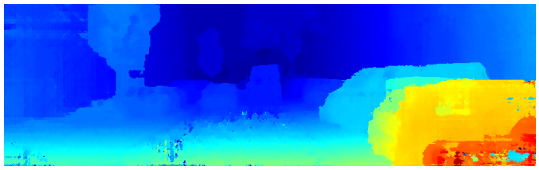
Flow error map

Figure 6.8 Our results on KITTI testing sequences (002). Black pixels in error heat maps indicate missing ground truth while darker shaded pixels indicate occluded pixels.

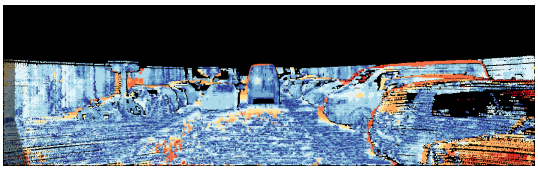
Reference image



Estimated motion segmentation

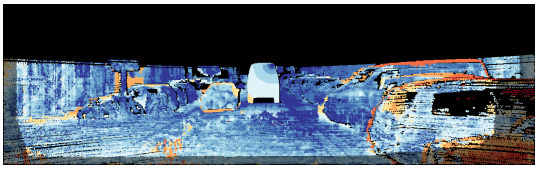


Estimated disparity map



Disparity error map

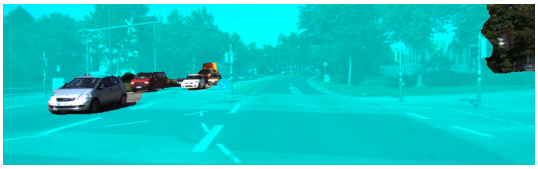
Estimated flow map



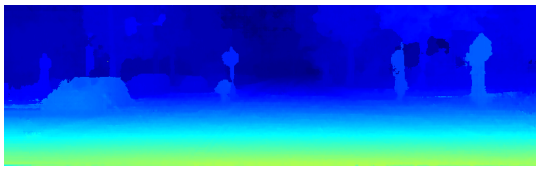
Flow error map

Figure 6.9 Our results on KITTI testing sequences (006). Black pixels in error heat maps indicate missing ground truth while darker shaded pixels indicate occluded pixels.

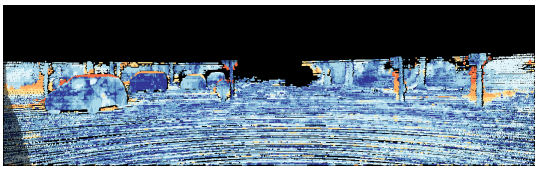
Reference image



Estimated motion segmentation

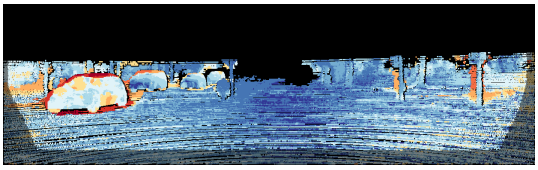


Estimated disparity map



Disparity error map

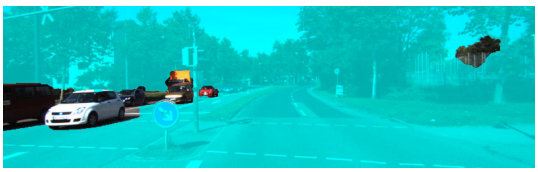
Estimated flow map



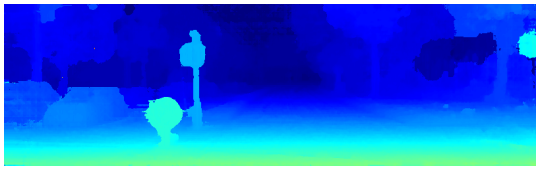
Flow error map

Figure 6.10 Our results on KITTI testing sequences (010). Black pixels in error heat maps indicate missing ground truth while darker shaded pixels indicate occluded pixels.

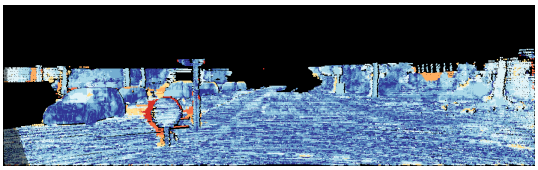
Reference image



Estimated motion segmentation

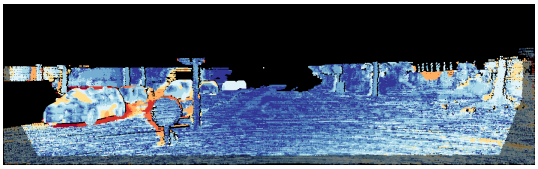


Estimated disparity map



Disparity error map

Estimated flow map



Flow error map

Figure 6.11 Our results on KITTI testing sequences (011). Black pixels in error heat maps indicate missing ground truth while darker shaded pixels indicate occluded pixels.

Table 6.3 Evaluation on Sintel dataset [Butler *et al.*, 2012]. We show error rates (%) for disparity (D1), flow (Fl), scene flow (SF) and motion segmentation (MS) averaged over the frames. Cell colors in OSF [Menze and Geiger, 2015] and PRSM [Vogel *et al.*, 2015] columns show performances relative to ours; blue shows where our method is better, red shows where it is worse. We outperform OSF most of the time.

	D1-all			Fl-all			SF-all			MS-all	
	Ours	OSF	PRSM	Ours	OSF	PRSM	Ours	OSF	PRSM	Ours	OSF
alley_1	5.92	**5.28**	7.43	2.11	7.33	**1.58**	**6.91**	10.04	7.90	**5.40**	17.45
alley_2	2.08	1.31	**0.79**	1.20	1.44	**1.08**	2.99	2.49	**1.63**	1.94	**1.31**
ambush_2	**36.93**	55.13	41.77	72.68	87.37	**51.33**	80.33	90.96	**61.92**	**1.72**	32.76
ambush_4	**23.30**	24.05	24.09	45.23	49.16	**41.99**	49.81	53.25	**46.14**	20.98	**19.82**
ambush_5	18.54	19.54	**17.72**	**24.82**	44.70	25.23	35.15	52.26	**34.12**	**2.50**	19.39
ambush_6	30.33	**26.18**	29.41	44.05	54.75	**41.98**	49.93	58.46	**47.08**	53.95	**24.98**
ambush_7	**23.47**	71.58	35.07	27.87	22.47	**3.35**	44.51	77.94	**36.92**	**26.77**	36.08
bamboo_1	9.67	9.71	**7.34**	4.11	4.04	**2.41**	11.05	10.81	**8.35**	4.43	**4.17**
bamboo_2	19.27	18.08	**17.06**	3.65	4.86	**3.58**	21.39	21.24	**19.23**	**4.08**	4.54
bandage_1	20.93	**19.37**	21.22	4.00	18.40	**3.30**	23.72	36.57	**23.37**	**33.32**	46.66
bandage_2	22.69	23.53	**22.44**	4.76	13.12	**4.06**	24.19	32.33	**23.62**	**16.37**	41.14
cave_4	6.22	5.86	**4.27**	**14.62**	33.94	16.32	17.53	36.04	**17.71**	**16.13**	16.92
market_2	6.81	6.61	**5.27**	5.17	10.08	**4.77**	10.38	14.52	**8.54**	**8.97**	13.90
market_5	**13.25**	13.67	15.38	**26.31**	29.58	28.38	**29.93**	31.60	32.00	**15.26**	15.33
market_6	10.63	10.29	**8.99**	13.13	16.39	**10.72**	18.07	20.18	**15.09**	**3.59**	37.63
mountain_1	**0.23**	0.78	0.42	17.05	88.60	**3.71**	17.05	88.61	**3.85**	31.63	**0.00**
shaman_2	**24.77**	28.27	25.49	**0.56**	1.67	**0.46**	**25.07**	29.43	25.75	30.98	**27.04**
shaman_3	**27.09**	52.22	33.92	**1.31**	11.45	1.75	27.61	55.51	34.43	**3.81**	29.64
sleeping_2	3.52	2.97	**1.74**	0.02	0.01	**0.00**	3.52	2.97	**1.74**	**0.00**	0.54
temple_2	5.96	5.54	**4.92**	9.66	10.52	**9.51**	**9.82**	10.55	9.87	**1.32**	4.13
temple_3	**10.65**	16.62	11.04	62.34	81.39	**32.10**	63.56	81.86	**34.60**	**4.20**	25.42
AVERAGE	**15.35**	19.84	15.99	18.32	28.16	**13.70**	27.26	38.93	**23.52**	**13.68**	19.95

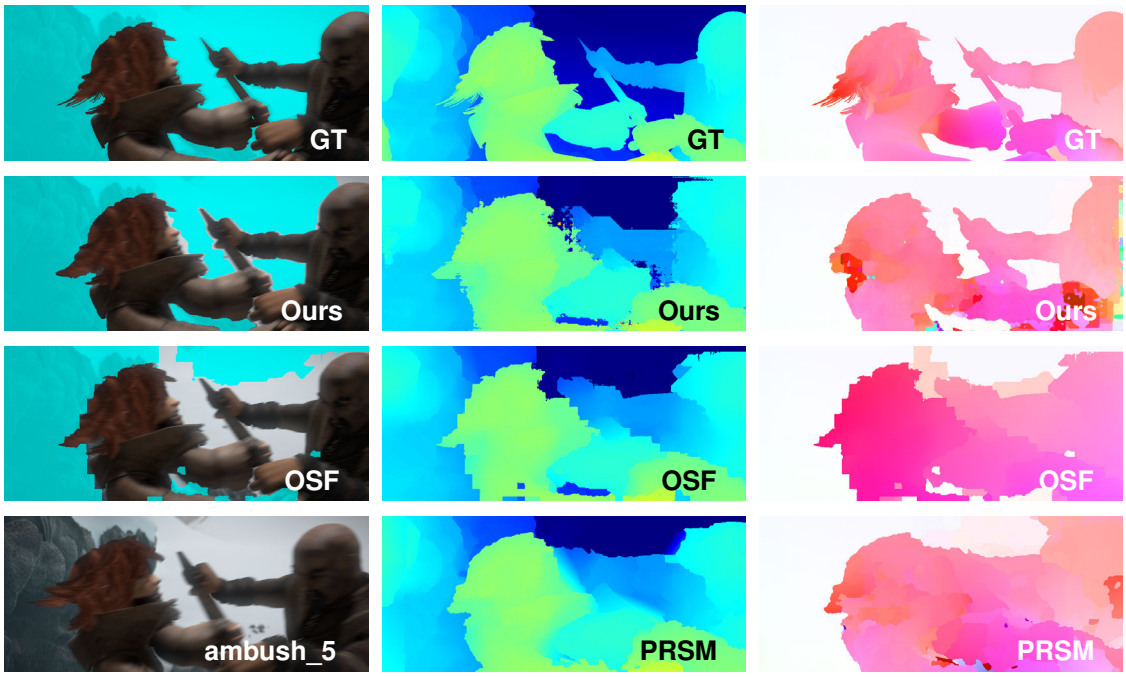


Figure 6.12 Comparisons on *ambush_5* from Sintel. [LEFT] Motion segmentation results – our method, OSF [Menze and Geiger, 2015] and ground truth (GT). [MIDDLE] Disparity and [RIGHT] Flow maps estimated by our method, OSF [Menze and Geiger, 2015] and PRSM [Vogel *et al.*, 2015] and the ground truth versions.

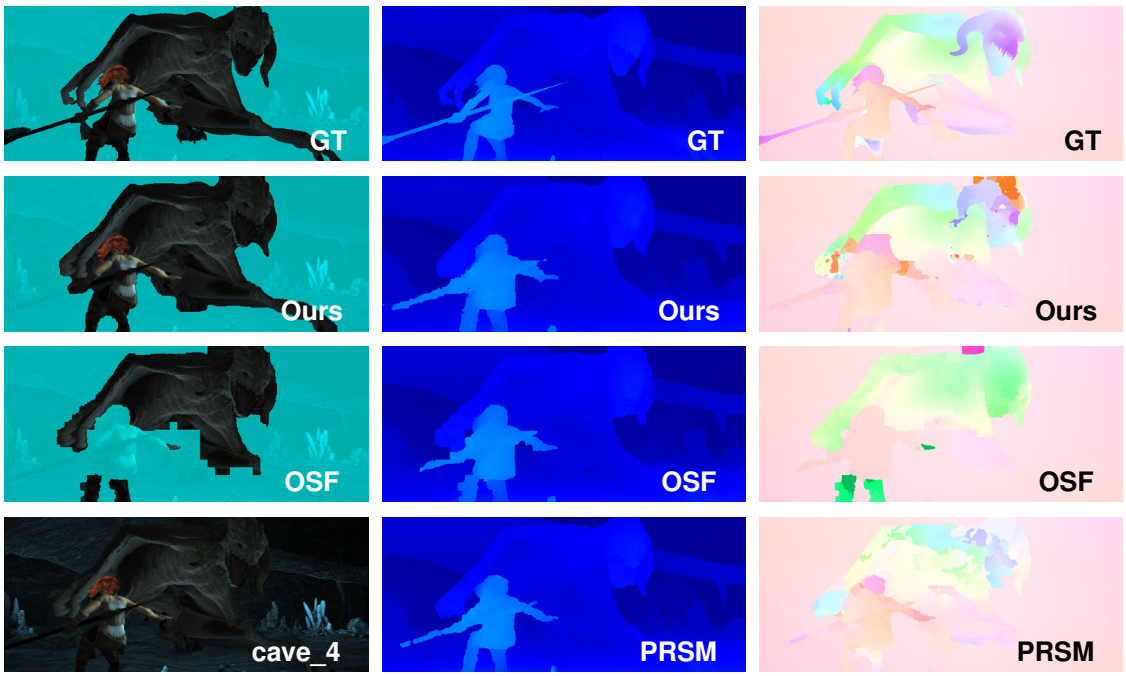


Figure 6.13 Comparisons on *cave_4* from Sintel. See also Figure. 6.12 for descriptions.

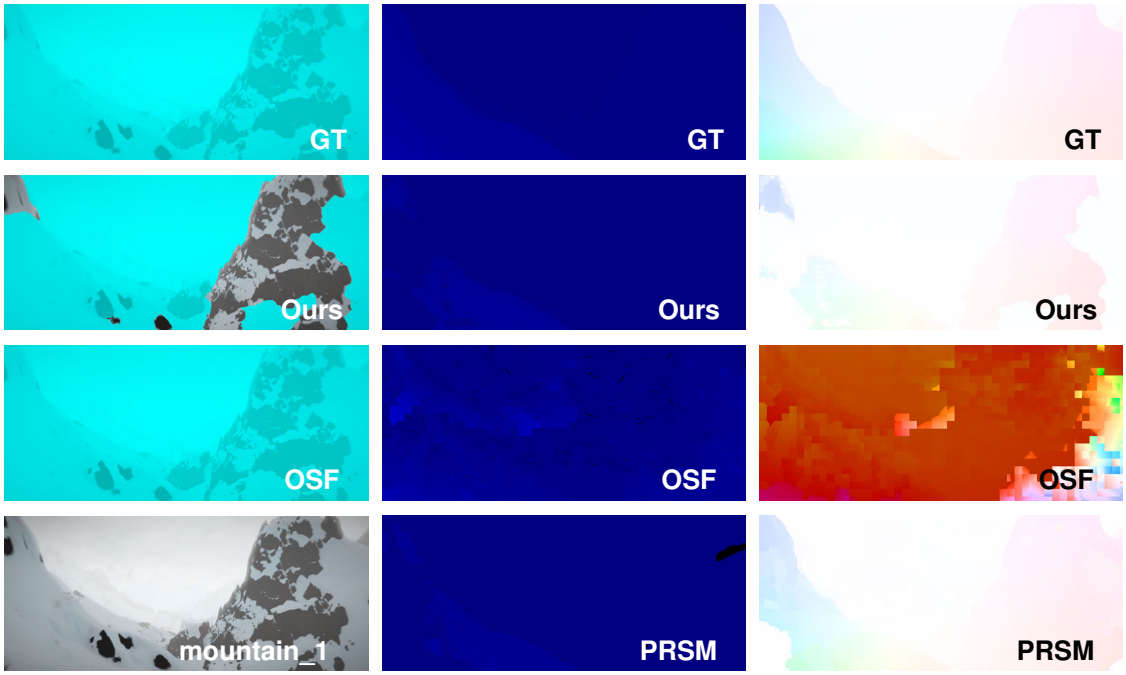


Figure 6.14 Comparisons on *mountain_1* from Sintel. See also Figure. 6.12 for descriptions.

