University of California
Los Angeles

Cluster Sampling and its Applications to Segmentation, Stereo and Motion

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

Adrian Gheorghe Barbu

2005

The dissertation of Adrian Gheorghe Barbu is approved.

Adnan Darwiche

Stefano Soatto

Alan L. Yuille

Song-Chun Zhu, Committee Chair

University of California, Los Angeles

2005

To my wife, Diana Barbu

Table of Contents

List of Figures

Acknowledgments

I would like to thank my advisor Song-Chun Zhu for his vision, continued enthusiasm and support. I would like to thank Alan Yuille, Yingnian Wu, Zhuowen Tu and Feng Han for helpful discussions.

Vita

1995	B.S. (Mathematics), University of Bucharest.
1995–2000	Teaching Assistant, Mathematics Department, OSU.
2000	Ph.D. (Mathematics), OSU.
2000–2002	Research Assistant, Computer Science Department, OSU. Advisor Professor Song-Chun Zhu.
2002–2005	Research Assistant, Computer Science Department, UCLA. Advisor Professor Song-Chun Zhu.

Publications

Generalizing Swendsen-Wang to sampling arbitrary posterior probabilities. IEEE Transactions on Pattern Analysis and Machine Intelligence, August 2005. A reversible stochastic algorithm capable of sampling arbitrary probability functions of labeled graphs is presented. The algorithm is applied in image segmentation and stereo. It is shown to be at least 400 times faster than Gibbs sampler. and shown to obtain smaller energies than Belief Propagation in 6 minutes and a final energy within 1% of that of Graph Cuts.

Cluster sampling and its applications in image analysis. Submitted to J. of Computational and Graphical Statistics. The Swendsen-Wang algorithm is general-

ized to sample arbitrary posterior probabilities on graphs, and multi-grid and multi-level versions are developed. Applications in image segmentation and motion segmentation combined with image segmentation in a hierarchical representation are presented.

Interactions of incorporated carbon atoms and dimer vacancies on the Si(001) surface. J. of Eng. Materials and Technology, 2005. The self-organization into lines of C atoms and DVs on the Si surface is reproduced by Monte Carlo simulation, concurring with recent experimental findings.

Incorporating Visual Knowledge Representation in Stereo Reconstruction. Submitted to Int. Conf. on Comp. Vis. 2005.We represent mid-level knowledge of a scene using a dictionary of 3D primitives for edges and junctions, and use this representation to perform stereo reconstruction for images containing textureless regions, tree branches and free-form surfaces.

Multigrid and Multi-level Swendsen-Wang Cuts for Hierarchic Graph Partition. IEEE Conf on Comp. Vis. and Patt. Rec. 2004. We extend our Swendsen-Wang Cuts algorithm to multi-grid and multi-level versions, while still maintaining reversibility. We show results for hierarchical image-motion segmentation.

Motion Estimation by Swendsen-Wang Cuts. IEEE Conf on Comp. Vis. and Patt. Rec. 2004.We show how to use Swendsen-Wang Cuts for motion estimation and we use simple clustering and stochastic boundary evolution to obtain motion segmentation results. We model occluded pixels and that allows us to obtain accurate segmentation results.

On the relationship between image and motion segmentation. SCVMA workshop, Eur. Conf. on Comp. Vis. 2004. We use a generative model to track image regions through multiple frames, using the Swendsen-Wang Cuts algorithm.

Graph Partition By Swendsen-Wang Cuts. Int. Conf. on Comp. Vis. 2003. We describe a reversible stochastic algorithm working on arbitrary energy functions and we show it is at least 400 times faster than Gibbs sampler. We present applications for image segmentation and curve grouping.

On the range of non-vanishing p-torsion cohomology for GLn(Fp). J. Algebra, August 2004.

On a conjecture of Ash. J. Algebra, May 2002.

The ring generated by the elements of degree 2 in $H^*(U_n(\mathbb{F}_1), \mathbb{Z})$. J. Algebra, March 2001.

On the cohomology $GL_n(\mathbb{F}_1)$*, with* $\mathbb{F}_1$*, coefficients.* PhD thesis, OSU 2000.

Abstract of the Dissertation

Cluster Sampling and its Applications to Segmentation, Stereo and Motion

by

Adrian Gheorghe Barbu

Doctor of Philosophy in Computer Science

University of California, Los Angeles, 2005

Professor Song-Chun Zhu, Chair

Many computer vision problems can be formulated as graph partition problems that minimize energy functions. Generally applicable algorithms like the Gibbs sampler can perform the minimization task, but they are very slow to converge, especially since the graphs in vision tasks are large ($10^5 - 10^6$ nodes). On the other hand, computationally effective algorithms like Graph Cuts and Belief Propagation are specialized to particular forms of energy functions, and they cannot be applied for complex statistical models using generative models and high-order priors. In this thesis, a new stochastic algorithm capable of sampling arbitrary energy functions defined on graph partitions is presented. To increase efficiency, the algorithm uses the image information to make informed jumps in the search space. The image information is given in the form of edge weights and represents an empirical probability that the nodes connected by the edge belong to the same object. At each step, the algorithm creates clusters of nodes by turning on/off the edges randomly according to their weights, and changes the label of all nodes in one cluster (connected component) in a single move. Each move is accepted or rejected according to an acceptance probability given by a simple and explicit

equation. The algorithm is applied to 4 important problems in computer vision: image segmentation, perceptual organization, stereo matching and motion segmentation. To address different computational or representational issues, multi-grid, multi-level and multi-cue variants of the algorithm are presented. In image segmentation, the algorithm's performance is compared to the Gibbs sampler, while in stereo matching, it is compared to Graph Cuts and Belief Propagation.

CHAPTER 1

Introduction

Markov chain Monte Carlo (MCMC) methods are general computing tools for simulation, inference, and optimization in many fields. The essence of MCMC is to design a Markov chain whose transition kernel $\mathcal{K}$ has an unique invariant (target) probability $\pi(\mathbf{X})$ pre-defined to a task. For example, $\pi(\mathbf{X})$ could be a Bayesian posterior probability or a probability governing the states of a physical system. In this thesis, we are interested in Markov chains with finite states $\mathbf{X}$ defined on graphs $\mathbf{G} =< V, E >$ where $\mathbf{X} = (x_1, x_2, ..., x_n)$ represents the states of the vertices $V = \{v_1, v_2, ..., v_n\}$. Such problems are often referred as graph coloring (or labeling) and have very broad applications in physics, biology, and computer science.

Graph partition is a fundamental problem in computer vision, which addresses a wide array of vision tasks:

- The line drawing interpretation consists of labeling the line segments of a line drawing with a set of labels corresponding to a dictionary of possible line types (occlusion edge, convex/concave interior edge), constrained by a dictionary of junction types.

- In image segmentation, the pixels of an image need to be partitioned into regions corresponding to the different intensity patterns existent in the image.

- In motion segmentation, the pixels of a pair (set of images) need to be partitioned into regions based on a coherent motion criterion.
- Stereo matching can also be regarded as a graph partition problem in which the pixel lattice needs to be partitioned based on the common depth (disparity) of the pixels.
- other examples include curve grouping, object recognition, etc.

Although the method presented in this thesis is applicable to general graphs and target probabilities, we shall focus on a number of examples in image analysis, such as image segmentation and motion analysis. For such applications, the graph $\mathbf{G}$ is very large with $O(10^4) - O(10^6)$ vertices which are image elements like pixels, and $\mathbf{G}$ has sparse neighbor connections, i.e. constant $O(1)$ connectivity. That is, the connectivity of a vertex does not grow with the number of vertices. The state x_i is the color (or label) for image segmentation or discretized motion velocity in motion analysis. The target probabilities $\pi(\mathbf{X})$ are usually Markov random fields whose conditional probabilities can be computed locally.

Historically, graph labeling algorithms were born with the Waltz algorithm [58] for line drawing interpretation. The Waltz algorithm filters out impossible combinations of labels by checking pairs of compatible labels at neighboring junctions. It is however not capable of providing a solution in ambiguous cases (e.g. the Necker cube illusion).

The Waltz algorithm was later refined into the relaxation labeling [43], which can address a more general set of problems by attaching probabilities to labels. The algorithm will propagate these probabilities between neighboring nodes until all probabilities stabilize. It is not guaranteed that the probabilities will provide a unique solution.

The relaxation labeling was generalized in two directions.

A stochastic generalization is the Gibbs sampler [17], a generally applicable MCMC algorithm which is proved to sample from a given probability distribution after a burn-in period.

A deterministic generalization of relaxation labeling is Belief Propagation [38, 39, 49, 62], which infers marginal probabilities at the nodes of the graph by exchanging of messages. Initially, Belief propagation was designed on trees and was proved to obtain the true marginal probabilities at the nodes. It was later generalized to graphs with loops to perform approximate inference on probabilities based on pairwise cliques.

Other approaches to graph partition (labeling) are the graph spectral analysis methods [61] such as Normalized Cut [48] that minimize a discriminative energy function defined in terms of the graph edge weights. The energy function modeled by the Normalized Cut is capable of generating clean results, even though the intensity regions can sometimes be broken into a small number of pieces. However, the Normalized Cut method has difficulties in modeling the diverse phenomena existent in images, for example it will break long curve-like regions.