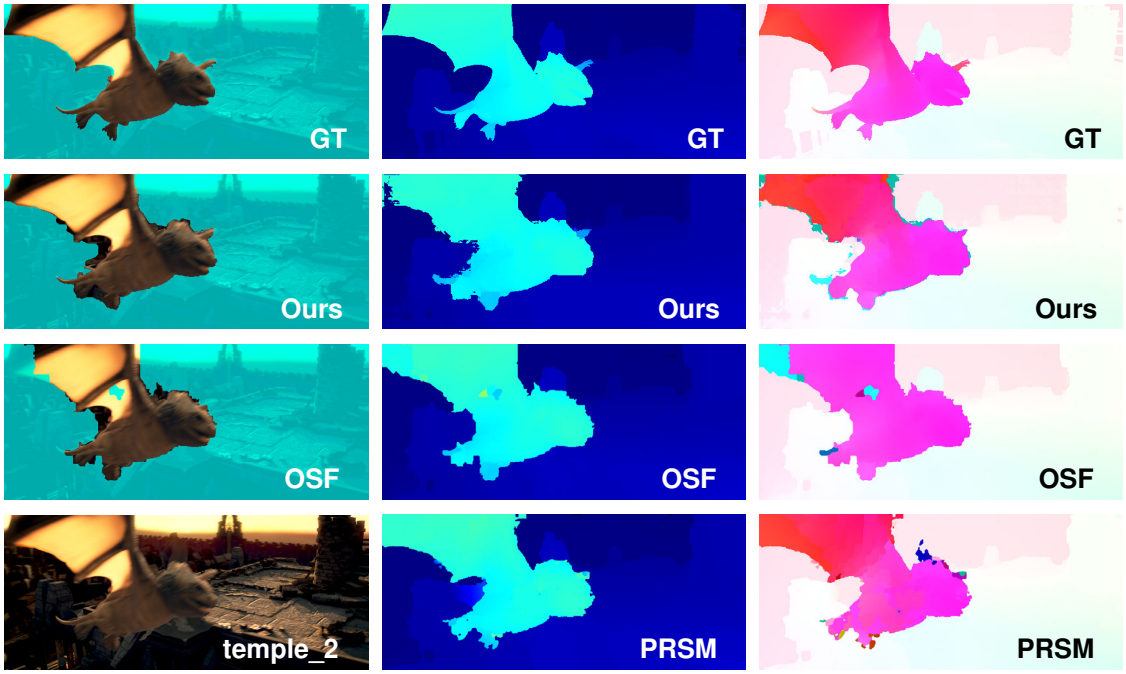


Figure 6.15 Comparisons on *temple_2* from Sintel. See also Figure. 6.12 for descriptions.

7 Conclusion

We now finally conclude this thesis by summarizing contributions and achievements made in this work. We further discuss possible future directions.

7.1 Summary of This Thesis

In this thesis, we studied image segmentation and dense correspondence tasks as well as their inference problems. Several tasks of image segmentation, cosegmentation, motion segmentation, binocular stereo vision, optical flow and general dense correspondence, were tackled sorely or jointly as energy minimization on various types of MRFs. In order to effectively optimize inherently discrete functions or highly non-convex continuous functions, we proposed discrete inference methods that were tailored for individual tasks. We categorized difficulties arising in discrete inference on various types of MRFs into three challenges —label space size, higher-order and non-submodular energy—, which were comprehensively addressed in this work. We now conclude each of the four studies presented in this work below.

In the first study, we tackled inference problems on non-submodular and higher-order MRFs that have binary variables. Such binary energy minimization problems naturally appear in low-level computer vision tasks such as image segmentation and binarization of gray images. The use of such sophisticated models also becomes crucial especially in medical image processing [Kitamura *et al.*, 2016; Gorelick *et al.*, 2012, 2013, 2014] where image observations can provide limited information of gray-scale noisy pixel intensities. Also, binary energy minimization is often imposed during estimation of more general multi-valued or continuous-valued variables. For such fundamental inference problems, we have developed a new theoretical insight that has led to a method unifying several existing optimization methods [Narasimhan and Bilmes, 2005; Ayed *et al.*, 2013, 2015; Gorelick *et al.*, 2014; Tang *et al.*, 2014]. Moreover, the proposed method has a mechanism to better avoid bad local minimums of non-submodular functions, and is thus more robust to initializations compared to existing methods. The proposed

method was evaluated on image segmentation and binarization tasks, where our method outperformed state-of-the-art methods [Narasimhan and Bilmes, 2005; Ayed *et al.*, 2013; Gorelick *et al.*, 2014; Tang *et al.*, 2014] and was shown to be often more efficient.

In the second study, we have proposed an efficient and accurate binocular stereo matching method, whose model and inference are both designed to favor piecewise planar surfaces as a realistic scene assumption. By borrowing the unary data term from [Bleyer *et al.*, 2011] and pairwise smoothness term from [Olsson *et al.*, 2013], we have formulated the proposed method as a pixelwise local 3D surface plane model with piecewise planar smoothness regularization, which forms a pairwise MRF with a continuous 3D label space. In order to efficiently infer this rich model, we have proposed a new inference technique named *local expansion moves*, which extends well-known expansion moves [Boykov *et al.*, 2001] by incorporating the spatial propagation and randomization search mechanisms of PatchMatch [Barnes *et al.*, 2009]. Unlike conventional fusion-based approaches to continuous MRF inference [Lempitsky *et al.*, 2010], the proposed method does not require solution proposals generated by external methods. Furthermore, the proposed local expansion moves produce submodular binary energies during the inference, which are optimally minimized via graph cuts. To further accelerate the computations, we haven shown that the proposed algorithm is easily parallelizable on multiple CPU cores, and also shown that it can incorporate a fast cost-map filtering scheme of [Lu *et al.*, 2013]. The proposed method was evaluated on the Middlebury stereo benchmark [Scharstein and Szeliski, 2002] and achieved the state-of-the-art performance among more than 160 stereo algorithms. The parallelization using four CPU cores and the fast cost-map filtering scheme were shown to gain 3.5x and 5.3x of speed-ups, respectively. Combined together, the faster version of the proposed method achieved more than 18x of speed-ups and even improved accuracy as being able to use more sophisticated filtering [He *et al.*, 2013].

In the third study, we have proposed a joint task of general dense correspondence and cosegmentation for two images. Given two images that show same or similar objects in different scenes, the task is defined to segment out the common object regions in each image as well as finely align the object regions to each other. This task unifies previously proposed two tasks of cosegmentation [Rother *et al.*, 2006] and general dense correspondence [Liu *et al.*, 2011], and is useful in applications such as 3D shape recovery from image clutters [Vicente *et al.*, 2014], label transferring [Smith *et al.*, 2013], and non-parametric scene parsing [Liu *et al.*, 2011; Karsch *et al.*, 2014]. For this problem, we have proposed a hierarchical MRF model with joint labels of segmentation and correspondence. The proposed model parameterizes the correspondence field using piecewise similarity transformations and recovers a mapping between the estimated common object regions. The hierarchy is employed to increase robustness in estimating correspondence between

objects with different appearances, by constraining inference across various coarseness of superpixels. Unlike prior hierarchical methods which assume that the structure is given, we have proposed an iterative technique that dynamically recovers the structure along with the labeling. This joint inference involves higher-order energy optimization. We have proposed an energy minimization approach using iterated graph cuts by extending our local expansion move method. We have also introduced a new dataset that provides 400 image pairs with ground truth segmentation masks and correspondence maps. The proposed method was shown to outperform state-of-the-art methods designed specifically for either cosegmentation [Joulin *et al.*, 2010; Faktor and Irani, 2013] or correspondence estimation [Liu *et al.*, 2011; Kim *et al.*, 2013; Yang *et al.*, 2014].

Finally in the fourth study, we have proposed an efficient scene flow method for stereo image sequences that also recovers motion segmentation of moving objects as well as camera ego-motion. The proposed method unifies four independent tasks — binocular stereo, optical flow, motion segmentation and visual odometry— providing rich information of disparity, 2D flow and binary segmentation indicating moving-objects at every pixel along with the camera motion estimate. Such information is useful in video analysis and editing, 3D mapping, autonomous driving [Menze and Geiger, 2015] and mobile robotics. For the inference, we have proposed a multi-staged framework where the solution to one task benefits others. Even though we employ simple optimization methods in each stage, the proposed framework leads to higher accuracy and also increases computational efficiency. The proposed method was evaluated on the KITTI 2015 scene flow benchmark [Menze and Geiger, 2015] and was ranked third. Furthermore, a CPU implementation of the proposed method ran in 2–3 seconds per frame which was 1–3 orders of magnitude faster than the top six methods taking 1–50 minutes per frame. The proposed method was also thoroughly evaluated on challenging Sintel sequences [Butler *et al.*, 2012] with fast camera and object motion, where our method consistently outperformed [Menze and Geiger, 2015], which was ranked second on the KITTI benchmark.

7.2 Future Directions

In the followings, we discuss possible future directions of this work.

Higher-Order Terms for Continuous MRFs

The optimization technique of higher-oder energies developed in Chapter 3 is based on binary MRF formulations. However, it is interesting to explore inference problems of continuous higher-order MRFs. For example, such formulations appear in the image matting problem, where foreground and background is expressed by a continuous alpha-

matte map as generalization of a binary mask in image segmentation. In a standard benchmark of image matting [Rhemann *et al.*, 2009], many existing methods use continuous inference approaches inspired by a seminal work of [Levin *et al.*, 2008]. The use of discrete-continuous inference enables us to use more sophisticated non-convex models which may lead to higher accuracy.

Continuous MRF Optimization for Other Applications

We believe that our optimization strategy presented in Chapter 4 is not limited to binocular stereo matching problems but can be broadly applied to more general corresponding field estimation such as multi-view stereo and optical flow. In fact, in Chapter 5, the technique was successfully applied to the inference problem of general dense correspondence and cosegmentation. More thorough evaluations on various dense correspondence tasks will verify broad applicability of the proposed optimization technique and will also more contribute to the community.

Dense Correspondence and Cosegmentation for Multiple Images

The joint method of dense correspondence and cosegmentation estimation proposed in Chapter 5 uses two images as input. However, some recent works [Cho *et al.*, 2015; Zhou *et al.*, 2015; Joulin *et al.*, 2012] have shown that the use of multiple images can provide more information and thus leads to higher accuracy of correspondence. There, the concept of cycle consistency [Zhou *et al.*, 2015] has shown to be a powerful constraint for enforcing correspondence consistencies between multiple images. Extending the proposed method for a multi-image case using cycle consistency is a promising direction of this study.

Stereo Scene Flow with Multi-Model Motion Segmentation

The scene flow method proposed in Chapter 6 estimates binary motion segmentation indicating moving and static objects in a scene. Such motion segmentation cannot distinguish multiple independently-moving objects in a scene. Therefore, it is interesting to extend the binary motion segmentation to multi-model motion segmentation. A similar model has been explored by [Menze and Geiger, 2015], but their method is computationally very expensive taking 2–50 minutes per frame. As our method can process each frame in only 2–3 seconds, we can expect a much faster method by extending ours.

Acknowledgements

I would like to express my deepest gratitude to following people, without whom I could not get here.

Firstly, I would thank to Professor Yoichi Sato, who has supervised me for three years during the Ph.D. program. I am really grateful for providing me an excellent research environment without any hesitation, not to mention his valuable advice on research. I personally respect him as a top researcher who has everlasting passion to learn new things, as well as a great educationist who always gives primary consideration to benefits for life of individual persons. I am grateful for showing me such an ideal figure as a researcher and a man.

Secondly, I would thank to the committee members, Professor Yoichi Sato, Professor Kiyoharu Aizawa, Professor Shin'ichi Sato, Professor Takeshi Naemura, Professor Toshihiko Yamasaki, and Professor Hiroshi Ishikawa. Their valuable feedback was indispensable to polish this thesis.

Professor Takeshi Naemura had been also my supervisor for three years during the bachelor and master's programs. A part of achievements in this thesis was made under his supervision. I am really thankful for always urging me to keep high motivation for research. Without his words I could not be as much productive and passionate as I was during the three years. I would also thank for his never changing support even after leaving the laboratory.

Besides my advisors, I would thank to Professor Yasuyuki Matsushita at Osaka University. He was my research mentor during my first internship at Microsoft Research Asia in 2012. The four months that I spent at Microsoft have built my fundamental basis of doing research. Everyday's discussions with him have gradually developed a sense of what makes good research inside me. His words also gave me confidence in my ability, which encouraged me to go on to the Ph.D. program for continuing challenging research.

I would also like thank to Dr. Sudipta Sinha and Dr. David Wipf. Dr. Sinha was my research mentor during internships at Microsoft Research in summer 2015 and 2016. Under his supervision, I did great works that contributed a lot to this thesis. Dr. Wipf was also my mentor in early 2016 while I was visiting Microsoft Research Asia. He gave me wonderful lectures and discussions that have greatly broadened my expertise.

I would also thank to Professor Keita Takahashi at Nagoya University and Dr. Pham Viet Quoc at Toshiba Inc. They also supervised me when I was a bachelor student at Professor Naemura's laboratory. I am grateful for teaching me how exciting and fascinating computer vision is.

I would thank to laboratory staff, Dr. Ryo Yonetani, Dr. Keita Higuchi, Dr. Minjie Cai, Ms. Sakie Suzuki and Ms. Yoko Imagawa for their support. I also thank to other lab members, especially Jinze Yu, Daiki Matsumoto, Rie Kamikubo, Shintaro Murakami, Nattawan Tantirujananont, Yuki Sugita, with whom I joined this laboratory and spent most of the time.

I would like to give special thanks to following people. To staff at Khaosan Atami Hot-Spring Hotel & Hostel where I began and wrote this thesis for a week from November 18, 2016. To friends who enthusiastically joined house-parties that I hosted in these three years. To interns and all other friends whom I met during internships at Microsoft Research in Redmond and Beijing.

Finally, I would thank to my family and parents, who have always believed in me, respected my decisions, and given me all necessary support.

December the 9th, 2016
Tatsunori Taniai

References

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Susstrunk, S. (2012). Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **34**(11), 2274–2282.

Agarwala, A., Dontcheva, M., Agrawala, M., Drucker, S., Colburn, A., Curless, B., Salesin, D., and Cohen, M. (2004). Interactive Digital Photomontage. *ACM Transactions on Graphraphics*, **23**(3), 294–302.

Alismail, H. S. and Browning , B. (2014). Direct disparity space: Robust and real-time visual odometry. Technical Report CMU-RI-TR-14-20, Robotics Institute, Pittsburgh, PA.

Arbelaez, P., Maire, M., Fowlkes, C., and Malik, J. (2011). Contour Detection and Hierarchical Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **33**(5), 898–916.

Ayed, I. B., Chen, H. M., Punithakumar, K., Ross, I., and Li, S. (2010). Graph cut segmentation with a global constraint: Recovering region distribution via a bound of the Bhattacharyya measure. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3288–3295.

Ayed, I. B., Gorelick, L., and Boykov, Y. (2013). Auxiliary Cuts for General Classes of Higher Order Functionals. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1304–1311.

Ayed, I. B., Punithakumar, K., and Li, S. (2015). Distribution matching with the bhattacharyya similarity: A bound optimization framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **37**(9), 1777–1791.

Bai, X., Wang, J., Simons, D., and Sapiro, G. (2009). Video SnapCut: robust video object cutout using localized classifiers. *Proceedings of SIGGRAPH (ACM Transactions on Graphraphics)*, **28**(3), 70:1–70:11.

Baker, S. and Matthews, I. (2004). Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision (IJCV)*, **56**(3), 221–255.