Another popular method [44] maps a simple energy function into a min-cut problem, which is solved using the max-flow algorithm. One such method is the Graph Cuts [9], which applies the max-flow algorithm repeatedly, for different pairs of labels, until convergence. Applications of Graph Cuts include dense stereo matching and image inpainting.

Following the MCMC direction, we see generalizations of the Gibbs sampler to multigrid [20], parameter expansion [34], parallel tempering [18]. The slow mixing of such methods is attributed to the strong coupling between the variables in the graph.

One well celebrated algorithm that addresses the coupling between the variables is the Swendsen-Wang [50] method designed for simulating the Ising/Potts models [29, 40] in statistical physics. It is often called the cluster sampling method. At each iteration, the SW method forms clusters of vertices as connected components by sampling Bernoulli variables defined on the edges. Then, it flips the color of all vertices in one or all clusters simultaneously.

The SW method is found to mix rapidly under certain conditions. For example, Cooper and Frieze [10] show that SW has polynomial mixing time for graphs with $O(1)$ connectivity, such as the Ising/Potts models even at critical temperature. Gore and Sinclair [21] showed that SW has exponential mixing time when $\mathbf{G}$ is a complete graph. Huber [28] designed bounding chains for the SW method so as to diagnose exact sampling in some temperature range of the Potts model (see Fig. 2.2). The SW convergence can also be analyzed with a maximal correlation technique ([35], chapter 7). Despite its success, the power of the SW method and its analyses is very limited for two reasons:

1. It is only applicable to the Ising/Potts models and cannot be applied to arbitrary probabilities on general graphs.

2. It does not make use of the data information in designing the probability for the binary variables on edges, and thus in clustering the vertices. Because of this, the SW algorithm slows down drastically in the presence of "external fields" (i.e. data energy terms).

In this thesis, we present a general cluster sampling algorithm which generalizes the SW-method in the following aspects:

1. Designed from the Metropolis-Hastings perspective, it is applicable to general probabilities on graphs.

2. The edge probabilities, which represent compatibilities of adjacent vertices, are designed using discriminative probabilities computed from the input data. Therefore the clustering step is informed by the data (external field) and leads to significant speedup as observed experimentally.

3. In a modified version, it can be viewed as a generalized Gibbs sampler which samples the color of a cluster according to a conditional probability (like the Gibbs sampler) weighted by a product of a small number of edge probabilities. This can also be viewed as a generalized hit-and-run method.

4. It is extended to multi-grid and multi-level graphs for hierarchic graph labeling.

In our experiments on image analysis (segmentation, curve grouping, motion and stereo), the algorithm is at least $O(10^2)$ times faster than the single-site Gibbs sampler (see Figs.3.3, 3.4, 3.5). When working on pixels, the algorithm is actually incomparably faster than the Gibbs sampler, since the Gibbs sampler cannot obtain the same energy level in any reasonable time. Compared to Graph Cuts [9] and Belief Propagation [49, 53] on a Potts model, our algorithm outperforms Belief Propagation and comes within 1% of the energy level of Graph Cuts, as shown in Figure 6.2.

In the literature, there are two famous interpretations of the SW-method which lead to various analyses or generalizations. Both view the SW method as a data augmentation method [51].

1. The first is the Random Cluster Model (RCM) by Edwards and Sokal [11]. It augments the target probability $\pi(\mathbf{X})$ with a new set of binary variables $\mathbf{U}$ on the edges. The joint probability $p_{\text{ES}}(\mathbf{X}, \mathbf{U})$ has a marginal probability $\pi(\mathbf{X})$ and two conditional probabilities $p_{\text{ES}}(\mathbf{X}|\mathbf{U})$ and $p_{\text{ES}}(\mathbf{U}|\mathbf{X})$ which are

easy to sample. In this model, the clustering and labeling are decoupled completely. It leads to the design of bounding chain [28] for exact sampling.

2. The second is the slice sampling and decoupling method by Higdon [26]. It augments $\pi(\mathbf{X})$ by a set of continuous variables W as the "bond strength" on edges to increase the connectivity of the space, and then sample the labels under the constraints of these variables (i.e. slice sampling). Higdon applied this method to some image analysis examples and also studied a partial decoupling method which has a coupling factor controlled by the data.

In this thesis, we take a third route by interpreting SW as a Metropolis-Hastings step with auxiliary variables for proposing the moves. Each step is a reversible jump [23] and observes the detailed balance equations. The key observation is that the proposal probability ratio can be calculated neatly as a ratio of products of probabilities on a small number of edges on the border of the cluster.

The thesis is organized as follows. In Chapter 2 we present the theoretical aspects of our generalized cluster sampling method, also named Swendsen-Wang Cuts. Chapter 3 shows the first experiment on image segmentation and a performance comparison with the single site Gibbs sampler. In Chapter 4 we present experiments in perceptual organization, namely curve grouping. Then we proceed to the multi-grid and multi-level cluster sampling in Chapter 5 and show applications of these methods on hierarchical image-motion segmentation. The thesis is concluded in Chapter 6 with experiments on stereo matching, including a comparison of our method with Graph Cuts and Belief Propagation.

CHAPTER 2

The Swendsen-Wang Cuts Algorithm

In this chapter, we present the Swendsen-Wang Cuts algorithm, which is the central part of this thesis.

We start with a review of the original Swendsen-Wang algorithm [50] and the Potts model [40] on which it was originally developed and give two interpretations in Section (2.1). Then we derive a generalized method by the Metropolis-Hastings perspective in Section (2.2). A number of variant methods are presented in Section (2.3), including the cluster Gibbs sampler and the multiple flipping scheme.

2.1 Background: The Swendsen-Wang Algorithm and its interpretations

In this section, we review the Potts model, the original Swendsen-Wang method and its two interpretations. The review is made concrete enough so that important results can be followed.

2.1.1 Swendsen-Wang on Potts model

Let $\mathbf{G} =< V, E >$ be an adjacency graph, such as a lattice with 4 nearest neighbor connections. Each vertex $v_i \in V$ is assigned a state variable x_i taking values from a finite number of labels (or colors), $x_i \in \{1, 2, ..., \mathrm{L}\}$. The total number of labels

L is pre-defined, and the Potts model for a homogeneous Markov field is defined as,

$$\pi_{\text{PTS}}(\mathbf{X}) = \frac{1}{Z} \exp\{\beta \sum_{<i,j>\in E} \mathbf{1}(x_i = x_j)\}. \tag{2.1}$$

$\mathbf{1}(x_i = x_j)$ is a Boolean function. It is equal to 1 if its condition $x_i = x_j$ is observed, and is 0 otherwise. In more general cases, $\beta = \beta(v_i, v_j)$ may be position dependent. Most computer vision applications use $\beta > 0$, also named the ferro-magnetic model, prefering similar colors for neighboring vertices. The case $\beta < 0$ is the anti-ferromagnetic model. The Potts models and its extensions are used as *a priori* probabilities in many Bayesian inference tasks.

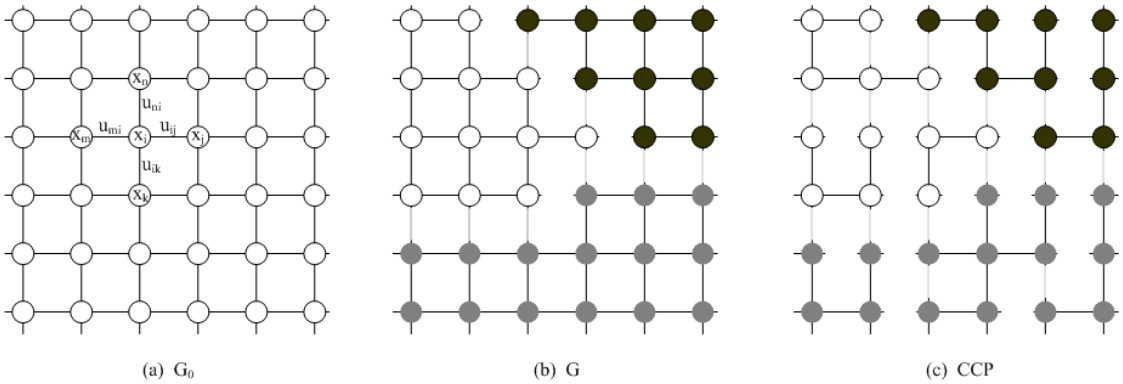


Figure 2.1: Illustating the SW method. (a) An adjacency graph $\mathbf{G}$ with each edge $< i, j >$ augmented with a binary variable $\mu_{ij} \in \{1, 0\}$. (b) A labeling of the Graph $\mathbf{G}$, where the edges connecting vertices of different colors are removed. (c). A number of connected components obtained by turning off some edges in (b) probabilistically.