Barnes, C., Shechtman, E., Finkelstein, A., and Goldman, D. B. (2009). PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing. *Proceedings of SIGGRAPH (ACM Transactions on Graphraphics)*, **28**(3), 24:1–24:11.

Barnes, C., Shechtman, E., Goldman, D. B., and Finkelstein, A. (2010). The Generalized Patch-Match Correspondence Algorithm. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 29–43.

Basha, T., Moses, Y., and Kiryati, N. (2012). Multi-view scene flow estimation: A view centered variational approach. *International Journal of Computer Vision (IJCV)*, pages 1–16.

Batra, D., Univerity, C. M., Kowdle, A., Parikh, D., Luo, J., and Chen, T. (2010). icoseg: Interactive co-segmentation with intelligent scribble guidance. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Beaton, A. E. and Tukey, J. W. (1974). The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data. *Technometrics*, **16**(2), 147–185.

Besse, F., Rother, C., Fitzgibbon, A., and Kautz, J. (2012). PMBP: PatchMatch Belief Propagation for Correspondence Field Estimation. In *Proceedings of British Machine Vision Conference (BMVC)*, pages 132.1–132.11.

Besse, F., Rother, C., Fitzgibbon, A. W., and Kautz, J. (2014). PMBP: patchmatch belief propagation for correspondence field estimation. *International Journal of Computer Vision (IJCV)*, **110**(1), 2–13.

Birchfield, S. and Tomasi, C. (1998). A Pixel Dissimilarity Measure That is Insensitive to Image Sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **20**(4), 401–406.

Bleyer, M., Rhemann, C., and Rother, C. (2011). PatchMatch Stereo - Stereo Matching with Slanted Support Windows. In *Proceedings of British Machine Vision Conference (BMVC)*, pages 14.1–14.11.

Boros, E., Hammer, P. L., and Sun, X. (1991). Network Flows and Minimization of Quadratic Pseudo-Boolean Functions. Technical Report RRR 17-1991, RUTCOR Research Report.

Boykov, Y. and Kolmogorov, V. (2004). An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **26**(9), 1124–1137.

Boykov, Y., Veksler, O., and Zabih, R. (1998). Markov Random Fields with Efficient Approximations. In *Proceedings of International Conference on Computer Vision (ICCV)*, pages 648—655.

Boykov, Y., Veksler, O., and Zabih, R. (2001). Fast Approximate Energy Minimization via Graph Cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **23**(11), 1222–1239.

Boykov, Y., Komodakis, N., Kolmogorov, V., and Torr, P. (2007). Discrete Optimization in Computer Vision. In *Tutorials at International Conference on Computer Vision (ICCV)*. http://www.csd.uoc.gr/~komod/ICCV07_tutorial/.

Boykov, Y. Y. and Jolly, M.-P. (2001). Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *Proceedings of International Conference on Computer Vision (ICCV)*, volume 1, pages 105–112.

Butler, D. J., Wulff, J., Stanley, G. B., and Black, M. J. (2012). A naturalistic open source movie for optical flow evaluation. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 611–625.

Cech, J., Matas, J., and Perdoch, M. (2010). Efficient sequential correspondence selection by cosegmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **32**(9), 1568–1581.

Čech, J., Sanchez-Riera, J., and Horaud, R. (2011). Scene flow estimation by growing correspondence seeds. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3129–3136.

Chen, Q. and Koltun, V. (2016). Full flow: Optical flow estimation by global optimization over regular grids. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Chen, X., Shrivastava, A., and Gupta, A. (2014). Enriching visual knowledge bases via object discovery and segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2035–2042.

Cho, M., Kwak, S., Schmid, C., and Ponce, J. (2015). Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1201–1210.

Criminisi, A., Sharp, T., and Blake, A. (2008). Geos: Geodesic image segmentation. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 99–112.

Dai, J., Wu, Y. N., Zhou, J., and Zhu, S.-C. (2013). Cosegmentation and cosketch by unsupervised learning. In *Proceedings of International Conference on Computer Vision (ICCV)*, pages 1305–1312.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 886–893.

Delong, A., Osokin, A., Isack, H. N., and Boykov, Y. (2012). Fast Approximate Energy Minimization with Label Costs. *International Journal of Computer Vision (IJCV)*, **96**(1), 1–27.

Derome, M., Plyer, A., Sanfourche, M., and Le Besnerais, G. (2016). A prediction-correction approach for real-time optical flow computation using stereo. In *Proc. of German Conference on Pattern Recognition*, pages 365–376.

Dollár, P. and Zitnick, C. L. (2015). Fast edge detection using structured forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*.

Drory, A., Haubold, C., Avidan, S., and Hamprecht, F. A. (2014). Semi-global matching: a principled derivation in terms of message passing. *Pattern Recognition*, pages 43–53.

Duchenne, O., Bach, F., Kweon, I.-S., and Ponce, J. (2011). A tensor-based algorithm for high-order graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **33**(12), 2383–2395.

El-Zehiry, N. and Grady, L. (2010). Fast Global Optimization of Curvature. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3257–3264.

Facciolo, G., de Franchis, C., and Meinhardt, E. (2015). MGM: A Significantly More Global Matching for Stereovision. In *Proceedings of British Machine Vision Conference (BMVC)*, pages 90.1–90.12.

Faktor, A. and Irani, M. (2013). Co-segmentation by composition. In *Proceedings of International Conference on Computer Vision (ICCV)*, pages 1297–1304.

Farnebäck, G. (2003). Two-frame motion estimation based on polynomial expansion. In *Proc. of the 13th Scandinavian Conference on Image Analysis*, SCIA'03, pages 363–370.

Felzenszwalb, P. and Huttenlocher, D. (2004). Efficient Belief Propagation for Early Vision. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 261–268.

Felzenszwalb, P. F. and Huttenlocher, D. P. (2006). Efficient belief propagation for early vision. *International Journal of Computer Vision (IJCV)*, **70**(1), 41–54.

Fujishige, S. (2005). *Submodular Functions and Optimization*, volume 58. Elsevier Science.

Geman, S. and Geman, D. (1984). Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **6**(6), 721–741.

Gorelick, L., Schmidt, F. R., Boykov, Y., Delong, A., and Ward, A. D. (2012). Segmentation with non-linear regional constraints via line-search cuts. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 583–0597.

Gorelick, L., Schmidt, F. R., and Boykov, Y. (2013). Fast Trust Region for Segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1714–1721.

Gorelick, L., Boykov, Y., Veksler, O., Ben Ayed, I., and Delong, A. (2014). Submodularization for Binary Pairwise Energies. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Greig, D. M., Porteous, B. T., and Seheult, A. H. (1986). Discussion of: On the Statistical Analysis of Dirty Pictures (by J. E. Besag.). *Journal of the Royal Statistical Society*, **48**, 282—-284.

Greig, D. M., Porteous, B. T., and Seheult, A. H. (1989). Exact Maximum A Posteriori Estimation for Binary Images. *Journal of the Royal Statistical Society*, **51**, 271–279.

HaCohen, Y., Shechtman, E., Goldman, D. B., and Lischinski, D. (2011). Non-rigid dense correspondence with applications for image enhancement. *ACM Transactions on Graphraphics*, **30**(4), 70:1–70:10.

Hammer, P., Hansen, P., and Simeone, B. (1984). Roof Duality, Complementation and Persistency in Quadratic 0-1 Optimization. *Mathematicl Programming*, **28**, 121–155.

Hammer, P. L. (1965). Some network flow problems solved with pseudo-boolean programming. *Operations Research*, **13**(3), 388–399.

Hariharan, B., Arbelaez, P., Bourdev, L., Maji, S., and Malik, J. (2011). Semantic contours from inverse detectors. In *Proceedings of International Conference on Computer Vision (ICCV)*.

Hartley, R. I. and Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition.

Hassner, T., Mayzels, V., and Zelnik-Manor, L. (2012). On sifts and their scales. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

He, K., Sun, J., and Tang, X. (2010). Guided image filtering. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 1–14.

He, K., Sun, J., and Tang, X. (2013). Guided Image Filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **35**(6), 1397–1409.

He, X., Zemel, R. S., and Carreira-Perpiñán, M. Á. (2004). Multiscale conditional random fields for image labeling. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 695–702.

Heise, P., Klose, S., Jensen, B., and Knoll, A. (2013). PM-Huber: PatchMatch with Huber Regularization for Stereo Matching. In *Proceedings of International Conference on Computer Vision (ICCV)*. (accepted).

Herbst, E., Ren, X., and Fox, D. (2013). Rgb-d flow: Dense 3-d motion estimation using color and depth. In *Proceedings of International Conference on Robotics and Automation (ICRA)*, pages 2276–2282.

Hirschmuller, H. (2008). Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **30**(2), 328–341.

Hirschmuller, H. and Scharstein, D. (2007). Evaluation of Cost Functions for Stereo Matching. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hong, L. and Chen, G. (2004). Segment-based Stereo Matching using Graph Cuts. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 74–81.

Horn, B. K. P. and Schunck, B. G. (1981). Determining optical flow. *Artificial Intelligence*, **17**, 185–203.

Hornacek, M., Fitzgibbon, A., and Rother, C. (2014). Sphereflow: 6 dof scene flow from rgb-d pairs. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3526–3533.

Hosni, A., Gelautz, M., and Bleyer, M. (2012). Accuracy-Efficiency Evaluation of Adaptive Support Weight Techniques for Local Stereo Matching. In *Proceedings of DAGM/OAGM Symposium*, pages 337–346.

Hosni, A., Rhemann, C., Bleyer, M., Rother, C., and Gelautz, M. (2013). Fast cost-volume filtering for visual correspondence and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **35**(2), 504–511.

Huguet, F. and Devernay, F. (2007). A variational method for scene flow estimation from stereo sequences. In *Proceedings of International Conference on Computer Vision (ICCV)*, pages 1–7.

Hur, J., Lim, H., Park, C., and Ahn, S. C. (2015). Generalized deformable spatial pyramid: Geometry-preserving dense correspondence estimation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1392–1400.

Ihler, A. and McAllester, D. (2009). Particle Belief Propagation. *International Conference on Artificial Intelligence and Statistics*, **5**, 256–263.

Isack, H. and Boykov, Y. (2012). Energy-Based Geometric Multi-model Fitting. *International Journal of Computer Vision (IJCV)*, **97**(2), 123–147.

Ishikawa, H. (2003). Exact Optimization for Markov Random Fields with Convex Priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **25**(10), 1333–1336.

Ishikawa, H. (2011). Transformation of General Binary MRF Minimization to the First-Order Case. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **33**(6), 1234–1249.

Ishikawa, H. and Geiger, D. (1998a). Occlusions, discontinuities, and epipolar lines in stereo. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 232—-248.

Ishikawa, H. and Geiger, D. (1998b). Segmentation by grouping junctions. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 125—-131.

Iwata, S., Fleischer, L., and Fujishige, S. (2001). A Combinatorial Strongly Polynomial Algorithm for Minimizing Submodular Functions. *Journal of the ACM*, **48**(4), 761–777.

Iyer, R., Jegelka, S., and Bilmes, J. (2013). Fast Semidifferential-based Submodular Function Optimization. In *Proceedings of International Conference on Machine Learning (ICML)*, volume 28, pages 855–863.

Jaimez, M., Souiai, M., Stueckler, J., Gonzalez-Jimenez, J., and Cremers, D. (2015a). Motion cooperation: Smooth piece-wise rigid scene flow from rgb-d images. In *Proc. of the Int. Conference on 3D Vision (3DV)*.

Jaimez, M., Souiai, M., Gonzalez-Jimenez, J., and Cremers, D. (2015b). A primal-dual framework for real-time dense rgb-d scene flow. In *Proceedings of International Conference on Robotics and Automation (ICRA)*, pages 98–104.

Jegelka, S. and Bilmes, J. (2011). Submodularity beyond Submodular Energies: Coupling Edges in Graph Cuts. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1897–1904.

Jiang, H. (2012). Linear Solution to Scale Invariant Global Figure Ground Separation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 678–685.

Joulin, A., Bach, F., and Ponce, J. (2010). Discriminative clustering for image co-segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Joulin, A., Bach, F., and Ponce, J. (2012). Multi-class cosegmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Joulin, A., Tang, K., and Fei-Fei, L. (2014). Efficient image and video co-localization with frank-wolfe algorithm. In *Proceedings of European Conference on Computer Vision (ECCV)*.

Karsch, K., Liu, C., and Kang, S. B. (2014). DepthTransfer: Depth extraction from video using non-parametric sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*.

Kennedy, R. and Taylor, C. J. (2015). Hierarchically-constrained optical flow. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kim, G., Xing, E. P., Fei-Fei, L., and Kanade, T. (2011). Distributed cosegmentation via submodular optimization on anisotropic diffusion. In *Proceedings of International Conference on Computer Vision (ICCV)*, pages 169–176. IEEE.

Kim, J., Liu, C., Sha, F., and Grauman, K. (2013). Deformable spatial pyramid matching for fast dense correspondences. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2307–2314.

Kitamura, Y., Li, Y., Ito, W., and Ishikawa, H. (2016). Data-Dependent Higher-Order Clique Selection for Artery–Vein Segmentation by Energy Minimization. *International Journal of Computer Vision (IJCV)*, **117**(2), 142–158.

Klaus, A., Sormann, M., and Karner, K. (2006). Segment-Based Stereo Matching Using Belief Propagation and a Self-Adapting Dissimilarity Measure. In *Proceedings of International Conference on Pattern Recognition (ICPR)*, volume 3, pages 15–18.

Kohli, P. and Torr, P. H. S. (2007). Dynamic Graph Cuts for Efficient Inference in Markov Random Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **29**(12), 2079–2088.

Kohli, P., Kumar, M. P., and Torr, P. H. S. (2007). P3 Beyond: Solving Energies with Higher Order Cliques. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kohli, P., Ladický, L., and Torr, P. H. (2009). Robust higher order potentials for enforcing label consistency. *International Journal of Computer Vision (IJCV)*, **82**(3).

Kohli, P., Osokin, A., and Jegelka, S. (2013). A Principled Deep Random Field Model for Image Segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1971–1978.

Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press.

Kolmogorov, V. (2006). Convergent Tree-Reweighted Message Passing for Energy Minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **28**(10), 1568–1583.

Kolmogorov, V. and Rother, C. (2007). Minimizing Nonsubmodular Functions with Graph Cuts – A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **29**(7), 1274–1279.

Kolmogorov, V. and Zabih, R. (2001). Computing Visual Correspondence with Occlusions using Graph Cuts. In *Proceedings of International Conference on Computer Vision (ICCV)*, volume 2, pages 508–515.

Kolmogorov, V. and Zabih, R. (2002). Multi-camera Scene Reconstruction via Graph Cuts. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 82–96.

Kolmogorov, V. and Zabin, R. (2004). What Energy Functions Can Be Minimized via Graph Cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **26**(2), 147–159.

Kolmogorov, V., Boykov, Y., and Rother, C. (2007). Applications of Parametric Maxflow in Computer Vision. In *Proceedings of International Conference on Computer Vision (ICCV)*.

Kothapa, R., Pacheco, J., and Sudderth, E. B. (2011). *Max-Product Particle Belief Propagation*. Master's thesis, Brown University.

Kowdle, A., Sinha, S. N., and Szeliski, R. (2012). Multiple view object cosegmentation using appearance and stereo cues. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 789–803. Springer Berlin Heidelberg.

Kwatra, V., Schödl, A., Essa, I., Turk, G., and Bobick, A. (2003). Graphcut Textures: Image and Video Synthesis Using Graph Cuts. *ACM Transactions on Graphraphics*, **22**(3), 277–286.

Laude, E., Mollenhoff, T., Moeller, M., Lellmann, J., and Cremers, D. (2016). Sublabel-accurate convex relaxation of vectorial multilabel energies. In *Proceedings of European Conference on Computer Vision (ECCV)*.

Lei, C. and Yang, Y.-H. (2009). Optical flow estimation on coarse-to-fine region-trees using discrete optimization. In *Proceedings of International Conference on Computer Vision (ICCV)*, pages 1562–1569. IEEE.

Lei, C., Selzer, J., and Yang, Y.-H. (2006). Region-tree based stereo using dynamic programming optimization. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2378–2385.

Lempitsky, V., Rother, C., and Blake, A. (2007). LogCut - Efficient Graph Cut Optimization for Markov Random Fields. In *Proceedings of International Conference on Computer Vision (ICCV)*, pages 1–8.

Lempitsky, V., Roth, S., and Rother, C. (2008). FusionFlow: Discrete-Continuous Optimization for Optical Flow Estimation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Lempitsky, V., Rother, C., Roth, S., and Blake, A. (2010). Fusion Moves for Markov Random Field Optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **32**(8), 1392–1405.

Lepetit, V., F.Moreno-Noguer, and P.Fua (2009). EPnP: An Accurate O(n) Solution to the PnP Problem. *International Journal of Computer Vision (IJCV)*, **81**(2).

Levin, A., Rav-Acha, A., and Lischinski, D. (2008). Spectral Matting. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **30**(10), 1699–1712.

Li, R. and Sclaroff, S. (2008). Multi-scale 3d scene flow from binocular stereo sequences. *Computer Vision and Image Understanding*, **110**(1), 75–90.

Lin, Y.-L., Morariu, V. I., Hsu, W., and Davis, L. S. (2014). Jointly optimizing 3d model fitting and fine-grained classification. In *Proceedings of European Conference on Computer Vision (ECCV)*.

Liu, C., Yuen, J., and Torralba, A. (2011). SIFT Flow: Dense Correspondence across Scenes and Its Applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **33**(5), 978–994.

Liu, J. and Sun, J. (2010). Parallel graph-cuts by adaptive bottom-up merging. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2181–2188.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, **60**(2), 91–110.

Lu, J., Shi, K., Min, D., Lin, L., and Do, M. (2012). Cross-based Local Multipoint Filtering. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 430–0437.

Lu, J., Yang, H., Min, D., and Do, M. N. (2013). Patch Match Filter: Efficient Edge-Aware Filtering Meets Randomized Search for Fast Correspondence Field Estimation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1854–1861.

Lucas, B. D. and Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 674–679.

Lv, Z., Beall, C., Alcantarilla, P. F., Li, F., Kira, Z., and Dellaert, F. (2016). A continuous optimization approach for efficient and accurate scene flow. In *Proceedings of European Conference on Computer Vision (ECCV)*.

Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., and Brox, T. (2016). A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Menze, M. and Geiger, A. (2015). Object scene flow for autonomous vehicles. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3061–3070.

Mollenhoff, T., Laude, E., Moeller, M., Lellmann, J., and Cremers, D. (2016). Sublabel-accurate relaxation of nonconvex energies. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Monasse, P. (2011). Quasi-Euclidean Epipolar Rectification. *Image Processing On Line*. http://www.ipol.im/pub/art/2011/m_qer/.

Mukherjee, L., Singh, V., and Dyer, C. (2009). Half-Integrality based Algorithms for Cosegmentation of Images. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2028–2035.

Murphy, K. P., Weiss, Y., and Jordan, M. I. (1999). Loopy Belief Propagation for Approximate Inference: An Empirical Study. In *Proceedings of Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 467–475.

Narasimhan, M. and Bilmes, J. A. (2005). A Submodular-supermodular Procedure with Applications to Discriminative Structure Learning. In *Proceedings of Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 404–412.

Olsson, C., Ulen, J., and Boykov, Y. (2013). In Defense of 3D-Label Stereo. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1730–1737.

Pacheco, J., Zuffi, S., Black, M. J., and Sudderth, E. (2014). Preserving Modes and Messages via Diverse Particle Selection. In *Proceedings of International Conference on Machine Learning (ICML)*, volume 32, pages 1152–1160.

Petschnigg, G., Szeliski, R., Agrawala, M., Cohen, M., Hoppe, H., and Toyama, K. (2004). Digital photography with flash and no-flash image pairs. *ACM Transactions on Graphraphics*, **23**(3), 664–672.

Pham, V.-Q., Takahashi, K., and Naemura, T. (2010). Real-Time video matting based on bilayer segmentation. In *Proceedings of Asian Conference on Computer Vision (ACCV)*, volume 2, pages 489–501.

Pham, V.-Q., Takahashi, K., and Naemura, T. (2011). Foreground-Background Segmentation using Iterated Distribution Matching. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2113–2120.

Pock, T., Schoenemann, T., Graber, G., Bischof, H., and Cremers, D. (2008). A convex formulation of continuous multi-label problems. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 792–805.

Pons, J.-P., Keriven, R., Faugeras, O., and Hermosillo, G. (2003). Variational stereovision and 3d scene flow estimation with statistical similarity measures. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 597–602.

Pons, J.-P., Keriven, R., and Faugeras, O. (2007). Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. *International Journal of Computer Vision (IJCV)*, **72**(2), 179–193.

Price, B. L., Morse, B., and Cohen, S. (2010). Geodesic Graph Cut for Interactive Image Segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3288–3295.

Qiu, W., Wang, X., Bai, X., Tu, Z., *et al.* (2014). Scale-space sift flow. In *2014 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1112–1119. IEEE.

Quiroga, J., Brox, T., Devernay, F., and Crowley, J. (2014). Dense semi-rigid scene flow estimation from rgbd images. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 567–582.

Rajan, D. and Chaudhuri, S. (2002). An MRF-Based Approach to Generation of Super-Resolution Images from Blurred Observations. *Journal of Mathematical Imaging and Vision*, **16**(1), 5–15.

Revaud, J., Weinzaepfel, P., Harchaoui, Z., and Schmid, C. (2015). EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Rhemann, C., Rother, C., Wang, J., Gelautz, M., Kohli, P., and Rott, P. (2009). A Perceptually Motivated Online Benchmark for Image Matting. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1826–1833. http://www.alphamatting.com.

Rhemann, C., Hosni, A., Bleyer, M., Rother, C., and Gelautz, M. (2011). Fast Cost-Volume Filtering for Visual Correspondence and Beyond. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3017–3024.

Rother, C., Kolmogorv, V., and Blake, A. (2004). GrabCut: Interactive Foreground Extraction Using Iterated Graph Cuts. *Proceedings of SIGGRAPH (ACM Transactions on Graphraphics)*, **23**, 309–314.

Rother, C., Kumar, S., Kolmogorov, V., and Blake, A. (2005). Digital Tapestry. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 589–596.

Rother, C., Kolmogorov, V., Minka, T., and Blake, A. (2006). Cosegmentation of Image Pairs by Histogram Matching–Incorporating a Global Constraint into MRFs. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 993–100.

Roy, S. and Cox, I. (1998). A Maximum-Flow Formulation of the N-camera Stereo Correspondence Problem. In *Proceedings of International Conference on Computer Vision (ICCV)*.

Rubinstein, M., Joulin, A., Kopf, J., and Liu, C. (2013). Unsupervised joint object discovery and segmentation in internet images. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1939–1946.

Rubio, J. C., Serrat, J., Lopez, A., and Paragios, N. (2012). Unsupervised co-segmentation through region matching. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 749–756.

Saito, M. (2016). *Study of Fast and Accurate Optimization Methods for Markov Random Fields*. Ph.D. thesis, Tohoku University.

Scharstein, D. and Szeliski, R. (2002). A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *International Journal of Computer Vision (IJCV)*, **47**(1-3), 7–42. (Middlebury Stereo Benchmark) http://vision.middlebury.edu/stereo/.

Schrijver, A. (2000). A Combinatorial Algorithm Minimizing Submodular Functions in Strongly Polynomial Time. *Journal of Combinational Theory Ser. B*, **80**(2), 346–355.

Shi, J. and Malik, J. (2000). Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **22**(8), 888–905.

Sibson, R. (1981). A Brief Description of Natural Neighbour Interpolation. In *Interpreting multivariate data*, chapter 2, pages 21–36. John Wiley & Sons.

Sivic, J., Russell, B. C., Efros, A., Zisserman, A., Freeman, W. T., *et al.* (2005). Discovering objects and their location in images. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 370–377.

Smith, B. M., Zhang, L., Brandt, J., Lin, Z., and Yang, J. (2013). Exemplar-based face parsing. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3484–3491.

Strandmark, P. and Kahl, F. (2010). Parallel and distributed graph cuts by dual decomposition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2085–2092.

Sun, D., Roth, S., and Black, M. J. (2010). Secrets of optical flow estimation and their principles. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2432–2439.

Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M., and Rother, C. (2008). A Comparative Study of Energy Minimization Methods for Markov Random Fields with Smoothness-Based Priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **30**(6), 1068–1080.

Tang, M., Ben Ayed, I., and Boykov, Y. (2014). Pseudo-Bound Optimization for Binary Energies. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 691–707.

Taniai, T., Pham, V.-Q., Takahashi, K., and Naemura, T. (2012). Image segmentation using dual distribution matching. In *Proceedings of British Machine Vision Conference (BMVC)*, pages 74.1–74.11.

Taniai, T., Matsushita, Y., and Naemura, T. (2014). Graph cut based continuous stereo matching using locally shared labels. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1613–1620.

Taniai, T., Matsushita, Y., and Naemura, T. (2015). Superdifferential cuts for binary energies. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2030–2038.

Taniai, T., Matsushita, Y., Sato, Y., and Naemura, T. (2016a). Continuous Stereo Matching Using Local Expansion Moves. arXiv:1603.08328, http://arxiv.org/abs/1603.08328.

Taniai, T., Sinha, S. N., and Naemura, T. (2016b). Joint recovery of dense correspondence and cosegmentation in two images. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Tao, H., Sawhney, H., and Kumar, R. (2001). A Global Matching Framework for Stereo Computation. In *Proceedings of International Conference on Computer Vision (ICCV)*, volume 1, pages 532–539.

Todorovic, S. and Ahuja, N. (2008). Region-based hierarchical image matching. *International Journal of Computer Vision (IJCV)*, **78**(1), 47–66.

Toivanen, P. J. (1996). New Geodesic Distance Transforms for Gray-scale Images. *Pattern Recogn. Lett.*, **17**(5), 437–450.

Tomasi, C. and Manduchi, R. (1998). Bilateral Filtering for Gray and Color Images. In *Proceedings of International Conference on Computer Vision (ICCV)*, pages 839–846.

Tuytelaars, T., Lampert, C. H., Blaschko, M. B., and Buntine, W. (2010). Unsupervised object discovery: A comparison. *International Journal of Computer Vision (IJCV)*, **88**(2), 284–302.

Valgaerts, L., Bruhn, A., Zimmer, H., Weickert, J., Stoll, C., and Theobalt, C. (2010). Joint estimation of motion, structure and geometry from stereo sequences. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 568–581.

Van den Bergh, M., Boix, X., Roig, G., de Capitani, B., and Van Gool, L. (2012). Seeds: Superpixels extracted via energy-driven sampling. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 13–26.

Vedula, S., Baker, S., Rander, P., Collins, R., and Kanade, T. (1999). Three-dimensional scene flow. In *Proceedings of International Conference on Computer Vision (ICCV)*, volume 2, pages 722–729.

Vedula, S., Baker, S., Rander, P., Collins, R. T., and Kanade, T. (2005). Three-dimensional scene flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **27**(3), 475–480.

Vicente, S., Kolmogorov, V., and Rother, C. (2009). Joint optimization of segmentation and appearance models. In *Proceedings of International Conference on Computer Vision (ICCV)*, pages 755–762.

Vicente, S., Kolmogorov, V., and Rother, C. (2010). Cosegmentation Revisited: Models and Optimization. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 465–479.

Vicente, S., Rother, C., and Kolmogorov, V. (2011). Object cosegmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2217–2224.

Vicente, S., Carreira, J., Agapito, L., and Batista, J. (2014). Reconstructing PASCAL VOC. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 41–48.

Vogel, C., Schindler, K., and Roth, S. (2011). 3d scene flow estimation with a rigid motion prior. In *Proceedings of International Conference on Computer Vision (ICCV)*, pages 1291–1298.

Vogel, C., Schindler, K., and Roth, S. (2013). Piecewise rigid scene flow. In *Proceedings of International Conference on Computer Vision (ICCV)*, pages 1377–1384.

Vogel, C., Roth, S., and Schindler, K. (2014). View-consistent 3d scene flow estimation over multiple frames. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 263–278.

Vogel, C., Schindler, K., and Roth, S. (2015). 3d scene flow estimation with a piecewise rigid scene model. *International Journal of Computer Vision (IJCV)*, **115**(1), 1–28.

Vu, H. H., Labatut, P., Pons, J. P., and Keriven, R. (2012). High Accuracy and Visibility-Consistent Dense Multiview Stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **34**(5), 889–901.

Wainwright, M., Jaakkola, T., and Willsky, A. (2002). Map estimation via agreement on (hyper)trees: Message-passing and linear programming approaches. *IEEE Trans. on Information Theory*, **51**, 3697–3717.

Wang, L. and Yang, R. (2011). Global Stereo Matching Leveraged by Pparse Ground Control Points. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3033–3040.

Wang, Y., Zhang, J., Liu, Z., Wu, Q., Chou, P. A., Zhang, Z., and Jia, Y. (2016). Handling occlusion and large displacement through improved rgb-d scene flow estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, **26**(7), 1265–1278.

Wang, Z.-F. and Zheng, Z.-G. (2008). A Region based Stereo Matching Algorithm using Cooperative Optimization. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wedel, A., Brox, T., Vaudrey, T., Rabe, C., Franke, U., and Cremers, D. (2011). Stereoscopic scene flow computation for 3d motion understanding. *International Journal of Computer Vision (IJCV)*, **95**(1), 29–51.

Wei, Y. and Quan, L. (2005). Asymmetrical Occlusion Handling Using Graph Cut for Multi-View Stereo. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 902–909.

Woodford, O., Torr, P. H. S., Reid, I., and Fitzgibbon, A. (2008). Global Stereo Reconstruction under Second Order Smoothness Priors. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

Woodford, O., Torr, P., Reid, I., and Fitzgibbon, A. (2009). Global Stereo Reconstruction under Second-Order Smoothness Priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **31**(12), 2115–2128.

Xu, L., Jia, J., and Matsushita, Y. (2012). Motion Detail Preserving Optical Flow Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **34**(9), 1744–1757.

Yang, H., Lin, W., and Lu, J. (2014). DAISY filter flow: A generalized discrete approach to dense correspondences. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3406–3413.

Yedidia, J. S., Freeman, W. T., and Weiss, Y. (2000). Generalized Belief Propagation. In *Advances in Neural Information Processing Systems (NIPS)*, volume 13, pages 689–695.

Yoon, K.-J. and Kweon, I.-S. (2005). Locally Adaptive Support-Weight Approach for Visual Correspondence Search. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 924–931.

Zabih, R. and Woodfill, J. (1994). Non-parametric Local Transforms for Computing Visual Correspondence. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 151–158.

Zbontar, J. and LeCun, Y. (2015). Computing the Stereo Matching Cost with a Convolutional Neural Network. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zbontar, J. and LeCun, Y. (2016). Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, **17**, 1–32.

Zhang, C., Shen, C., and Shen, T. (2015). Unsupervised feature learning for dense correspondences across scenes. *International Journal of Computer Vision (IJCV)*, pages 1–18.

Zhang, Q., Xu, L., and Jia, J. (2014). 100+ times faster weighted median filter (wmf). In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2830–2837.

Zhang, Y. and Kambhamettu, C. (2001). On 3d scene flow and structure estimation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages II–778.

Zhou, T., Lee, Y. J., Yu, S. X., and Efros, A. A. (2015). Flowweb: Joint image set alignment by weaving consistent, pixel-wise correspondences. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1191–1200.

List of Publications

Journal Papers

[J1] **Tatsunori Taniai**, Pham Viet-Quoc, Keita Takahashi, and Takeshi Naemura. (2013). *Foreground Background Image Segmentation using Dual Distribution Matching* (in Japanese), *IEICE Transactions on Information and Systems*, volume J96-D, number 8, pages 1764–1777. (Extended version of [C4])

International Conferences

[C1] **Tatsunori Taniai**, Sudipta N. Sinha, Yoichi Sato. (2016). "Joint Recovery of Dense Correspondence and Cosegmentation in Two Images". In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA.