As Fig.2.1.(a) illustrates, the SW method introduces a set of auxiliary variables on the edges.

$$\mathbf{U} = \{\mu_{ij} : \ \mu_{ij} \in \{0, 1\}, \ \forall < i, j >\in E\}. \tag{2.2}$$

The edge $< i, j >$ is disconnected (or turned off) if and only if $\mu_{ij} = 0$. μ_{ij} follows

a Bernoulli distribution conditional on x_i, x_j.

$$\mu_{ij}|(x_i, x_j) \sim \text{Bernoulli}(\rho \mathbf{1}(x_i = x_j)), \quad \rho = 1 - e^{-\beta}. \tag{2.3}$$

$\mu_{ij} = 1$ with probability ρ if $x_i = x_j$, and $\mu_{ij} = 1$ with probability 0 if $x_i \neq x_j$. The SW method iterates two steps.

1. The clustering step. Given the current state $\mathbf{X}$, it samples the auxiliary variables in $\mathbf{U}$ according to eqn. (2.3). It first turns off all edges $< i, j >$ deterministically if $x_i \neq x_j$, as Fig.2.1.(b) shows. Then it turns off the remain edges with probability ρ. The edge $< i, j >$ is divided into the "on" and "off" sets respectively depending on whether $\mu_{ij} = 1$ or 0.

$$E = E_{\text{on}}(\mathbf{U}) \cup E_{\text{off}}(\mathbf{U}). \tag{2.4}$$

The edges in $E_{\text{on}}(\mathbf{U})$ induce a number of connected components shown in Fig. 2.1.(c). We denote all these connected components by,

$$\text{CP}(\mathbf{U}) = \{\text{cp}_i : \ i = 1, 2, ..., K, \ \text{with} \cup i = 1^K \text{cp}_i = V\}. \tag{2.5}$$

Vertices in each connected component cp_i have the same color.

2. The flipping step. It selects one connected component cp $\in$ CP at random and assigns a common color y to all vertices in cp. y follows a uniform probability,

$$x_i = y \ \forall v_i \in cp, \quad y \sim \text{unif}\{1, 2, ..., \text{L}\}. \tag{2.6}$$

In this step, one may choose to repeat the random color flipping for all the connected components in CP($\mathbf{U}$) independently, as they are decoupled given the edges in $E_{\text{on}}(\mathbf{U})$.

In one modified version by Wolff [60], one may choose a vertex $v \in V$ and grow a connected component following the Bernoulli trials on edges around v. This saves some computation in the clustering step, and bigger components have a higher chance to be selected.

2.1.2 SW Interpretation 1: data augmentation and RCM

The SW method described above is far from what was presented in the original paper [50]. Instead our description follows the interpretation by Edward and Sokal [11], who augmented the Potts model to a joint probability on both $\mathbf{X}$ and $\mathbf{U}$,

$$\begin{aligned} p_{\mathrm{ES}}(\mathbf{X},\mathbf{U}) &= \frac{1}{Z}\prod_{<i,j>\in E}[(1-\rho)\mathbf{1}(\mu_{ij}=0)+\rho\mathbf{1}(\mu_{ij}=1)\cdot\mathbf{1}(x_i=x_j)] \quad (2.7)\\ &= \frac{1}{Z}[(1-\rho)^{|E_{\mathrm{off}}(\mathbf{U})|}\cdot\rho^{E_{\mathrm{on}}(\mathbf{U})}]\cdot\prod_{<i,j>\in E_{\mathrm{on}}(\mathbf{U})}\mathbf{1}(x_i=x_j). \quad (2.8)\end{aligned}$$

The second factor $\prod_{<i,j>\in E_{\mathrm{on}}(\mathbf{U})}\mathbf{1}(x_i=x_j)$ is in fact a hard constraint on $\mathbf{X}$ and $\mathbf{U}$. Let the space of $\mathbf{X}$ be

$$\Omega=\{1,2,...,\mathrm{L}\}^{|V|}. \quad (2.9)$$

Under this hard constraint, the labeling $\mathbf{X}$ is reduced to a subspace $\Omega_{\mathrm{CP}(\mathbf{U})}$ where each connected component must have the same label,

$$\prod_{<i,j>\in E_{\mathrm{on}}(\mathbf{U})}\mathbf{1}(x_i=x_j)=\mathbf{1}(\mathbf{X}\in\Omega_{\mathrm{CP}(\mathbf{U})}). \quad (2.10)$$

The joint probability $p_{\mathrm{ES}}(\mathbf{X},\mathbf{U})$ observes two nice properties, and both are easy to verify.

Proposition 1. *The Potts model is a marginal probability of the joint probability,*

$$\sum_{\mathbf{U}}p_{\mathrm{ES}}(\mathbf{X},\mathbf{U})=\pi_{\mathrm{PTS}}(\mathbf{X}). \quad (2.11)$$

The other marginal probability is the random cluster model π_{RCM}*,*

$$\sum_{\mathbf{X}}p_{\mathrm{ES}}(\mathbf{X},\mathbf{U})=\pi_{\mathrm{RCM}}(\mathbf{U})=\frac{1}{Z}(1-\rho)^{|E_{\mathrm{off}}(\mathbf{U})|}\cdot\rho^{E_{\mathrm{on}}(\mathbf{U})}\mathrm{L}^{|\mathrm{CP}(\mathbf{U})|}. \quad (2.12)$$

Proposition 2. *The conditional probabilities of* $p_{\mathrm{ES}}(\mathbf{X}, \mathbf{U})$ *are*

$$p_{\mathrm{ES}}(\mathbf{U}|\mathbf{X}) = \prod_{<i,j>\in E} p(\mu_{ij}|x_i, x_j), \quad \text{with } p(\mu_{ij}|x_i, x_j) = \text{Bernoulli}(\rho \mathbf{1}(x_i = x_j)),$$
$$p_{\mathrm{ES}}(\mathbf{X}|\mathbf{U}) = \text{unif}[\Omega_{\mathrm{CP}(\mathbf{U})}] = \begin{cases} (\frac{1}{\mathrm{L}})^{|\mathrm{CP}(\mathbf{U})|} & \text{for } \mathbf{X} \in \Omega_{\mathrm{CP}(\mathbf{U})} \\ 0 & \text{otherwise} \end{cases}. \tag{2.13}$$

Therefore the two SW steps can be viewed as sampling the two conditional probabilities.

1. Clustering step: $\mathbf{U} \sim p_{\mathrm{ES}}(\mathbf{U}|\mathbf{X})$, i.e. $\mu_{ij}|(x_i, x_j) \sim \text{Bernoulli}(\rho \mathbf{1}(x_i = x_j))$.
2. Flipping step: $\mathbf{X} \sim p_{\mathrm{ES}}(\mathbf{U}|\mathbf{X})$, i.e. $\mathbf{X}(\mathrm{cp}_i) \sim \text{Unif}\{1, 2, ..., \mathrm{L}\}$, $\forall \mathrm{cp}_i \in \mathrm{CP}(\mathbf{U})$.

As $(\mathbf{X}, \mathbf{U}) \sim p_{\mathrm{ES}}(\mathbf{X}, \mathbf{U})$, discarding the auxiliary variables $\mathbf{U}$, we have $\mathbf{X}$ following the marginal of $p_{\mathrm{ES}}(\mathbf{X}, \mathbf{U})$. The goal is achieved,

$$\mathbf{X} \sim \pi_{\mathrm{PTS}}(\mathbf{X}). \tag{2.14}$$

The beauty of this data augmentation method [51] is that the labeling of the connected components are completely decoupled (independent) given the auxiliary variables. As $\rho = 1 - e^{-\beta}$, it tends to choose smaller clusters if the temperature ($T \propto \frac{1}{\beta}$) in the Potts model is high, and in low temperature it chooses large clusters and it can overcome the coupling problem of the single site Gibbs sampler.

2.1.3 Some theoretical results

Let the Markov chain have kernel $\mathcal{K}$ and initial state $\mathbf{X}_o$, in t steps the Markov chain state follows probability $p_t = \delta(\mathbf{X} - \mathbf{X}_o)\mathcal{K}^t$ where $\delta(\mathbf{X} - \mathbf{X}_o)$ (for $\delta(\mathbf{X} - \mathbf{X}_o) =$

1 for $\mathbf{X} = \mathbf{X}_o$ and 0 otherwise) is the initial probability. The convergence of the Markov chain is often measured by the total variation

$$||p_t - \pi||_{\text{TV}} = \frac{1}{2} \sum_{\mathbf{X}} |p_t(\mathbf{X}) - \pi(\mathbf{X})|. \tag{2.15}$$

The mixing time of the Markov chain is defined by

$$\tau = \max_{\mathbf{X}_o} \min\{t : \ ||p_t - \pi||_{\text{TV}} \leq \epsilon\}. \tag{2.16}$$

τ is a function of ϵ and the graph complexity $M = |\mathbf{G}|$ in terms of the number of vertices and connectivity. The Markov chain is said to mix rapidly if $\tau(M)$ is polynomial or logarithmic.

Empirically, the SW method is found to mix rapidly. Recently some analytic results on its performance have surfaced. Cooper and Frieze [10] proved using a path coupling technique that SW mixes rapidly on sparsely connected graphs.