[C2] **Tatsunori Taniai**, Yasuyuki Matsushita, Takeshi Naemura. (2015). "Superdifferential Cuts for Binary Energies". In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2030–2038, Boston, MA, USA.

[C3] **Tatsunori Taniai**, Yasuyuki Matsushita, Takeshi Naemura. (2015). "Graph Cut based Continuous Stereo Matching using Locally Shared Labels". In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1613–1620, Columbus, OH, USA.

[C4] **Tatsunori Taniai**, Pham Viet-Quoc, Keita Takahashi, and Takeshi Naemura. (2012). "Image Segmentation using Dual Distribution Matching". In *Proceedings of the 23rd British Machine Vision Conference (BMVC)*, pages 74.1–74.11, Surrey, UK.

Domestic Conferences

[D1] **Tatsunori Taniai** and Takeshi Naemura. (2013). In *Proceedings of the Meetings on Image Recognition and Understanding (MIRU)*.

[D2] **Tatsunori Taniai**, Pham Viet-Quoc, Keita Takahashi, and Takeshi Naemura. (2012). In *Proceedings of the Meetings on Image Recognition and Understanding (MIRU)*.

Technical Report

[R1] **Tatsunori Taniai**, Yasuyuki Matsushita, Yoichi Sato, Takeshi Naemura. (2016). "Continuous Stereo Matching using Local Expansion Moves". In *arXiv:1603.08328*, cs.CV. (Extended version of [C3])

Contents of Main Chapters

The contents provided in the main chapters of this thesis correspond to following published or unpublished papers.

- Chapter 3 corresponds to [C2].
- Chapter 4 corresponds to [R1]. (Main ideas from [C3] have been polished)
- Chapter 5 corresponds to [C1].
- Chapter 6 corresponds to an unpublished work.

Appendix

A

Supplementary Information for Chapter 3

In this supplementary chapter, we present proofs and implementation details associated with the proposed technique described in Chapter 3. The contents of this chapter are summarized below.

- Section A.1: a proof for Proposition 3.1 shown in Section 3.3.1.
- Section A.2: the definition of the geodesic distance used in Section 3.3.3.

A.1 Proof of Proposition 3.1

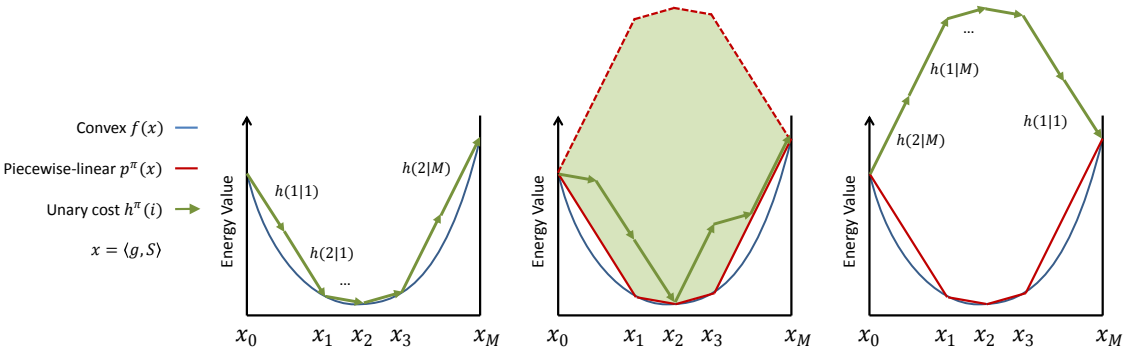
Given a grouped permutation $\pi = \{\pi(1), \pi(2), \cdots, \pi(M)\}$ and its corresponding bound $H^\pi(S|S^t)$ defined by Eqs. (3.14) and (3.15), we prove Proposition 3.1 by showing the tightness and boundness of Eq. (3.7) for $H^\pi(S|S^t)$, respectively, using the following two lemmas.

Lemma A.1. ***Tightness***. *It holds that* $H^\pi(S|S^t) = R(S)$ *for any* $S = S_j^\pi$.

Proof. Given S_j^π with any $j \in \{0, 1, \cdots, M\}$, it holds that

$$
\begin{aligned}
H^\pi(S_j^\pi|S^t) &= R(\emptyset) + \sum_{k=1}^{j} \sum_{i \in \pi(k)} g(i) \left[R(S_k^\pi) - R(S_{k-1}^\pi) \right] / \langle g, \pi(k) \rangle && \text{(A.1)} \\
&= R(\emptyset) + \sum_{k=1}^{j} \left[R(S_k^\pi) - R(S_{k-1}^\pi) \right] && \text{(A.2)} \\
&= R(S_j^\pi), && \text{(A.3)}
\end{aligned}
$$

which proves the statement. It also proves $H^\pi(S^t|S^t) = R(S^t)$, since there exists $S^t = S_j^\pi$ for some j. □



(a) Best approximation (b) Lower and upper envelops (c) Worst approximation

Figure A.1 Illustration of our bounds. A convex $f(x)$ and its piecewise-linear bounds $p^\pi(x)$ are visualized by blue and red solid-lines, respectively. The green arrows $h(k|j)$ show the unary costs $h^\pi(i)$ our bound $H^\pi(S|S^t)$ where i is the k-th element in the group $\pi(j)$. The green region in (b) shows the range in which the profiles of $H^\pi(S|S^t)$ lie. Note that these functions are shown in a continuous domain $x = \langle g, S \rangle$.

Lemma A.2. ***Boundness***. *It holds that* $H^\pi(S|S^t) \geq R(S) = f(\langle g, S \rangle)$ *for any* S.

Proof. We show the outline of the proof. Let $x_j = \left\langle g, S_j^\pi \right\rangle$, $(j = 0, 1, \cdots, M)$ be a sequence of scalar values, and let $p^\pi(x) : [x_0, x_M] \to \mathbb{R}$ be a *continues* piecewise-linear approximation function of $f(x)$. Here, $p^\pi(x)$ consists of M linear-pieces with $M+1$ breakpoints at $x = x_j$ and it tightly bounds convex $f(x)$; *i.e.*, it holds that $p^\pi(x_j) = f(x_j)$ and $p^\pi(x) \geq f(x)$ for any j and x, respectively. We visualize $p^\pi(x)$ as solid red lines in Figure A.1. Note that this figure slightly differs from Figure 3.4 in that here functions are shown in a continuous domain $x = \langle g, S \rangle$. Using this $p^\pi(x)$ we can show that

$$H^\pi(S|S^t) \geq p^\pi(\langle g, S \rangle) \geq f(\langle g, S \rangle) = R(S). \tag{A.4}$$

The first inequality is proved by the fact that $p^\pi(\langle g, S \rangle)$ gives a lower-envelop to $H^\pi(S|S^t)$. We illustrate this in Figure A.1. If S evolves along the chain sequence S_j^π $(j = 0, 1, \cdots, M)$, the profile of $H^\pi(S|S^t)$ is as tight as $p^\pi(\langle g, S \rangle)$, as shown in Figure A.1 (a). Generally, S evolves arbitrarily, which means that the green arrows in Figure A.1 (a) can be re-arranged arbitrarily as illustrated in Figure A.1 (b). Here, we can show that any profile of $H^\pi(S|S^t)$ cannot be lower than $p^\pi(x)$, if $f(x)$ is convex. Additionally, the profile of our bound in the worst case can be depicted as Figure A.1 (c). □

A.2 Geodesic Distance

In this section, we summarize the definition of the geodesic distance proposed in [Criminisi *et al.*, 2008]. Note that the distance $D(i|S, I)$ used in the main chapter refers to

$D_s^s(i|S,I)$ in Eq. (A.12) below. We now explain show this distance is defined.

Let $D(i|S,I):\Omega\to\mathbb{R}$ be an unsigned geodesic distance map from the boundary of segments S:

$$D(i|S,I)=\min_{\{p_j|j\in S\}} d_G(p_i,p_j), \tag{A.5}$$

where $d_G(p_i,p_j)$ is geodesic distance between two pixels $p_i,p_j\in\mathbb{Z}^2$:

$$d_G(p_i,p_j)=\min_{s\in\mathcal{P}}\sum_{k=2}^{|s|}\sqrt{\|p_{s(k)}-p_{s(k-1)}\|_2^2+\gamma^2\|I_{s(k)}-I_{s(k-1)}\|_2^2}. \tag{A.6}$$

Here, $\mathcal{P}$ is the set of all paths joining p_i and p_j. Note that when $\gamma=0$, the geodesic distance $d_G(p_i,p_j)$ becomes equivalent to the Euclidean distance $|p_i-p_j|$. Using the unsigned distance map, a signed geodesic distance map from S is defined as

$$D_s(i|S,I)=D(i|S,I)-D(i|\bar{S},I). \tag{A.7}$$

The use of $D_s(i|S,I)$ is more reasonable than the Euclidean distance for our purpose to create permutations σ. To illustrate this, see the two pixels a and b in Figure A.2 (a). When using the Euclidean distance shown in Figure A.2 (b), b has a shorter distance than a from S; hence, b is positioned before a in the permutation σ meaning that b is more likely to be within the true segments S^*. By contrast, when we use the geodesic distance $D_s(i|S,I)$ shown in Figure A.2 (c), it is likely that $D_s(a|S,I)<D_s(b|S,I)$, so a will come before b in σ.

However, as shown in Figure A.2 (d), this distance transform is sensitive to noise speckles such shown in Figure A.2 (e). For this issue, Criminisi *et al.* [2008] introduce *dilation* and *erosion* for segments S:

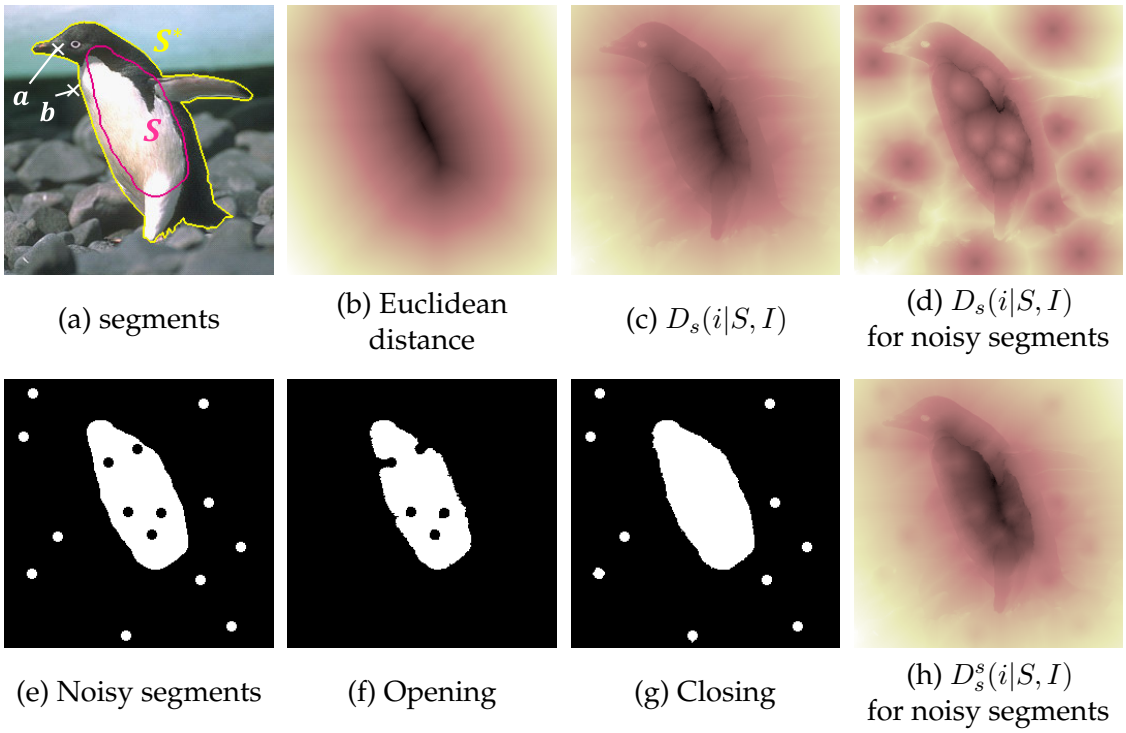
$$S_d = \{i\in\Omega\,|\,D_s(i|S,I)\le+\theta_d\}, \tag{A.8}$$
$$S_e = \{i\in\Omega\,|\,D_s(i|S,I)\le-\theta_e\}, \tag{A.9}$$

and further define *opening* and *closing* operations respectively as

$$S_o = \{i\in\Omega\,|\,D(i|S_e,I)\le+\theta_e\}, \tag{A.10}$$
$$S_c = \{i\in\Omega\,|\,D(i|\bar{S}_d,I)>+\theta_d\}. \tag{A.11}$$

The effects of opening and closing are visualized in Figs. A.2 (f) and (g). Using these operations, a robust geodesic distance is defined as

$$D_s^s(i|S,I)=\left[D(i|S_e,I)-\theta_e\right]-\left[D(i|\bar{S}_d,I)-\theta_d\right]. \tag{A.12}$$



(a) segments (b) Euclidean distance (c) $D_s(i|S,I)$ (d) $D_s(i|S,I)$ for noisy segments

(e) Noisy segments (f) Opening (g) Closing (h) $D^s_s(i|S,I)$ for noisy segments

Figure A.2 Illustration of geodesic distance [Criminisi *et al.*, 2008]. Given (a) an image and segments (pink), the use of (c) geodesic distance $D_s(i|S,I)$ is more reasonable than (b) the Euclidean distance. However, (d) $D_s(i|S,I)$ is sensitive to (e) noisy speckles in segments. With the presence of (f) opening and (g) closing effects, (h) $D^s_s(i|S,I)$ is robust to such speckles.

As shown in Figure A.2 (h), this distance is robust to noise speckles. The parameters θ_d and θ_e reflect the maximum sizes of speckles in foreground S and background $\bar{S}$, respectively. Throughout our paper, we use $\theta_d = \theta_e = 5$ and $\gamma = 10/255$. Note that if $\theta_e = \theta_d = \gamma = 0$, then $D^s_s(i|S,I)$ becomes the standard Euclidean distance.

B

Supplementary Information for Chapter 4

In this supplementary chapter, we provide a proof associated with the proposed technique described in Chapter 4.

B.1 Submodularity

Proof. For the completeness, we repeat the original proof given in [Olsson *et al.*, 2013] with our notations. Obviously, $\bar{\psi}_{pq}(\alpha, \alpha) = 0$. Therefore,

$$
\begin{aligned}
&\bar{\psi}_{pq}(\alpha,\alpha) + \bar{\psi}_{pq}(\beta,\gamma) \\
&\quad = \bar{\psi}_{pq}(\beta,\gamma) && \text{(B.1)}\\
&\quad = |d_p(\beta) - d_p(\gamma)| + |d_q(\beta) - d_q(\gamma)| && \text{(B.2)}\\
&\quad = |\,(d_p(\beta) - d_p(\alpha)) - (d_p(\gamma) - d_p(\alpha))\,| + |\,(d_q(\beta) - d_q(\alpha)) - (d_q(\gamma) - d_q(\alpha))\,| && \text{(B.3)}\\
&\quad \leq |d_p(\beta) - d_p(\alpha)| + |d_p(\gamma) - d_p(\alpha)| + |d_q(\beta) - d_q(\alpha)| + |d_q(\gamma) - d_q(\alpha)| && \text{(B.4)}\\
&\quad = \bar{\psi}_{pq}(\beta,\alpha) + \bar{\psi}_{pq}(\alpha,\gamma). && \text{(B.5)}
\end{aligned}
$$

Thus, $\bar{\psi}_{pq}(f_p, f_q)$ satisfies the submodularity of expansion moves. For its truncated function,

$$
\begin{aligned}
&\min\left(\bar{\psi}_{pq}(\alpha,\alpha),\tau\right) + \min\left(\bar{\psi}_{pq}(\beta,\gamma),\tau\right) \\
&\quad = \min\left(\bar{\psi}_{pq}(\beta,\gamma),\tau\right) && \text{(B.6)}\\
&\quad \leq \min\left(\bar{\psi}_{pq}(\beta,\alpha) + \bar{\psi}_{pq}(\alpha,\gamma),\tau\right) && \text{(B.7)}\\
&\quad \leq \min\left(\bar{\psi}_{pq}(\beta,\alpha),\tau\right) + \min\left(\bar{\psi}_{pq}(\alpha,\gamma),\tau\right). && \text{(B.8)}
\end{aligned}
$$

Therefore, the truncated function of $\bar{\psi}_{pq}(f_p, f_q)$ also satisfies the submodularity, and so does $\psi_{pq}(f_p, f_q)$. □

C

Supplementary Information for Chapter 5

In this supplementary chapter, we present derivations and proofs associated with the proposed technique described in Chapter 5. Some additional notes and implementation details are also provided. The contents of this chapter are summarized below.

- Section C.1: a note on the use of the continuous alpha map formulation.
- Section C.2: derivations of our approximation energy described in Section 5.4.1.
- Section C.3: details for initialization of color models described in Section 5.5.3.
- Section C.4: a proof of submodularity for our pairwise term.
- Section C.5: a note on our strategy of tuning parameters.

C.1 Continuous Alpha Map Formulation

Here, we explain why in our method, the per-pixel segmentation labels must be continuous alpha-matte values $\alpha \in [0, 1]$ rather than binary values $\{0, 1\}$. If α were binary, the flows $\mathbf{T}_i$ at nodes labeled background ($\alpha_i = 0$) would be under-constrained, because the flow data term $\mathcal{E}^i_{\text{flo}}$ in Eq. (5.3) at such nodes would always be a constant λ_{occ} regardless of the values of $\mathbf{T}_i$. This would be problematic, because if true foreground nodes are incorrectly labeled background in early stages of our inference process, it would be harder to recover their true flow labels in later iterations. To avoid this issue, we require α to be a continuous value that is larger than a small positive value (0.1 in our implementation). By doing this we will have meaningful flow labels $\mathbf{T}_i$ even at nodes labeled (incorrectly) background, because those flow labels still slightly affect matching energies of $\mathcal{E}^i_{\text{flo}}$.

C.2 Energy Approximation

We present the derivations of our approximation energy described in Section 5.4.1.

First, we derive the energy function $\mathcal{E}(f, G^{k+1})$ in the form of Eq. (5.15). In order to

simplify the energy formulation in Eq. (5.8), we denote energies involved in each layer as

$$\mathcal{E}^l_{\text{lay}}(f, G) = \mathcal{E}_{\text{mrf}}(f|L_l) + \mathcal{E}^l_{\text{reg}}(f|G) + \mathcal{E}^l_{\text{gra}}(V_l) \tag{C.1}$$

and rewrite the energy function $\mathcal{E}(f, G^{k+1})$ as the sum of layer energies

$$\begin{aligned}
\mathcal{E}(f, G^{k+1}) &= \sum_{l=0}^{k+1} \mathcal{E}^l_{\text{lay}}(f, G^{k+1}) && \text{(C.2)}\\
&= \sum_{l=0}^{k} \mathcal{E}^l_{\text{lay}}(f, G^k) + \mathcal{E}^{k+1}_{\text{lay}}(f, G^{k+1}) && \text{(C.3)}\\
&= \underbrace{\mathcal{E}(f, G^k)}_{\mathcal{E}(f|G^k)} + \underbrace{\mathcal{E}^{k+1}_{\text{lay}}(f, G^{k+1})}_{\mathcal{E}_{\text{top}}(f, L^{k+1})}. && \text{(C.4)}
\end{aligned}$$

Assuming that G^k is known from the previous iteration, we denote $\mathcal{E}(f, G^k)$ as $\mathcal{E}(f|G^k)$, and $\mathcal{E}^{k+1}_{\text{lay}}(f, G^{k+1})$ as $\mathcal{E}_{\text{top}}(f, L^{k+1})$ to obtain Eq. (5.15).

To approximate the above $\mathcal{E}(f, G^{k+1})$, we create a temporary graph $\hat{G}^{k+1}$ as an approximation of G^{k+1}, by duplicating the top layer of G^k as $L'_k = (V'_k, E'_k) \leftarrow (V_k, E_k)$. We further define a labeling $\hat{f}$ on this temporary graph $\hat{G}^{k+1}$. Since f and $\hat{f}$ are defined on the different graphs (G^{k+1} and $\hat{G}^{k+1}$) or different top layers (V_{k+1} and V'_k), we cannot simply assume $f = \hat{f}$. However, V'_k representing a superpixel segmentation is the finest form of any possible V_{k+1} due to the tree structure of G. Therefore, we can always define $\hat{f}$ so that f and $\hat{f}$ are equivalent $f \equiv \hat{f}$, *i.e.*, the pixelwise labeling included by both f and $\hat{f}$ are identical.

Using $\hat{f}$ and $\hat{G}^{k+1}$, our approximation function $\hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1})$ for $\mathcal{E}(f, G^{k+1})$ in the form of Eq. (5.17) is obtained by substituting $G^{k+1} \leftarrow \hat{G}^{k+1}$ and $f \leftarrow \hat{f}$ into $\mathcal{E}(f, G^{k+1})$.

$$\begin{aligned}
\hat{\mathcal{E}}(\hat{f}|\hat{G}^{k+1}) &= \mathcal{E}(\hat{f}, \hat{G}^{k+1}) && \text{(C.5)}\\
&= \underbrace{\mathcal{E}(\hat{f}, G^k)}_{\mathcal{E}(\hat{f}|G^k)} + \underbrace{\mathcal{E}^{k+1}_{\text{lay}}(\hat{f}, \hat{G}^{k+1})}_{\mathcal{A}(\hat{f})}. && \text{(C.6)}
\end{aligned}$$

Here, because G^{k+1} and $\hat{G}^{k+1}$ share the same structure except for the top layers, the energies $\mathcal{E}(\cdot|G^k)$ involved in the bottom hierarchy G^k are equivalent between Eqs. (C.4) and (C.6). To discuss how $\mathcal{A}(\hat{f})$ approximates $\mathcal{E}^{k+1}_{\text{top}}(f, L^{k+1})$, we write it as

$$\begin{aligned}
\mathcal{A}(\hat{f}) = \lambda_{\text{flo}} \sum_{i \in V'_k} \mathcal{E}^i_{\text{flo}}(\hat{f}_i) + \lambda_{\text{seg}} \sum_{i \in V'_k} \mathcal{E}^i_{\text{seg}}(\hat{f}_i) + \sum_{(s,t) \in E'_k} w_{st}\, \mathcal{E}^{st}_{\text{reg}}(\hat{f}_s, \hat{f}_t)\\
+ \sum_{(p,c) \in E^{\text{pc}'}_{k+1}} w_{pc}\, \mathcal{E}^{pc}_{\text{reg}}(f_p, f_c) + \mathcal{E}^{k+1}_{\text{gra}}(\hat{f}|V'_k). \qquad \text{(C.7)}
\end{aligned}$$

Here, the conversion for the three terms $\mathcal{E}^i_{\text{flo}}$, $\mathcal{E}^i_{\text{seg}}$ and $\mathcal{E}^{pc}_{\text{reg}}$ is exact, *i.e.*, those terms in $\mathcal{E}_{\text{top}}$ and corresponding terms in $\mathcal{A}$ yield the same energies as long as $f \equiv \hat{f}$. Next, we explain why these three conversions are exact and why the conversion for the two remaining terms are approximate.

Exact Conversion of Flow and Cosegmentation Data Terms

Exactness for the unary terms $\mathcal{E}^i_{\text{flo}}$ and $\mathcal{E}^i_{\text{seg}}$ in Eqs. (5.3) and (5.5) is shown in the same way. Notice that nodes in V_{k+1} are always obtained by merging nodes of V'_k, by following the rule of Eq. (5.19). Therefore, we can assume the domain Ω_i of each node $i \in V_{k+1}$ is the union of the domains of a connected component C_i of nodes in V'_k.

$$\Omega_i = \bigcup_{i' \in C_i} \Omega_{i'} \tag{C.8}$$

Furthermore, from $f \equiv \hat{f}$ it holds that $f_i = \hat{f}_{i'}$ for $i \in V_{k+1}$ and $i' \in C_i$. Using these properties, a unary term $\mathcal{E}^i$ in $\mathcal{E}_{\text{top}}$ can be exactly converted to the form in $\mathcal{A}$ as follows. (Changes from previous equations are colored by blue).

$$\sum_{i \in V_{k+1}} \mathcal{E}^i(f_i) = \sum_{i \in V_{k+1}} \sum_{\mathbf{p} \in \Omega_i} \phi_{\mathbf{p}}(f_i) \tag{C.9}$$

$$= \sum_{i \in V_{k+1}} \sum_{i' \in C_i} \sum_{\mathbf{p} \in \Omega_{i'}} \phi_{\mathbf{p}}(f_i) \tag{C.10}$$

$$= \sum_{i \in V_{k+1}} \sum_{i' \in C_i} \sum_{\mathbf{p} \in \Omega_{i'}} \phi_{\mathbf{p}}(\hat{f}_{i'}) \tag{C.11}$$

$$= \sum_{i' \in V'_k} \sum_{\mathbf{p} \in \Omega_{i'}} \phi_{\mathbf{p}}(\hat{f}_{i'}) \tag{C.12}$$

$$= \sum_{i \in V'_k} \mathcal{E}^i(\hat{f}_i) \tag{C.13}$$

Exact Conversion of Multi-layer Regularization Term

We perform a similar derivation for the multi-layer regularization term $\mathcal{E}^{pc}_{\text{reg}}$ in Eq. (5.10). From Figures 5.3 (c) and (d), we can see that each of the parent-child edges $(p, c) \in E^{\text{pc}}_{k+1}$ in the top layer of G^{k+1} has a corresponding edge $(p', c) \in E^{\text{pc}'}_{k+1}$ in $\hat{G}^{k+1}$ that has the same child c. Furthermore, for each one of those edges, $\mathbf{T}_p(\mathbf{p}) = \mathbf{T}_{p'}(\mathbf{p})$ and $\alpha_p = \alpha_{p'}$,

since $f \equiv \hat{f}$. Therefore, we can exactly convert $\mathcal{E}^{pc}_{\text{reg}}$ in $\mathcal{E}_{\text{top}}$ to the form in $\mathcal{A}$ as follows.

$$\sum_{(p,c)\in E^{\text{pc}}_{k+1}} w_{pc}\, \mathcal{E}^{pc}_{\text{reg}}(f_p, f_c) = \sum_{(p,c)\in E^{\text{pc}}_{k+1}} w_{pc} \left[\lambda_{\text{pc1}} \min\{\alpha_p, \alpha_c\}\psi^{pc}(\mathbf{c}_c) + \lambda_{\text{pc2}}|\alpha_p - \alpha_c|\right] \tag{C.14}$$

$$= \sum_{(p,c)\in E^{\text{pc}}_{k+1}} |\Omega_c| \left[\lambda_{\text{pc1}} \min\{\alpha_p, \alpha_c\}\min\{\|\mathbf{T}_p(\mathbf{c}_c) - \mathbf{T}_c(\mathbf{c}_c)\|_2, \tau_{\text{pc}}\} + \lambda_{\text{pc2}}|\alpha_p - \alpha_c|\right] \tag{C.15}$$

$$= \sum_{(p',c)\in E^{\text{pc}'}_{k+1}} |\Omega_c| \left[\lambda_{\text{pc1}} \min\{\alpha_{p'}, \alpha_c\} \min\{\|\mathbf{T}_{p'}(\mathbf{c}_c) - \mathbf{T}_c(\mathbf{c}_c)\|_2, \tau_{\text{pc}}\} + \lambda_{\text{pc2}}|\alpha_{p'} - \alpha_c|\right] \tag{C.16}$$

$$= \sum_{(p,c)\in E^{\text{pc}'}_{k+1}} w_{pc}\, \mathcal{E}^{pc}_{\text{reg}}(\hat{f}_p, \hat{f}_c) \tag{C.17}$$

Approximate Conversion of Spatial Regularization Term

For the spatial regularization term $\mathcal{E}^{st}_{\text{reg}}$ in Eq. (5.6), we split it into two parts.

$$\sum_{(s,t)\in E'_k} w_{st}\, \mathcal{E}^{st}_{\text{reg}}(\hat{f}_s, \hat{f}_t) = \lambda_{\text{st1}} \sum_{(s,t)\in E'_k} w_{st}\, \min\{\alpha_s, \alpha_t\} \sum_{\mathbf{p}\in B_{st}} \psi^{st}(\mathbf{p})/|B_{st}| + \lambda_{\text{st2}} \sum_{(s,t)\in E'_k} w_{st}\, |\alpha_s - \alpha_t|. \tag{C.18}$$

Here, the first and second parts evaluate flow and segmentation smoothness, respectively. We can show exact conversion for the segmentation smoothness part. To show this, we classify the edges of E'_k in $\hat{G}^{k+1}$ into two types: Type A) edges $(s', t') \in A$ across two different components $s' \in C_s$ and $t' \in C_t$. Type B) edges $(s'', t'') \in B$ within the same component $s'', t'' \in C_i$. Notice that $\mathcal{E}^{st}_{\text{reg}}(f_s, f_t) = 0$ for Type A edges, because $f_s = f_t$ holds in the same component. We now derive exact conversion for the segmentation smoothness part as follows.

$$\sum_{(s,t)\in E_{k+1}} w_{st}\, |\alpha_s - \alpha_t| = \sum_{(s,t)\in E_{k+1}} \left[\sum_{(s',t')\in A_{st}} w_{s't'}\right] |\alpha_s - \alpha_t| \tag{C.19}$$

$$= \sum_{(s,t)\in E_{k+1}} \sum_{(s',t')\in A_{st}} w_{s't'}\, |\alpha_{s'} - \alpha_{t'}| \tag{C.20}$$

$$= \sum_{(s',t')\in A} w_{s't'}\, |\alpha_{s'} - \alpha_{t'}| \tag{C.21}$$

$$= \sum_{(s',t')\in A} w_{s't'}\, |\alpha_{s'} - \alpha_{t'}| + \sum_{(s'',t'')\in B} w_{s''t''}\, |\alpha_{s''} - \alpha_{t''}| \tag{C.22}$$

$$= \sum_{(s,t)\in E'_k} w_{st}\, |\alpha_s - \alpha_t| \tag{C.23}$$

Here, $w_{st} = \sum w_{s't'}$ in Eq. (C.19) is from Eq. (5.14), but the definition of (s', t') can be equivalently replaced as Type A edges $(s', t') \in A_{st}$ where $s' \in C_s$ and $t' \in C_t$. Equation (C.20) is from $f \equiv \hat{f}$, where it holds that $\alpha_i = \alpha'_i$ for $i \in V_{k+1}$ and $i' \in C_i$.

In contrast, the conversion of the flow smoothness part in Eq. (C.18) is not always exact. However, the pixel locations $\mathbf{p}$ where the flow difference penalties $\psi^{st}(\mathbf{p})$ actually occur are the same in $\mathcal{E}_{\text{top}}$ and $\mathcal{A}$. Furthermore, the total costs of the flow smoothness part are equally bounded by $\sum_{(s,t)\in E_{k+1}} \lambda_{\text{st1}} w_{st} \tau_{\text{st}}$ in both $\mathcal{E}_{\text{top}}$ and $\mathcal{A}$. Thus, Eq. (C.18) is a good approximation for the spatial regularization term.

Approximate Conversion of Graph Validity Term

To derive an approximation $\mathcal{E}^{k+1}_{\text{gra}}(\hat{f}|V'_k)$ for the graph validity term $\mathcal{E}^{k+1}_{\text{gra}}(V_{k+1})$ in Eq. (5.11), we need to deal with two issues. 1) We need to convert variables from the node structure V_{k+1} in $\mathcal{E}_{\text{top}}$ to the labeling $\hat{f}$ on V'_k in $\mathcal{A}$. 2) The approximation function must be pairwise submodular energies for allowing graph cut based optimization.