Theorem 1. (Cooper and Frieze 1999) *Let* $n = |V|$ *and* Δ *be the maximum number of edges at any single vertex, and* L *the number of colors in Potts model. If* $\mathbf{G}$ *is a tree, then the SW mixing time is* $O(n)$ *for any* β *and* L. *If* $\Delta = O(1)$, *then there exists* $\rho_o = \rho(\Delta)$ *such that if* $\rho \leq \rho_o$ *(i.e. higher than a certain temperature), then SW has polynomial mixing time for all* L.

A negative case was constructed by Gore and Jerrum [22] on complete graphs.

Theorem 2. (Gore and Jerrum 1997) *If* $\mathbf{G}$ *is a complete graph and* $\text{L} > 2$, *then for* $\beta = \frac{2(\text{L}-1)\ln(\text{L}-1)}{n(\text{L}-2)}$, *the SW does not mix rapidly.*

In the image analysis applications, our graph often observes the Copper-Frieze condition and the graph is far from being complete.

Most recently an exact sampling technique was developed for SW on Potts model by Huber [28] for very high or very low temperatures. It designs a bounding

chain which assumes that each vertex $v_i \in V$ has a set of colors S_i initialized with the full set $|S_i| = \mathrm{L}$, $\forall i$. The Bernoulli probability for the auxiliary variables μ_{ij} is changed to

$$\mathbf{U}^{\mathrm{bd}} = \{\mu_{ij}^{\mathrm{bd}} : \mu_{ij}^{\mathrm{bd}} \in \{0,1\},\ \mu_{ij} \sim \mathrm{Bernoulli}(\rho \mathbf{1}(S_i \cap S_j \neq \emptyset))\}. \tag{2.17}$$

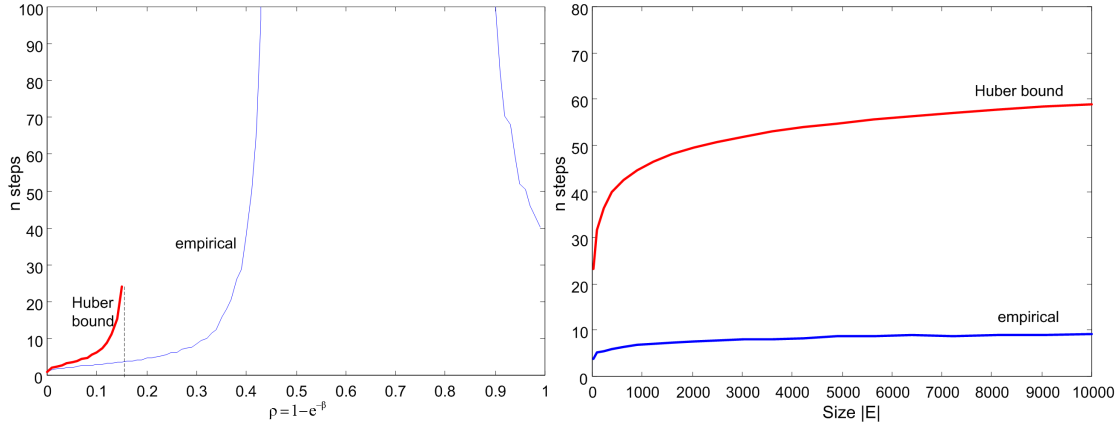
Thus $\mathbf{U}^{\mathrm{bd}}$ has more edges than $\mathbf{U}$ in the original SW chain, i.e. $\mathbf{U} \subset \mathbf{U}^{\mathrm{bd}}$. When $\mathbf{U}^{\mathrm{bd}}$ collapses to $\mathbf{U}$, then all SW chains starting with arbitrary initial states have collapsed into the current single chain. Thus it must have converged (exact sampling). The collapsing step is called the "coupling time".

Theorem 3. (Huber 2002) *Let $n = |V|$ and $m = |E|$, at high temperature, $\rho < \frac{1}{2(\Delta-1)}$, the bounding chain couples completely by time $O(\ln(2m))$ with probability at least $1/2$. At lower temperature, $\rho \geq 1 - \frac{1}{m\mathrm{L}}$, then the coupling time is $O((m\mathrm{L})^2)$ with probability at least $1/2$.*

In fact the Huber bound is not very tight, as one may expect. Fig. 2.2(a) plots the results on a 5×5 lattice with torus boundary condition on the Ising model for the empirical coupling time against $\rho = 1 - e^{-\beta}$. The coupling time is large near the critical temperature (didn't plot). The Huber bound for the high temperature starts with $\rho_o = 0.16$ and is plotted by the short curve. The bound for the low temperature starts with $\rho_o > 0.99$ which is not visible. Fig.2.2.(b) plots the coupling time at $\rho = 0.15$ against the graph size $m = |E|$ and the Huber bound.

Despite the encouraging successes discussed above, the SW method is limited in two aspects.

Limitation 1. It is only valid for the Ising and Potts models, and furthermore it requires that the number of labels L is known in advance. In many applications,



(a) (b)

Figure 2.2: The coupling time empirical plots and the Huber bounds for Ising model.

such as image analysis, L is the number of objects (or image regions) which has to be inferred from the input data.

Limitation 2. It slows down in the presence of external field, i.e input data. For example, in the image analysis problem, our goal is to infer the label $\mathbf{X}$ from the input image $\mathbf{I}$ and the target probability is a Bayesian posterior probability where $\pi_{\text{PTS}}(\mathbf{X})$ is used as a prior model,

$$\pi(\mathbf{X}) = \pi(\mathbf{X}|\mathbf{I}) \propto \mathcal{L}(\mathbf{I}|\mathbf{X})\pi_{\text{PTS}}(\mathbf{X}). \tag{2.18}$$

$\mathcal{L}(\mathbf{I}|\mathbf{X})$ is the likelihood model, such as independent Gaussians $N(\bar{\mathbf{I}}_c, \sigma_c^2)$ for each coloring $c = 1, 2, ..., \text{L}$,

$$\mathcal{L}(\mathbf{I}|\mathbf{X}) \propto \prod_{c=1}^{\text{L}} \prod_{x_i=c} \frac{1}{\sqrt{2\pi}\sigma_c} \exp\{-\frac{(\mathbf{I}(v_i) - \bar{\mathbf{I}}_c)^2}{2\sigma_c^2}\}. \tag{2.19}$$

The slowing down is in large amount attributed to the fact that the Bernoulli probability $\rho = 1 - e^{-\beta}$ for the auxiliary variables is calculated independently of the input image.

2.1.4 SW Interpretation 2: slice sampling and decoupling

In the presence of external field (data), the SW method can be interpreted and extended by the auxiliary method proposed by Higdon [26]. Suppose we write the target probability in a more general form,

$$\pi(\mathbf{X}) = \frac{1}{Z}\prod_{v_i \in V}\phi_i(x_i)\cdot\prod_{<i,j>\in E}\psi(x_i,x_j),\quad \phi()>0,\psi()>0. \tag{2.20}$$

For the Potts model above, we have $\psi(x_i,x_j) = e^{\beta\mathbf{1}(x_i=x_j)}$. Higdon [26] introduced a continuous variable on the edges as the *bond strength*,

$$W = \{\omega_{ij}:\ \omega_{ij}\in[0,+\infty),\forall <i,j>\in E\} \tag{2.21}$$

In contrast to the Bernoulli probability for the binary variable μ_{ij} in eqn. (2.3), the bond variables follow uniform probabilities, depending on $\mathbf{X}$,

$$\omega_{ij}|(x_i,x_j)\sim \text{Unif}[0,\psi(x_i,x_j)] = \psi^{-1}(x_i,x_j)\mathbf{1}(0\le\omega_{ij}\le\psi(x_i,x_j)). \tag{2.22}$$

Thus a conditional probability is constructed as

$$p_{\text{HGD}}(W|\mathbf{X}) = \prod_{<i,j>\in E}p(\omega_{ij}|x_i,x_j) = \prod_{<i,j>\in E}\psi^{-1}(x_i,x_j)\mathbf{1}(0\le\omega_{ij}\le\psi(x_i,x_j)). \tag{2.23}$$

This formula is chosen to cancel the internal field in a joint probability,

$$p_{\text{HGD}}(\mathbf{X},W) = \pi(\mathbf{X})p(W|\mathbf{X}) = \frac{1}{Z}[\prod_{v_i\in V}\phi_i(x_i)]\cdot[\prod_{<i,j>\in E}\mathbf{1}(0\le\omega_{ij}\le\psi(x_i,x_j))]. \tag{2.24}$$

We have the second conditional probability by the Bayes rule,

$$p_{\text{HGD}}(\mathbf{X}|W) = \frac{1}{Z'}[\prod_{v_i\in V}\phi_i(x_i)]\cdot[\prod_{<i,j>\in E}\mathbf{1}(0\le\omega_{ij}\le\psi(x_i,x_j))] \tag{2.25}$$

That is, given the bond strength ω_{ij}, x_i and x_j must be sampled so that the condition $\psi(x_i,x_j)\ge\omega_{ij}$ is observed. This idea is called "slice sampling". In

case of the Potts model, this becomes,

$$p(\mathbf{X}|W) = \frac{1}{Z'}[\prod_{v_i \in V} \phi_i(x_i)] \cdot [\prod_{<i,j> \in E} \mathbf{1}(0 \leq \omega_{ij} \leq e^{\beta \mathbf{1}(x_i = x_j)}] \tag{2.26}$$