For the first issue, we apply the variable conversion of Eq. (5.19) and regard V_{k+1} as a function $V_{k+1}(\hat{f})$ that represents a set of connected components C_i of nodes in V'_k assigned the same label. Thus, $\mathcal{E}^{k+1}_{\text{gra}}(V_{k+1})$ is converted to a function of $\hat{f}$ as follows.

$$\mathcal{E}^{k+1}_{\text{gra}}(V_{k+1}) = \lambda_{\text{nod}} \beta^{k+1} |V_{k+1}| - \lambda_{\text{col}} \sum_{i \in V_{k+1}} \sum_{\mathbf{p} \in \Omega_i} \ln P(\mathbf{I}_\mathbf{p} | \boldsymbol{\theta}^i) \tag{C.24}$$

$$= \lambda_{\text{nod}} \beta^{k+1} |V_{k+1}(\hat{f})| - \lambda_{\text{col}} \sum_{i \in V_{k+1}} \sum_{\mathbf{p} \in \Omega_i} \ln P(\mathbf{I}_\mathbf{p} | \boldsymbol{\theta}^{C_i}) \tag{C.25}$$

$$= \lambda_{\text{nod}} \beta^{k+1} |V_{k+1}(\hat{f})| - \lambda_{\text{col}} \sum_{i' \in V'_k} \sum_{\mathbf{p} \in \Omega_{i'}} \ln P(\mathbf{I}_\mathbf{p} | \boldsymbol{\theta}^{C_i}) \tag{C.26}$$

Here, $|V_{k+1}(\hat{f})|$ is the count of the components defined by the labeling $\hat{f}$, and $\boldsymbol{\theta}^{C_i}$ is the color distribution within the region of a component C_i that $i' \in V'_k$ belongs to. The fact that the computation of both $|V_{k+1}(\hat{f})|$ and $\boldsymbol{\theta}^{C_i}$ involves regional (higher-order) information of $\hat{f}$ raises the second issue.

To deal with the second issue of higher-order terms, we relax the connectivity of $|V_{k+1}(\hat{f})|$ and treat it as the count of unique labels $\hat{f}_i$ in $i \in V'_k$ without considering their spatial connections.

$$|V_{k+1}(\hat{f})| \simeq \sum_{L \in \{\text{all labels}\}} \delta_L(\hat{f}), \tag{C.27}$$

where $\delta_L(\hat{f}) = 1$ if $\exists i \in V'_k : \hat{f}_i = L$; otherwise $\delta_L(\hat{f}) = 0$. In this manner, $|V_{k+1}(\hat{f})|$ becomes *label costs* [Delong *et al.*, 2012] of $\hat{f}$, and the formulation of Eq. (C.26) is the same as that of multi-region segmentation of [Delong *et al.*, 2012]. In their model fitting approach, the label costs are optimized as pairwise submodular terms under alpha expansion moves with additional auxiliary variables. Our optimization approach using

local expansion moves allows the same strategy. Furthermore, the distribution $\boldsymbol{\theta}^{C_i}$ is treated as a label $\boldsymbol{\theta}_i$ given by $\hat{f}_i$, rather than a value computed from C_i. Thus, the likelihood terms in Eq. (C.26) are approximated as unary potentials as follows.

$$-\sum_{i' \in V'_k} \sum_{\mathbf{p} \in \Omega_{i'}} \ln P(\mathbf{I}_\mathbf{p} | \boldsymbol{\theta}^{C_i}) \simeq \sum_{i \in V'_k} \mathcal{E}^i_{\text{gra}}(\hat{f}_i) \tag{C.28}$$

where $\mathcal{E}^i_{\text{gra}}(\hat{f}_i)$ evaluates the given distribution label $\boldsymbol{\theta}_i$ included in $\hat{f}_i$ as

$$\mathcal{E}^i_{\text{gra}}(\hat{f}_i) = -\sum_{\mathbf{p} \in \Omega_i} \ln P(\mathbf{I}_\mathbf{p} | \boldsymbol{\theta}_i). \tag{C.29}$$

Note that the energy conversion is unnecessary for the graph terms in $\mathcal{E}(\hat{f}|G^k)$, because those terms are constant with the fixed G^k. Likewise, it is unnecessary in the whole process of the top-down labeling refinement phase.

Consequently, $\hat{f}$ becomes the following labeling on $\hat{G}^{k+1}$.

$$\hat{f}_i = \begin{cases} (\mathbf{T}_i, \alpha_i, \boldsymbol{\theta}_i) & \text{if } i \in V'_k \\ (\mathbf{T}_i, \alpha_i) & \text{if } i \in V_l \ (0 \leq l \leq k) \end{cases}. \tag{C.30}$$

The distribution label $\boldsymbol{\theta}_i$ of $i \in V'_k$ is initialized as the color distribution of the region Ω_i. Except for the cross-view proposer, the proposal generation for distribution labels is essentially the same as that of other labels $(\mathbf{T}, \alpha)$. The expansion and perturbation proposers simply copy the current label $\boldsymbol{\theta}_i$ of the target node i as a candidate. The average proposer generates candidates as the weighted sum of two distributions $w_i\boldsymbol{\theta}_i + w_j\boldsymbol{\theta}_j$. The cross-view proposer generates a candidate as the distribution within the region Ω_i of the target node i.

C.3 Initiailzation of Color Models

Here, we explain the implementation details of the initialization of color models $\{\boldsymbol{\theta}^F, \boldsymbol{\theta}^B\}$ omitted in Section 5.5.3.

Geodesic Distance

We first compute a geodesic distance map for each of the input images. At every pixel $\mathbf{p}$ we compute the shortest geodesic distance to any of the image boundary pixels $\mathbf{q} \in B$:

$$D(\mathbf{p}) = \min_{\mathbf{q} \in B} d(\mathbf{p}, \mathbf{q}), \tag{C.31}$$

where $d(\mathbf{p}, \mathbf{q})$ is the geodesic distance between two pixels $\mathbf{p}$ and $\mathbf{q}$ define as

$$d(\mathbf{p}, \mathbf{q}) = \min_{s \in \mathcal{P}} \sum_{k=1}^{|s|-1} \|\mathbf{I}(\mathbf{s}(k+1)) - \mathbf{I}(\mathbf{s}(k))\|_2. \tag{C.32}$$

Here, $\mathcal{P}$ is the set of all paths joining $\mathbf{p}$ and $\mathbf{q}$. The approximate computation of $D(\mathbf{p})$ is efficiently implemented using a linear-order algorithm of [Toivanen, 1996].

We further normalize the value range of the geodesic distance map by

$$\bar{D}(\mathbf{p}) = e^{-D(\mathbf{p})^2/\gamma}. \tag{C.33}$$

The parameter γ is given as $\gamma = \eta\sigma^2$ where $\sigma = E[\|\mathbf{I}(\mathbf{p}) - \mathbf{I}(\mathbf{q})\|_2]$ is computed as the expectation over all spatial neighbors $(\mathbf{p}, \mathbf{q})$, and η is set to 20 in our implementation. The values of $1 - \bar{D}(\mathbf{p})$ are visualized in the right part of Figure 5.4.

Seeds and Initial Mask Creation for GrabCut

Secondly, we compute seeds and initial masks of foreground and background as input to GrabCut [Rother *et al.*, 2004]. The seeds of foreground and background regions give constant unary likelihoods. The initial masks are used to initialize the color distributions used in GrabCut. We compute these regions using the ratio values and the geodesic distance as follows.

As explained in Section 5.5.3, we have three ratio values $\{r_1, r_2, r_3\}$ at each pixel computed from the three levels of the image pyramid. For each level, we normalize the ratio values to be in the range of $[0, 1]$ using the minimum and maximum ratio values. After the layerwise normalization, we integrate the three ratio values to obtain a single value as $r = r_1r_2r_3 + (1 - r_1)r_2r_3 + r_1(1 - r_2)r_3 + r_1r_2(1 - r_3)$. We then create the foreground / background seeds and foreground / background masks as regions where $r < 0.05$, $r > 0.95$, $r < 0.70$ and $r > 0.85$, respectively. The regions of foreground seed and mask are further reduced if the geodesic distance is $\bar{D}(\mathbf{p}) > 0.5$.

In our implementation, the color likelihood terms of GrabCut are implemented by 64^3 bins of RGB color histograms with a weight coefficient of 1. The pixels in the foreground/background seeds are assigned a constant likelihood value of 10. Using the geodesic distance in Eq. (C.33), we also add background likelihood values of $10\bar{D}(\mathbf{p})$. For efficiency, we use the superpixel nodes of V_1 during this step and reuse them again in our main algorithm. Finally, we obtain estimated color models $\{\boldsymbol{\theta}^F, \boldsymbol{\theta}^B\}$ of an image after a few iterations of GrabCut. We perform this computation for each of the two images.

C.4 Submodularity

As discussed in [Taniai *et al.*, 2016a, 2014] (Chapter 4) the submodularity condition of local expansion move energies in Eq. (5.20) is the same as that of conventional alpha expansion moves [Boykov *et al.*, 2001]. To prove that our energy is submodular under expansion moves, we need to show that our pairwise regularization terms $\mathcal{E}^{sp}_{\text{reg}}$ and $\mathcal{E}^{pc}_{\text{reg}}$ in Eqs. (5.6) and (5.10) are submodular. To simplify discussions, we rewrite these terms as a pairwise function, as follows.

$$\phi(\mathbf{x}, \mathbf{y}) = \min\{x, y\}\psi(\mathbf{x}, \mathbf{y}) + \lambda|x - y|. \tag{C.34}$$

Here, $\lambda \geq 0$ is a scalar weight, a bold $\mathbf{x}$ denotes a label vector of $(\mathbf{T}, \alpha)$ while a non-bold x denotes its scalar alpha label $\alpha \in [0, 1]$. The two terms $\mathcal{E}^{sp}_{\text{reg}}$ and $\mathcal{E}^{pc}_{\text{reg}}$ can be expressed in this form by properly defining $\psi(\mathbf{x}, \mathbf{y})$. Using this notation we prove the following two lemmas.

Lemma C.1. *If $\psi(,)$ satisfies the following three conditions for any $\mathbf{x}, \mathbf{y}, \mathbf{z}$*

$$0 \leq \psi(\mathbf{x}, \mathbf{y}) \leq \tau, \tag{C.35}$$

$$\psi(\mathbf{x}, \mathbf{x}) = 0, \tag{C.36}$$

$$\psi(\mathbf{x}, \mathbf{y}) + \psi(\mathbf{z}, \mathbf{z}) \leq \psi(\mathbf{x}, \mathbf{z}) + \psi(\mathbf{z}, \mathbf{y}), \tag{C.37}$$

and if

$$\tau \leq 2\lambda, \tag{C.38}$$

then $\phi(\mathbf{x}, \mathbf{y})$ is submodular under expansion moves, i.e.*, it satisfies the following submodularity condition of expansion moves [Boykov* et al.*, 2001; Kolmogorov and Zabin, 2004]:*

$$\phi(\mathbf{x}, \mathbf{y}) + \phi(\mathbf{z}, \mathbf{z}) \leq \phi(\mathbf{x}, \mathbf{z}) + \phi(\mathbf{z}, \mathbf{y}). \tag{C.39}$$

Proof. Notice that $\phi(\mathbf{z}, \mathbf{z}) = 0$. Using this and assuming $x \geq y$ without loss of generality, Eq. (C.39) can be expressed as

$$\min\{x, z\}\psi(\mathbf{x}, \mathbf{z}) + \lambda|x - z| + \min\{z, y\}\psi(\mathbf{z}, \mathbf{y}) + \lambda|z - y| - y\psi(\mathbf{x}, \mathbf{y}) - \lambda(x - y) \geq 0. \tag{C.40}$$

The proof for this inequity is divided into the following three cases depending on z.

Case 1 where $x \geq y \geq z \geq 0$. We show in this case that

Eq. (C.40, left)

$$= z\psi(\mathbf{x},\mathbf{z}) + \lambda(x-z) + z\psi(\mathbf{z},\mathbf{y}) + \lambda(y-z) - y\psi(\mathbf{x},\mathbf{y}) - \lambda(x-y) \tag{C.41}$$

$$= z\Big[\psi(\mathbf{x},\mathbf{z}) + \psi(\mathbf{z},\mathbf{y})\Big] - y\psi(\mathbf{x},\mathbf{y}) + 2\lambda(y-z) \tag{C.42}$$

$$\geq z\psi(\mathbf{x},\mathbf{y}) - y\psi(\mathbf{x},\mathbf{y}) + 2\lambda(y-z) \tag{C.43}$$

$$= (y-z)\Big[2\lambda - \psi(\mathbf{x},\mathbf{y})\Big] \tag{C.44}$$

$$\geq (y-z)\Big[2\lambda - \tau\Big] \tag{C.45}$$

$$\geq 0. \tag{C.46}$$

Case 2 where $x \geq z \geq y \geq 0$. Similarly, we show that

Eq. (C.40, left)

$$= z\psi(\mathbf{x},\mathbf{z}) + \lambda(x-z) + y\psi(\mathbf{z},\mathbf{y}) + \lambda(z-y) - y\psi(\mathbf{x},\mathbf{y}) - \lambda(x-y) \tag{C.47}$$

$$= z\psi(\mathbf{x},\mathbf{z}) + y\psi(\mathbf{z},\mathbf{y}) - y\psi(\mathbf{x},\mathbf{y}) \tag{C.48}$$

$$\geq y\Big[\psi(\mathbf{x},\mathbf{z}) + \psi(\mathbf{z},\mathbf{y}) - \psi(\mathbf{x},\mathbf{y})\Big] \tag{C.49}$$

$$\geq 0. \tag{C.50}$$

Case 3 where $z \geq x \geq y \geq 0$. Finally, we show that

Eq. (C.40, left)

$$= x\psi(\mathbf{x},\mathbf{z}) + \lambda(z-x) + y\psi(\mathbf{z},\mathbf{y}) + \lambda(z-y) - y\psi(\mathbf{x},\mathbf{y}) - \lambda(x-y) \tag{C.51}$$

$$= x\psi(\mathbf{x},\mathbf{z}) + y\psi(\mathbf{z},\mathbf{y}) - y\psi(\mathbf{x},\mathbf{y}) + 2\lambda(z-x) \tag{C.52}$$

$$\geq y\Big[\psi(\mathbf{x},\mathbf{z}) + \psi(\mathbf{z},\mathbf{y}) - \psi(\mathbf{x},\mathbf{y})\Big] + 2\lambda(z-x) \tag{C.53}$$

$$\geq 0. \tag{C.54}$$

□

Lemma C.2. *If $\psi(\mathbf{x},\mathbf{y})$ is given by a form of the truncated Euclidean distance as*

$$\psi(\mathbf{x},\mathbf{y}) = \min\{\|\mathbf{x}-\mathbf{y}\|_2, \tau\}, \tag{C.55}$$

then $\psi(\mathbf{x},\mathbf{y})$ satisfies the aforementioned three conditions of Eqs. (C.35) – (C.37).

Proof. The first and second conditions are obvious. We can also show the third condition

as follows.

$$\begin{aligned}
\psi(\mathbf{x},\mathbf{y}) + \psi(\mathbf{z},\mathbf{z}) &= \psi(\mathbf{x},\mathbf{y}) && \text{(C.56)}\\
&= \min\{\|\mathbf{x}-\mathbf{y}\|_2, \tau\} && \text{(C.57)}\\
&= \min\{\|(\mathbf{x}-\mathbf{z})-(\mathbf{y}-\mathbf{z})\|_2, \tau\} && \text{(C.58)}\\
&\leq \min\{\|\mathbf{x}-\mathbf{z}\|_2 + \|\mathbf{y}-\mathbf{z}\|_2, \tau\} && \text{(C.59)}\\
&\leq \min\{\|\mathbf{x}-\mathbf{z}\|_2, \tau\} + \min\{\|\mathbf{y}-\mathbf{z}\|_2, \tau\} && \text{(C.60)}\\
&= \psi(\mathbf{x},\mathbf{z}) + \psi(\mathbf{z},\mathbf{y}) && \text{(C.61)}
\end{aligned}$$

□

The above two lemmas directly derive the submodularity for the parent-children term $\mathcal{E}^{pc}_{\text{reg}}$ using substitutions $\lambda = \lambda_{\text{pc2}}$ and $\tau = \lambda_{\text{pc1}}\tau_{\text{pc}}$. By slightly modifying Eq. (C.55) for the spatial term $\mathcal{E}^{st}_{\text{reg}}$, it can also be shown to be submodular where $\lambda = \lambda_{\text{st2}}$ and $\tau = \lambda_{\text{st1}}\tau_{\text{st}}$.