Given W, the second product imposes a hard constraint on $\mathbf{X}$. If $\omega_{ij} \leq 1$, $\mathbf{1}(0 \leq \omega_{ij} \leq e^{\beta \mathbf{1}(x_i = x_j)}) = 1$ is satisfied for any x_i, x_j, because $\beta > 0$ and $e^{\beta \mathbf{1}(x_i = x_j)} \geq 1$. Thus it imposes no constraints on x_i, x_j. If $\omega_{ij} > 1$, then it imposes the constraint that $x_i = x_j$. Thus the auxiliary variables μ_{ij} and ω_{ij} are linked by the following equation,

$$\mu_{ij} = \mathbf{1}(\omega_{ij} > 1), \ \ \forall < i, j > \in E. \tag{2.27}$$

Thus turning on the edges is equivalent to $\omega_{ij} > 1$.

$$E_{\text{on}}(W) = \{e =< ij >: \ \omega_{ij} > 1, < i, j > \in E\}. \tag{2.28}$$

Given W, we have the set of connected components and the vertices in each component receive the same color.

$$\text{CP}(W) = \{\text{cp}_k : k = 1, 2, ..., K, \cup_{i=1}^{K} \text{cp}_k = V\}. \tag{2.29}$$

As the hard constraints are absorbed by the connected component, the conditional probability in eqn. (2.26) becomes

$$p_{\text{HGD}}(\mathbf{X}|W) = \prod_{k=1}^{K} \prod_{v_i \in \text{cp}_k} \phi_i(x_i). \tag{2.30}$$

As we can see, the coloring of each connected component is independent of other vertices (completely decoupled !). In the special case when $\phi_i(x_i) = 1$, it reduces to the RCM model in the previous subsection.

In summary $p_{\text{HGD}}(\mathbf{X}, W)$, like $p_{\text{ES}}(\mathbf{X}, \mathbf{U})$ in eqn.(2.7), has marginal probability being the target $\pi(\mathbf{X})$ and has two conditional probabilities that are easy to sample. There are two problems with this design.

Firstly, although the decoupling idea with conditional probability $p_{\mathrm{HGD}}(W|\mathbf{X})$ in eqn. (2.25) is valid for any pair clique Markov random field models and thus goes beyond the Potts model, the hard constraints may become impractical to compute for non-Potts model. That is, given W, the constraint conditions on $\mathbf{X}$ are no longer expressed as clustering. Many slice sampling methods suffer from this problem.

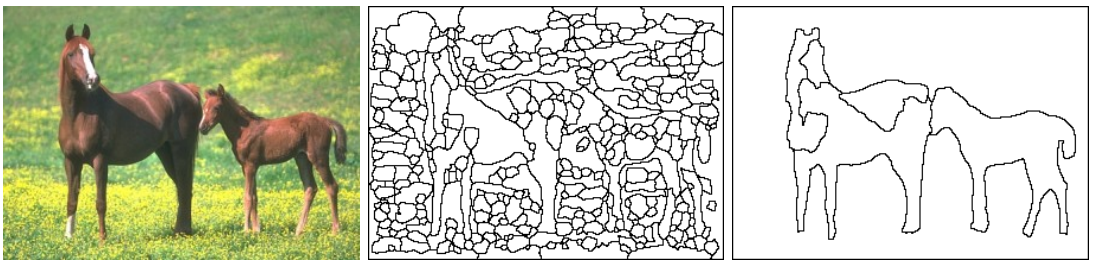
Secondly, although the flipping step in eqn.(2.30) makes use of the data, the clustering step in eqn. (2.23) does not. It is similar to the original SW method. This in practice often makes the constructed cluster ineffective.

2.2 Generalizing SW to arbitrary probabilities on graphs

In this section, we generalize the SW to arbitrary probabilities from the perspective of Metropolis-Hastings method ([37], [25]). Our method iterates three steps: (i) a clustering step driven by data, (ii) a label flipping step which can introduce new labels, and (iii) an acceptance step for the proposed labelling. A key observation is the simplicity of the formula expressing the acceptance probability.

We will clarify the three steps in the following three subsections, and then we show how our method applied to the Potts model reduces to the original SW.

We illustrate the algorithm by an example on image segmentation shown in Fig. 2.3. Fig. 2.3.(a) is an input image $\mathbf{I}$ on a lattice Λ, which is decomposed into a number of "atomic regions" to reduce the graph size in a preprocessing stage. Each atomic region has nearly constant intensity and is a vertex in the graph $\mathbf{G}$. Two vertices are connected if their atomic regions are adjacent (i.e. sharing boundary). Fig. 2.3.(c) is a result by our algorithm optimizing a Bayesian probability $\pi(\mathbf{X}) = \pi(\mathbf{X}|\mathbf{I})$ (see chapter 3 for details). The result $\mathbf{X}$ assigns a



(a). Input image (b). atomic regions (c). segmentation

Figure 2.3: Example of image segmentation. (a). Input image. (b). Atomic regions by edge detection followed by edge tracing and contour closing. each atomic region is a vertex in the graph **G**. c. Segmentation (labeling) result where each closed region is assigned a color or label.

uniform color to all vertices in each close region, which hopefully corresponds to an object in the scene or a part of it. Note that the number of objects or colors L is unknown, and we do not distinguish between the different permutations of the labels.

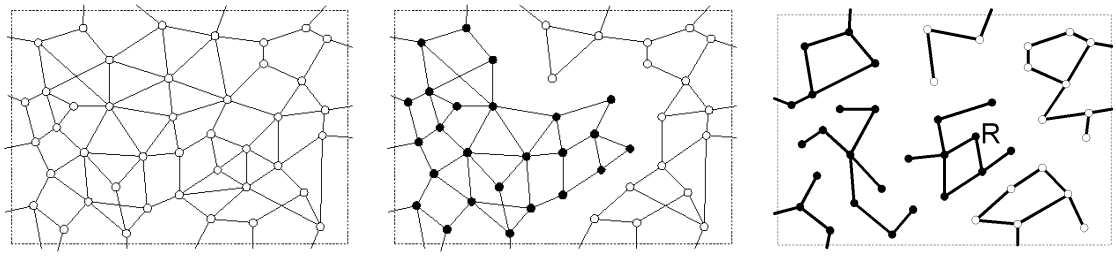
2.2.1 Step 1: data-driven clustering

We augment the adjacency graph **G** with a set of binary variables on the edges $\mathbf{U} = \{\mu_{ij} :< i, j >\in E\}$, as in the original SW method. Each μ_{ij} follows a Bernoulli probability depending on the current state of the two vertices x_i and x_j,

$$\mu_{ij}|(x_i, x_j) \sim \text{Bernoulli}(q_{ij}\mathbf{1}(x_i = x_j)), \quad \forall < i, j >\in E. \tag{2.31}$$

q_{ij} is an empirical probability on edge $< i, j >$ that tells how likely the two vertices v_i and v_j have the same label. In Bayesian inference where the target $\pi(\mathbf{X})$ is a posterior probability, then q_{ij} can be better informed by the data.

For the image segmentation example, $q_{ij} = \exp\{-\frac{1}{2}(KL(h_i||h_j) + KL(h_j||h_i))$



a. $\mathbf{G}$ b. Current partition state $\mathbf{X}$ c. CP

Figure 2.4: Three stages of graphs in the algorithm. a) Adjacency graph $\mathbf{G}$, b. The graph of the current partition (coloring) $\mathbf{X}$, c. connected components CP obtained in the clustering step.

is computed based on the similarity between image intensity histograms h_i, h_j at v_i and v_j (or their local neighborhoods) and it is an approximate to the marginal probability of $\pi(\mathbf{X}|\mathbf{I})$,

$$q_{ij} = q(x_i = x_j|\mathbf{I}(v_i), \mathbf{I}(v_j)) \approx \pi(x_i = x_j|\mathbf{I}). \tag{2.32}$$

The design of $q(x_i = x_j|\mathbf{I}(v_i), \mathbf{I}(v_j))$ is application specific and is part of the so called discriminative methods. In the applications chapters 3, 4, 5, 6 we will show how to define the edge weights for each individual application.

Our method will work for any q_{ij}, but a good choice will inform the clustering step and achieve faster convergence. In the ideal case when the segmentation is known, choosing $q_{ij} = 1$ if and only if i and j are part of the same label, otherwise $q_{ij} = 0$, will result in convergence in approximately $|L|$ steps.

Fig. 2.5 shows nine clustering examples of the horse image. In these examples, we set all vertices to the same color ($\mathbf{X} = c$) and sample the edge probability independently,

$$\mathbf{U}|(\mathbf{X} = c) \sim \prod_{<i,j>\in E} \text{Bernoulli}(q_{ij}). \tag{2.33}$$

The connected components in CP($\mathbf{U}$) are shown by the differently colored regions.

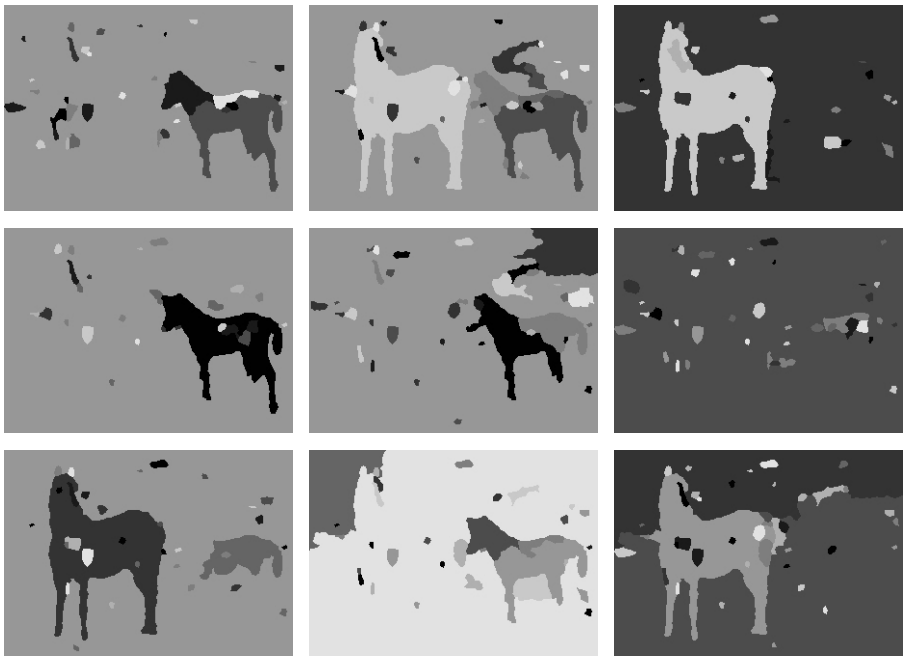


Figure 2.5: Nine examples of connected components for the horse image computed using discriminative edge probabilities.

We repeat the clustering step nine times. As we can see, the edge probabilities lead to "meaningful" clusters which correspond to distinct objects in the image. Such effects cannot be observed using constant edge probability.

2.2.2 Step 2: flipping of color

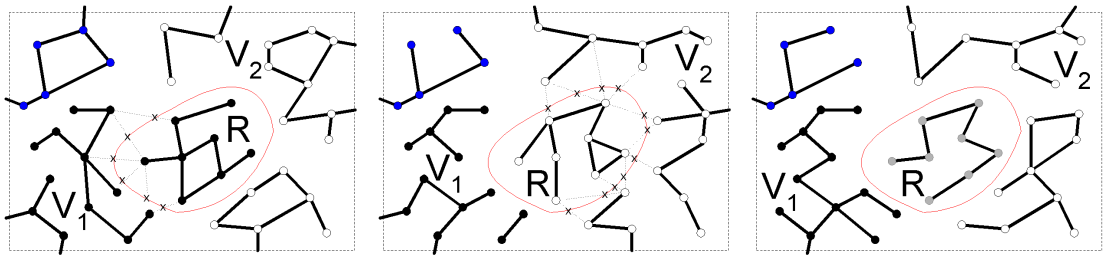
Let $\mathbf{X} = (V_1, V_2, ..., V_n)$ be the current coloring state. The edge variables $\mathbf{U}$, sampled conditional on $\mathbf{X}$, decompose $\mathbf{X}$ into a number of connected components

$$\mathrm{CP}(\mathbf{U}|\mathbf{X}) = \{\mathrm{cp}_i : i = 1, 2, ..., N(\mathbf{U}|\mathbf{X})\}. \tag{2.34}$$

Suppose we select one connected component $R \in \mathrm{CP}(\mathbf{U}|\mathbf{X})$ with color $\mathbf{X}_R = \ell \in \{1, 2, ..., n\}$, and assign its color to $\ell' \in \{1, 2, ..., n, n+1\}$ with some probability $q(l'|R, \mathbf{X})$ that needs to be designed, obtaining a new state $\mathbf{X}'$. The probability

$q(l'|R, \mathbf{X})$ is another place when one could use data driven information to speed-up convergence, instead of choosing a uniform probability. We will show in the applications chapters 3, 4, 5, 6 how it is designed in each case. A reasonably good choice is to assign a higher probability to colors of nodes adjacent to R and a small probability to all other colors and to a new color.

After changing the color of R from l to l', the total number of colors can stay the same, increase, decrease, as explained below and shown in Fig. 2.6.



(a) State $\mathbf{X}_A$ (b) State $\mathbf{X}_B$ (c) State $\mathbf{X}_C$

Figure 2.6: Three labeling states $\mathbf{X}_A, \mathbf{X}_B, \mathbf{X}_C$ that differ only in the color of a cluster R.

1. The canonical case: $R \subset V_\ell$ and $\ell' \leq n$. That is, a portion of V_ℓ is re-grouped into an existing color $V_{\ell'}$, and the number of colors remains $\mathrm{L} = n$ in π'. The moves between $\mathbf{X}_A \leftrightarrow \mathbf{X}_B$ in Fig. 2.6 are examples.

2. The merge case: $R = V_\ell$ in $\mathbf{X}$ is the set of all vertices that have color ℓ and $\ell' \leq n$, $\ell \neq \ell'$. That is, color V_ℓ is merged to $V_{\ell'}$, and the number of distinct colors reduces to $n - 1$ in $\mathbf{X}'$. The moves $\mathbf{X}_C \to \mathbf{X}_A$ or $\mathbf{X}_C \to \mathbf{X}_B$ in Fig. 2.6 are examples.

3. The split case: $R \subset V_\ell$ and $\ell' = n + 1$. V_ℓ is split into two pieces and the number of distinct color increases to $n + 1$ in $\mathbf{X}'$. The moves $\mathbf{X}_A \to \mathbf{X}_C$ in Fig.2.6 are examples.

Note that this color flipping step is also different from the original SW with Potts model as we allow new colors in each step. The number of color L is not fixed.

2.2.3 Step 3: accepting the flipping

The previous two steps basically have proposed a move between two states $\mathbf{X}$ and $\mathbf{X}'$ which differ in the coloring a connected component R. In the third step we accept the move with the probability given by the Metropolis-Hastings method:

$$\alpha(\mathbf{X} \to \mathbf{X}') = \min\{1, \frac{q(\mathbf{X}' \to \mathbf{X})}{q(\mathbf{X} \to \mathbf{X}')} \cdot \frac{\pi(\mathbf{X}')}{\pi(\mathbf{X})}\}. \tag{2.35}$$

$q(\mathbf{X}' \to \mathbf{X})$ and $q(\mathbf{X} \to \mathbf{X}')$ are the proposal probabilities between $\mathbf{X}$ and $\mathbf{X}'$. If the proposal is rejected, the Markov chain stays at state $\mathbf{X}$. To make this equation explicit, we need the following

Definition 1. *Let $\mathbf{X} = (V_1, V_2, ..., V_{\mathrm{L}})$ be a coloring state, and $R \in \mathrm{CP}(U|\mathbf{X})$ a connected component, the "cut" between R and the set V_k of nodes with color k is the set of edges between R and $V_k \backslash R$,*

$$\mathcal{C}(R, V_k) = \{< i, j >\in E : \ i \in R, j \in V_k \backslash R\}, \ \ \forall k.$$

The crosses in Fig. 2.6, state $\mathbf{X}_A$ and state $\mathbf{X}_B$ show the cuts $\mathcal{C}(R, V_1)$ and $\mathcal{C}(R, V_2)$ respectively. In Fig. 2.6.(c), $R = V_3$ and thus $\mathcal{C}(R, V_3) = \emptyset$ and we have $\prod_{<i,j>\in\mathcal{C}(R,V_3)}(1 - q_{ij}) = 1$.

Now we can state the main result, which gives the acceptance probability

Theorem 4. *The acceptance probability for the proposed cluster flipping is,*

$$\alpha(\mathbf{X} \to \mathbf{X}') = \min\{1, \frac{\prod_{<i,j>\in\mathcal{C}(R,V_{\ell'})}(1 - q_{ij})}{\prod_{<i,j>\in\mathcal{C}(R,V_{\ell})}(1 - q_{ij})} \cdot \frac{q(\mathbf{X}_R = \ell|R, \mathbf{X}')}{q(\mathbf{X}_R = \ell'|R, \mathbf{X})} \cdot \frac{\pi(\mathbf{X}')}{\pi(\mathbf{X})}\}. \tag{2.36}$$

In the special case when $R = V_\ell$, then the cut is $\mathcal{C}(R, V_\ell) = \emptyset$ and therefore we have $\prod_{<i,j>\in\mathcal{C}(R,V_\ell)}(1 - q_{ij}) = 1$.

Proof. The proof is given in Apendix A. □

2.2.4 The Swendsen-Wang Cuts Algorithm

In the image segmentation application in Chapter 3, we observe experimentally that our cluster sampling method is $O(100)$ times faster than the single site Gibbs sampler in terms of CPU time. We refer to plots and comparison in Figs.(3.3), (3.4) and (3.5) in Chapter 3 for details.

We give our algorithm described in steps 1-3 above the name "Swendsen-Wang Cuts". We can summarize it in the following two versions, which differ in the way they choose the connected component R.

The first version, SWC-1, turns on/off the edges of the whole graph and picks one component randomly.