C.5 Tuning Hyper Parameters

We explain our strategy of tuning parameters. Since the graph term is independent of the labeling, we start with a simple energy function consisting of only the graph term. We set $\lambda_{\text{col}} = 1$ and tune λ_{nod} so that $|V_1| \simeq 2|V_2|$ in the obtained graph. We then use the single layer model and tune parameters of the flow (λ_{flo}, τ_D) and segmentation (λ_{seg}) data terms and spatial smoothness term (λ_{st1}, λ_{st2}, τ_{st}). While checking segmentation quality, we tune λ_{seg} at around λ_{col} and λ_{st2} at around 50 (the default setting in GrabCut [Rother *et al.*, 2004]). The remaining flow-related parameters are tuned by checking flow quality. We finally use the hierarchical model and tune the parameters (λ_{pc1}, λ_{pc2}, τ_{pc}) of the multi-layer regularization.

D

Supplementary Information for Chapter 6

In this supplementary chapter, we present implementation details associated with the proposed techniques described in Chapter 6. Some additional notes and experiments are also provided. The contents of this chapter are summarized below.

- Section D.1: implementation details of SGM stereo used in Section 6.4.1.
- Section D.2: implementation details of SGM flow used in Section 6.4.5.
- Section D.3: implementation details of flow map refinement used in Section 6.4.5.
- Section D.4: implementation details of segmentation prior used in Section 6.4.7.
- Section D.5: a note on our strategy of tuning parameters.

D.1 SGM Stereo

In the binocular and epipolar stereo stages (Section 6.4.1 and 6.4.3), we solve stereo matching problems using the semi-global matching (SGM) algorithm [Hirschmuller, 2008]. We implement SGM as described in Section 2.3.4. Specifically, stereo matching is cast as a discrete labeling problem, where we estimate the disparity map $\mathcal{D}_{\mathbf{p}} = \mathcal{D}(\mathbf{p}) : \Omega \to D$ (where $D = \{D_{\text{min}}, \cdots, D_{\text{max}}\}$ is the disparity range) that minimizes the following 2D Markov random field (MRF) based energy function.

$$E_{\text{stereo}}(\mathcal{D}) = \sum_{\mathbf{p}\in\Omega} C_{\mathbf{p}}(\mathcal{D}_{\mathbf{p}}) + \sum_{(\mathbf{p},\mathbf{q})\in\mathcal{N}} cV_{\mathbf{pq}}(\mathcal{D}_{\mathbf{p}}, \mathcal{D}_{\mathbf{q}}). \tag{D.1}$$

In our implementation of SGM, the unary data term $C_{\mathbf{p}}$ in Eq. (D.1) is defined using truncated normalized cross-correlation in Eq. (6.5) in the main chapter. The smoothness penalties P_1 and P_2 of $V_{\mathbf{pq}}$ (see Eq. (2.26) for the definition of $V_{\mathbf{pq}}$) are defined as follows.

$$P_1 = \lambda_{\text{sgm}}/|\mathbf{p} - \mathbf{q}| \tag{D.2}$$

$$P_2 = P_1 \left(\beta + \gamma w^{\text{col}}_{\mathbf{pq}}\right) \tag{D.3}$$

Here, $w^{\text{col}}_{\mathbf{pq}}$ is the color edge-based weight used in Eq. (6.15) and we use parameters of $(\lambda_{\text{sgm}}, \beta, \gamma) = (200/255, 2, 2)$. The disparity range is fixed as $\{D_{\text{min}}, \cdots, D_{\text{max}}\} = \{0, \cdots, 255\}$ for the original image size of KITTI (since we downscale the images by a factor of 0.65, the disparity range is also downscaled accordingly). We also set the confidence threshold τ_{u} for the uncertainty map $\mathcal{U}$ to 2000 by visually inspecting $\mathcal{U}(\mathbf{p})$.

D.2 SGM Flow

We have extended the SGM algorithm for our optical flow problem in Section 6.4.5. Here, we estimate the flow map $\mathcal{F}_{\mathbf{p}} = \mathcal{F}(\mathbf{p}) : \Omega \rightarrow R$ (where $R = ([u_{\text{min}}, u_{\text{max}}] \times [v_{\text{min}}, v_{\text{max}}])$ is the 2D flow range) by minimizing the following 2D MRF energy.

$$E_{\text{flow}}(\mathcal{F}) = \sum_{\mathbf{p} \in \Omega} C'_{\mathbf{p}}(\mathcal{F}_{\mathbf{p}}) + \sum_{(\mathbf{p},\mathbf{q}) \in \mathcal{N}} cV'_{\mathbf{pq}}(\mathcal{F}_{\mathbf{p}}, \mathcal{F}_{\mathbf{q}}). \tag{D.4}$$

Similarly to SGM stereo, we use the NCC-based matching cost of Eq. (6.5) for the data term $C'_{\mathbf{p}}(\mathcal{F}_{\mathbf{p}})$ to evaluate matching photo-consistencies between I^0_t and I^0_{t+1}. We also define the smoothness term as

$$V'_{\mathbf{pq}}(\mathcal{F}_{\mathbf{p}}, \mathcal{F}_{\mathbf{q}}) = \begin{cases} 0 & \text{if } \mathcal{F}_{\mathbf{p}} = \mathcal{F}_{\mathbf{q}} \\ P_1 & \text{if } 0 < \|\mathcal{F}_{\mathbf{p}} - \mathcal{F}_{\mathbf{q}}\| \leq \sqrt{2} \\ P_2 & \text{otherwise} \end{cases} . \tag{D.5}$$

Since we use integer flow labels, the second condition in Eq. (D.5) is equivalent to saying that the components of the 2D vectors $\mathcal{F}_{\mathbf{q}} = (u_{\mathbf{q}}, v_{\mathbf{q}})$ and $\mathcal{F}_{\mathbf{p}} = (u_{\mathbf{p}}, v_{\mathbf{p}})$ can at-most differ by 1. We use the same smoothness penalties $\{P_1, P_2\}$ and the parameter settings with SGM stereo.

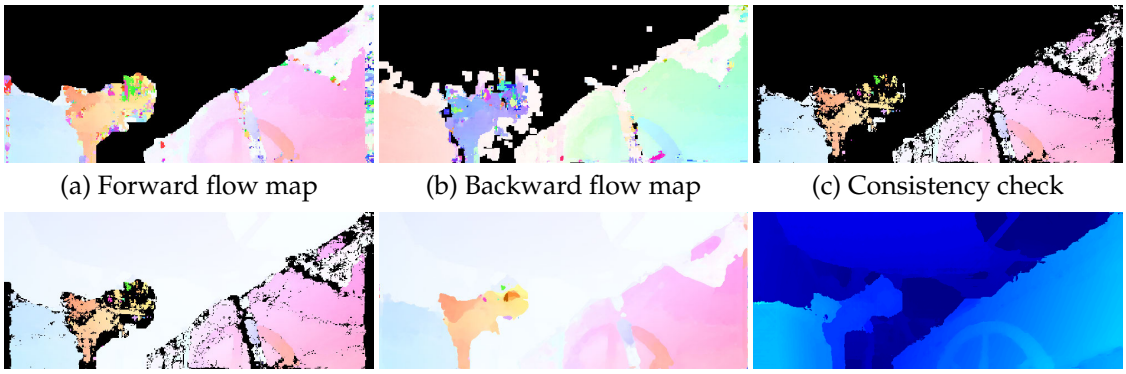
The optimization of Eq. (D.4) is essentially the same with SGM stereo, but the implementation of updating scan-line costs in Eq. (2.27) was extended to handle the new definition of the pairwise term $V'_{\mathbf{pq}}$. Therefore, Eq. (2.30) is modified using a flow label $\mathbf{u} = (u, v) \in R$ as follows.

$$L_{\mathbf{r}}(\mathbf{p}, \mathbf{u}) = C_{\mathbf{p}}(\mathbf{u}) + \min\{\bar{L}_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, \mathbf{u}), \bar{L}_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, \mathbf{u} + \Delta_{\pm 1}) + P_1, P_2\} \tag{D.6}$$

Here, $(\mathbf{u} + \Delta_{\pm 1})$ is enumeration of 8 labels neighboring to $\mathbf{u}$ in the 2D flow space.

D.3 Refinement of Flow Maps

In the optical flow stage of Section 6.4.5, we refine flow maps using consistency check and weighted median filtering. Similar schemes are commonly employed in stereo and optical flow methods such as [Zhang *et al.*, 2014; Hosni *et al.*, 2013; Bleyer *et al.*, 2011].



(a) Forward flow map (b) Backward flow map (c) Consistency check

(d) Background filling (e) Weighted median filtering (f) Disparity map as guidance image

Figure D.1 Process of flow map refinement.

Below we explain these steps.

We first estimate the forward flow map $\mathcal{F}^0$ (from I_t^0 to I_{t+1}^0) by SGM for only the foreground pixels of the initial segmentation $\tilde{\mathcal{S}}$ such as shown in Figure D.1 (a). Then, using this flow $\mathcal{F}^0$ and the mask $\tilde{\mathcal{S}}$, we compute a mask in the next image I_{t+1}^0 and estimate the backward flow map $\mathcal{F}^1$ (from I_{t+1}^0 to I_t^0) for those foreground pixels. This produces a flow map such as shown in Figure D.1 (b). We filter out outliers in $\mathcal{F}^0$ using bi-directional consistency check between $\mathcal{F}^0$ and $\mathcal{F}^1$ to obtain a flow map with holes (Figure D.1 (c)), whose background is further filled by the rigid flow $\mathcal{F}_{\text{rig}}$ (see Figure D.1 (d)). Finally, weighted median filtering is applied for the hole pixels followed by median filtering for all foreground pixels to obtain the non-rigid flow estimate such as shown in Figure D.1 (e).

At the final weighted median filtering step, the filter kernel $\omega_{\mathbf{pq}}^{\text{geo}} = e^{-d_{\mathbf{pq}}/\kappa_{\text{geo}}}$ is computed using geodesic distance $d_{\mathbf{pq}}$ on the disparity map $\mathcal{D}$ (Figure D.1 (f)) as the guidance image. For this, we define the distance between two adjacent pixels as

$$\text{dist}(\mathbf{p}_1, \mathbf{p}_2) = |\mathcal{D}(\mathbf{p}_1) - \mathcal{D}(\mathbf{p}_2)| + \|\mathbf{p}_1 - \mathbf{p}_2\|/100. \tag{D.7}$$

The geodesic distance $d_{\mathbf{pq}}$ is then computed for the pixels in the filter window $\mathbf{q} \in W_{\mathbf{p}}$ as the cumulative shortest-path distance from $\mathbf{q}$ to the center pixel $\mathbf{p}$. This is efficiently computed using an approximate algorithm [Toivanen, 1996]. We use the filter window $W_{\mathbf{p}}$ of 31×31 size and $\kappa_{\text{geo}} = 2$. The subsequent (constant-weight) median filtering further reduces outliers [Sun *et al.*, 2010], for which we use the window of 5×5 size.

D.4 Segmentation Ground Prior

The segmentation ground prior term mentioned in Section 6.4.7 is computed as follows. First, we detect the ground plane from the disparity map $\mathcal{D}(\mathbf{p})$. We use RANSAC to

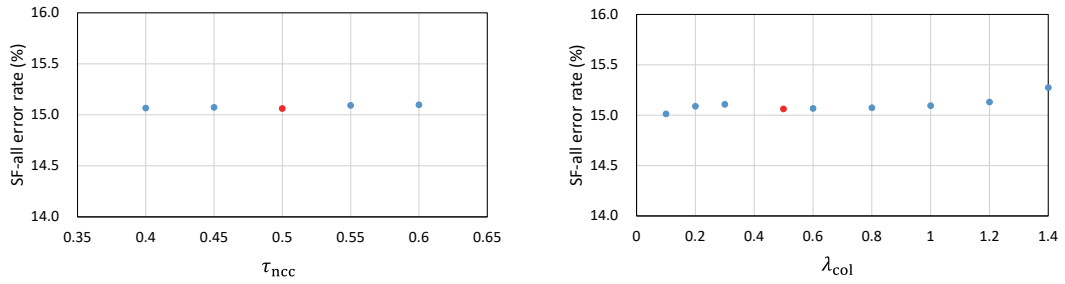


Figure D.2 Profiles of scene flow accuracies w.r.t. parameters τ_{ncc} (left) and λ_{col} (right). The error rates are evaluated on 200 training sequences from KITTI. The scores with the default parameter settings are colored by red.

fit a disparity plane $[d = au + bv + c]$ defined on the 2D image coordinates. Here, we assume that the cameras in the stereo rig are upright. Therefore, during RANSAC we choose disparity planes whose b is positive and high and $|a|$ is relatively small. Then, we compute the disparity residuals between $\mathcal{D}$ and the ground plane as $r_{\mathbf{p}} = |\mathcal{D}_{\mathbf{p}} - (a\mathbf{p}_u + b\mathbf{p}_v + c)|$, where (a, b, c) are the obtained plane parameters. Our ground prior as a cue of background is then defined as follows.

$$C_{\mathbf{p}}^{\text{gro}} = \lambda_{\text{gro}} \Big(\min(r_{\mathbf{p}}, \tau_{\text{gro}})/\tau_{\text{gro}} - 1 \Big) \tag{D.8}$$

When $r_{\mathbf{p}} = 0$, $C_{\mathbf{p}}^{\text{gro}}$ strongly favors background, and when $r_{\mathbf{p}}$ increases to τ_{gro}, it becomes 0. The thresholding value $\tau_{\mathbf{p}}$ is set to $0.01 \times D_{\text{max}}$. We use $\lambda_{\text{gro}} = 10$.

D.5 Parameter Settings

In this section, we explain our strategy of tuning parameters and also show effects of some parameters. Most of the parameters can be easily interpreted and tuned, and our method is fairly insensitive to parameter settings.

For example, the effects of the threshold τ_{u} for the uncertainty map $\mathcal{U}$ (Section 6.4.1), the threshold τ_w for the patch-variance weight $\omega_{\mathbf{p}}^{\text{var}}$ (Section 6.4.4), and κ_3 of the image edge-based weight $\omega_{\mathbf{pq}}^{\text{str}}$ (Section 6.4.4) can be easily analyzed by direct visualization as shown in Figure 6.3 (b), Figures 6.4 (b) and (e).

The parameters of SGM (discussed in Section D.1) can be tuned independently from the whole algorithm.

For the weights $(\lambda_{\text{ncc}}, \lambda_{\text{flo}}, \lambda_{\text{col}}, \lambda_{\text{potts}})$ in Section 6.4.4, we first tuned $(\lambda_{\text{ncc}}, \lambda_{\text{flo}}, \lambda_{\text{potts}})$ on a small number of sequences. Since the ranges of the NCC appearance term (Eq. (6.11)) and flow term (Eq. (6.12)) are limited to $[-1, 1]$, they are easy to interpret. Then, we tuned λ_{col} of the color term (Eq. (6.14)). Here, $\lambda_{\text{potts}}/\lambda_{\text{col}}$ is known to be usually around 10 - 60 from previous work [Rother *et al.*, 2004; Boykov and Jolly, 2001].

Even though we fine-tuned τ_{ncc} and λ_{col} for Sintel, they are insensitive on KITTI image sequences. We show the effects of these two parameters for KITTI training sequences in Figure D.2. The threshold τ_{ncc} for NCC-based matching costs was adjusted for Sintel because its synthesized images have lesser image noise compared to real images of KITTI. Also, the weight λ_{col} was adjusted for Sintel, to increase the weight on the prior color term (Section 6.4.7). For Sintel sequences, sometimes moving objects stop moving on a few frames and become stationary momentarily. In such cases, increasing λ_{col} improves the temporal coherence of the motion segmentation results. In the future we will improve the scheme for online learning of the prior color models, which will improve temporal consistency of motion segmentation and also will make λ_{col} more insensitive to settings.