Swendsen-Wang Cuts: SWC-1

Input: $\mathbf{G} =< V, E >$, $q_e, \forall e \in E$, and posterior $p(W|\mathbf{I})$.

Output: Samples $W \sim p(W|\mathbf{I})$.

1. Initialize a partition $\mathbf{X}$ by random clustering
2. Repeat, for current state $\mathbf{X} = (V_1, V_2, ..., V_n)$,
3. For $e =< i, j >\in E$, turn $\mu_{ij} =$ on with probability q_{ij} if $x_i = x_j$, else $\mu_{ij} =$ off.
4. $V_\ell = (V_{\ell 1}, ..., V_{\ell n_\ell})$ is divided into n_ℓ connected components for $\ell = 1, 2, ..., n$.
5. Collect all the connected components in $\mathrm{CP}(\mathbf{U}|\mathbf{X}) = \{V_{\ell i} : \ell = 1, .., n, i = 1, .., n_\ell\}$.
6. Select a connected component $R \in \mathrm{CP}(\mathbf{U}|\mathbf{X})$ with prob. $q(R \,|\, \mathrm{CP}(\mathbf{U}|\mathbf{X}))$, say $R \subset V_\ell$. (*Usually* $q(R \,|\, \mathrm{CP}(\mathbf{U}|\mathbf{X})) = \frac{1}{|\mathrm{CP}(\mathbf{U}|\mathbf{X})|}$ *is uniform*).
7. Propose to assign R a new label $\mathbf{c}_R = \ell'$ with probability $q(\ell'|R, \mathbf{X})$, obtaining $\mathbf{X}'$.
8. Accept the proposal with probability $\alpha(\mathbf{X} \to \mathbf{X}')$ defined in theorem 4 or 5.

The second version, SWC-2, grows the connected component R from a seed node v, in a similar fashion to the Wolff [60] variant of the original Swendsen-Wang algorithm.

Swendsen-Wang Cuts: SWC-2

1. Repeat, for current state $\mathbf{X} = (V_1, V_2, ..., V_n)$,
2. Select a seed vertex v, say $v \in V_\ell$ in $\mathbf{X}$. Set $R \leftarrow \{v\}$, $\mathcal{C} \leftarrow \emptyset$,
3. Repeat until $\mathcal{C} \cap \mathcal{C}(R, V_\ell \setminus R) = \mathcal{C}(R, V_\ell \setminus R)$,
4. For any $e =< i, j > \in \mathcal{C}(R, V_\ell \setminus R)$, $i \in R$, $j \in V_\ell \setminus R$.
5. If $x_i = x_j$, turn $\mu_{ij} =$ on with probability q_{ij}, else $\mu_{ij} =$ off,
6. If $\mu_{ij} =$ on, set $R \leftarrow R \cup \{j\}$, else $\mathcal{C} \leftarrow \mathcal{C} \cup \{e\}$.
7. Propose to assign R a new label ℓ' with probability $q(\mathbf{c}_R = \ell' | R, \mathbf{X})$.
8. Accept the move with probability $\alpha(\mathbf{X} \rightarrow \mathbf{X}')$ defined in theorem 4 or 5.

Theorem 4 assumed that the posterior probability only depends on the partition X. However, it is very often necessary to incorporate parametric models for the different regions (sets of nodes of the same color). For example, in image segmentation application in chapter 3, the intensity of each region is modeled either using a constant intensity model, a linear model with 3 parameters or a quadratic model with 6 parameters. Then the hidden variables will be $W = (\mathbf{X}, \mu)$, consisting of the partition $\mathbf{X} = \{V_1, ..., V_n\}$ and the model parameters $\mu = (\mu_1, ..., \mu_n\}$. If the model parameters are obtained deterministically, then Theorem 4 can be applied. If they are not deterministically obtained, we assume that they can be obtained by sampling from some modeling proposals $q_m(\mu_i | V_i)$. Then Theorem 4 extends naturally to

Theorem 5. *(SW Cuts with model switching). Consider a candidate component R selected by SWC. Let $q_m(\mu_i | V_i)$ be a proposal probability from which the model μ_i of a subgraph V_i of the partition is chosen by sampling. If the proposed move*

to reassign R from V_ℓ to $V_{\ell'}$, and then change the model of V_ℓ from $\mu_\ell^{\mathbf{X}}$ to $\mu_\ell^{\mathbf{X}'}$ and the model of $V_{\ell'}$ from $\mu_{\ell'}^{\mathbf{X}}$ to $\mu_{\ell'}^{\mathbf{X}'}$ is accepted with probability

$$\alpha(\mathbf{X} \to \mathbf{X}') = \min(1, \frac{q_m(\mu_\ell^{\mathbf{X}}|V_\ell \cup R)q_m(\mu_{\ell'}^{\mathbf{X}}|V_{\ell'} - R)}{q_m(\mu_\ell^{\mathbf{X}'}|V_\ell - R)q_m(\mu_{\ell'}^{\mathbf{X}'}|V_{\ell'} \cup R)} \frac{\prod_{<i,j>\in\mathcal{C}(R,V'_\ell)}(1-q_{ij})}{\prod_{<i,j>\in\mathcal{C}(R,V_\ell)}(1-q_{ij})} \frac{q(\ell|R,\mathbf{X}')\pi(\mathbf{X}')}{q(\ell'|R,\mathbf{X})\pi(\mathbf{X})}) \tag{2.37}$$

then the Markov chain is reversible and ergodic.

2.2.5 SW Interpretation 3: the Metropolis-Hastings perspective

Now we are ready to derive the original SW method as a special case.

Proposition 3. *If we set all the edge probabilities to a constant $q_{ij} = 1 - e^{-\beta}$, and we choose the new label l' uniformly, $q(\mathbf{X}_R = \ell'|R, \mathbf{X}) = 1/|L|$, then*

$$\frac{q(\mathbf{X}' \to \mathbf{X})}{q(\mathbf{X} \to \mathbf{X}')} = \frac{\prod_{<i,j>\in\mathcal{C}(R,V_\ell)}(1-q_{ij})}{\prod_{<i,j>\in\mathcal{C}(R,V_{\ell'})}(1-q_{ij})} = \exp\{\beta(|\mathcal{C}(R,V_{\ell'})| - |\mathcal{C}(R,V_\ell)|)\}, \tag{2.38}$$

where $|\mathcal{C}|$ is the cardinality of the set.

As $\mathbf{X}$ and $\mathbf{X}'$ only differ in labeling R, the potentials for the Potts model only differ at the "cracks" between R and V_ℓ and $V_{\ell'}$ respectively.

Proposition 4. *For the Potts model $\pi(\mathbf{X}) = p_o(\mathbf{X}) = \pi_{\text{PTS}}(\mathbf{X})$,*

$$\frac{\pi_{\text{PTS}}(\mathbf{X}_R = \ell'|\mathbf{X}_{\partial R})}{\pi_{\text{PTS}}(\mathbf{X}_R = \ell|\mathbf{X}_{\partial R})} = \exp\{\beta(|\mathcal{C}(R,V_\ell)| - |\mathcal{C}(R,V_{\ell'})|)\} \tag{2.39}$$

Therefore, following eq. (2.36) (where the proposal probabilities for the labels are uniform), the acceptance probability for the Potts model is always one, due to cancellation.

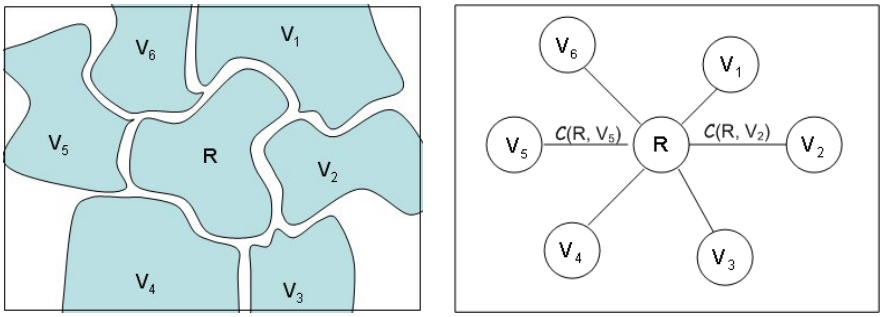
$$\alpha(\mathbf{X} \to \mathbf{X}') = 1. \tag{2.40}$$

Therefore the third acceptance step is always omitted. This interpretation is related to the Wolff [60] modification (see also Liu [35], p157).

2.3 Variants of the SWC method

In this section, we briefly discuss two variants of the cluster sampling method.

2.3.1 Cluster Gibbs sampling — the "hit-and-run" perspective



(a) (b)

Figure 2.7: Illustrating the cluster Gibbs sampler. (a) The cluster R has a number of neighboring components of uniform color. (b) The cuts between R and its neighboring colors. The sampler follows a conditional probability modified by the edge strength defined on the cuts.

With a slight change, we can modify the cluster sampling method to become a generalized Gibbs sampler.

Suppose that $R \in \mathrm{CP}(U|\mathbf{X})$ is the candidate chosen in the clustering step, Fig. 2.7 shows its cuts with adjacent sets

$$\mathcal{C}(R, V_k),\ k = 1, 2, ..., \mathrm{L}(\mathbf{X}).$$

We compute the cut weight γ_k as the strength of connectivity between R and $V_k \backslash R$,

$$\gamma_k = \prod_{<i,j> \in \mathcal{C}(R,V_k)} (1 - q_{ij}). \tag{2.41}$$

Proposition 5. *Let $\pi(\mathbf{X})$ be the target probability, in the notation above. If R is relabelled probabilistically with*

$$q(\mathbf{X}_R = k|R, \mathbf{X}) \propto \gamma_k \pi(\mathbf{X}_R = k|\mathbf{X}_{\partial R}), \quad k = 1, 2, ..., N(\mathbf{X}), \tag{2.42}$$

then the acceptance probability is always 1 in the third step.

Proof. Let the label of R in state $\mathbf{X}$ be $\mathbf{X}_R = \ell$ and after relabeling $\mathbf{X}'_R = \ell'$. From Theorem 4, we obtain

$$\alpha(\mathbf{X} \to \mathbf{X}') = \min\{1, \frac{\gamma_{\ell'}}{\gamma_\ell} \cdot \frac{q(\mathbf{X}_R = \ell|R, \mathbf{X}')}{q(\mathbf{X}_R = \ell'|R, \mathbf{X})} \cdot \frac{\pi(\mathbf{X}')}{\pi(\mathbf{X})}\}. \tag{2.43}$$

We observe that the number and values of γ_k do not depend on the particular value of $\mathbf{X}_R$, so in both states $\mathbf{X}, \mathbf{X}'$, all γ_k are the same. Since $\mathbf{X}_{\partial R} = \mathbf{X}'_{\partial R}$, we have

$$\sum_{k=1}^{N(\mathbf{X})} \gamma_k \cdot \pi(\mathbf{X}_R = k|\mathbf{X}_{\partial R}) = \sum_{k=1}^{N(\mathbf{X}')} \gamma_k \cdot \pi(\mathbf{X}'_R = k|\mathbf{X}'_{\partial R}) \tag{2.44}$$

so

$$\frac{q(\mathbf{X}_R = \ell|R, \mathbf{X}')}{q(\mathbf{X}_R = \ell'|R, \mathbf{X})} = \frac{\gamma_\ell \cdot \pi(\mathbf{X})}{\gamma_{\ell'} \cdot \pi(\mathbf{X}')} \tag{2.45}$$

So we get

$$\alpha(\mathbf{X} \to \mathbf{X}') = \min\{1, \frac{\gamma_{\ell'}}{\gamma_\ell} \cdot \frac{\gamma_\ell \cdot \pi(\mathbf{X})}{\gamma_{\ell'} \cdot \pi(\mathbf{X}')} \cdot \frac{\pi(\mathbf{X}')}{\pi(\mathbf{X})}\} = 1, \tag{2.46}$$

which means the move is always accepted. □

This yields a generalized Gibbs sampler which flips the color of a cluster according to a modified conditional probability.

Cluster Gibbs Sampler Algorithm: SWC-3

1. Repeat, for a current partition $\pi = (V_1, ..., V_n)$.
2. Clustering step: Select a candidate set R as in SWC-1 or SWC-2
3. Flipping step: relabel R according to eqn. (2.42)

The traditional single site Gibbs sampler [17] is a special case when $q_{ij} = 0$ for all $< i, j >$ and thus $R = \{v\}$ and $\gamma_k = 1$ for all k.

One may also view the above method from the perspective of hit-and-run. In continuous state space, a hit-and-run method [19] chooses a new direction $\vec{e}$ (random ray) at time t and then sample on this direction by $a \sim \pi(x + a\vec{e})$. Liu and Wu [34] extended it to any compact groups of actions. In finite state space Ω, one can choose any finite sets $\Omega_a \subset \Omega$ and then apply the Gibbs sampler within the set[1].

But it is difficult to choose good directions or subsets in hit-and-run methods. In the cluster Gibbs sampler presented above, the subset is selected by the auxiliary variables on the edges.

2.3.2 The multiple flipping scheme

Given a set of connected components $\mathrm{CP}(\mathbf{U}|\mathbf{X})$ (see eqn. (2.34)) after the clustering step, instead of flipping a single component R, we can flip all (or any chosen number of) connected components simultaneously. There is room for designing the proposal probabilities for labeling these connected components, independently or jointly. In what follows, we assume the labels are chosen independently for each connected component $\mathrm{cp} \in \mathrm{CP}(\mathbf{U}|\mathbf{X})$, by sampling from a proposal probability $q(\mathbf{X}_{\mathrm{cp}} = l|\mathrm{cp})$. Suppose we obtain a new labeling state $\mathbf{X}'$ after flipping. Let $E_{\mathrm{on}}(\mathbf{X}) \subset E$ and $E_{\mathrm{on}}(\mathbf{X}') \subset E$ be the subsets of edges that connect the vertices of same color in $\mathbf{X}$ and $\mathbf{X}'$ respectively. We define two cuts as the differences of the sets

$$\mathcal{C}(\mathbf{X} \to \mathbf{X}') = E_{\mathrm{on}}(\mathbf{X}') - E_{\mathrm{on}}(\mathbf{X}), \text{ and } \mathcal{C}(\mathbf{X}' \to \mathbf{X}) = E_{\mathrm{on}}(\mathbf{X}) - E_{\mathrm{on}}(\mathbf{X}'), \quad (2.47)$$

[1] Persi Diaconis once discussed a unifying view of hit-and-run for MCMC in a talk in 2002.

We denote the set of connected components which have different colors before and after the flipping by $D(\mathbf{X}, \mathbf{X}') = \{\text{cp} : \mathbf{X}_{\text{cp}} \neq \mathbf{X}'_{\text{cp}}\}$.

Proposition 6. *The acceptance probability of the multiple flipping scheme is*

$$\alpha(\mathbf{X} \to \mathbf{X}') = \min\{1, \frac{\prod_{<i,j>\in\mathcal{C}(\mathbf{X}\to\mathbf{X}')}(1-q_{ij})}{\prod_{<i,j>\in\mathcal{C}(\mathbf{X}'\to\mathbf{X})}(1-q_{ij})} \frac{\prod_{\text{cp}\in D(\mathbf{X},\mathbf{X}')} q(\mathbf{X}'_{\text{cp}}|\text{cp})}{\prod_{\text{cp}\in D(\mathbf{X},\mathbf{X}')} q(\mathbf{X}_{\text{cp}}|\text{cp})} \cdot \frac{p(\pi')}{p(\pi)}\} \tag{2.48}$$

Proof. We will proceed in a similar fashion with the proof of Theorem 4, and we maintain the same notations for $\mathbf{U}_{\text{on}}, \mathbf{U}_{\text{off}}, \text{CP}(\mathbf{U}|\mathbf{X})$.

In state $\mathbf{X}$, let $\mathbf{U}$ be one of the many sets of auxiliary variables that can be used to obtain the connected components $D(\mathbf{X}, \mathbf{X}')$. Then any $\text{cp} \in D(\mathbf{X}, \mathbf{X}')$ is connected through edges of $\mathbf{U}_{\text{on}}$. The probability to obtain state $\mathbf{X}'$ through flipping the components from $\text{CP}(\mathbf{U}|\mathbf{X})$ independently is

$$q(\mathbf{X}'|\mathbf{U}, \mathbf{X}) = \prod_{\text{cp}\in\text{CP}(\mathbf{U}|\mathbf{X})} q(\mathbf{X}'|\text{cp}) \tag{2.49}$$

The probability to go from state $\mathbf{X}$ to $\mathbf{X}'$ is

$$q(\mathbf{X}'|\mathbf{X}) = \sum_{\mathbf{U}} \prod_{\text{cp}\in\text{CP}(\mathbf{U}|\mathbf{X})} q(\mathbf{X}'|\text{cp}) \prod_{<i,j>\in\mathbf{U}_{\text{on}}} q_{ij} \prod_{<i,j>\in\mathbf{U}_{\text{off}}} (1-q_{ij}) \tag{2.50}$$

Let

$$^{-}\mathbf{U}_{\text{off}} = \mathbf{U}_{\text{off}} \backslash \mathcal{C}(\mathbf{X} \to \mathbf{X}') \tag{2.51}$$

Then

$$q(\mathbf{X}'|\mathbf{X}) = \prod_{<i,j>\in\mathcal{C}(\mathbf{X}\to\mathbf{X}')} (1-q_{ij}) \prod_{\text{cp}\in D(\mathbf{X},\mathbf{X}')} q(\mathbf{X}'|\text{cp}) \sum_{\mathbf{U}} \prod_{\text{cp}\in\text{CP}(\mathbf{U}|\mathbf{X})\backslash D(\mathbf{X},\mathbf{X}')} q(\mathbf{X}'|\text{cp}) \prod_{<i,j>\in\mathbf{U}_{\text{on}}} q_{ij} \prod_{<i,j>\in{}^{-}\mathbf{U}_{\text{off}}} (1-q_{ij}) \tag{2.52}$$

Similarly, the probability of going from state $\mathbf{X}'$ to $\mathbf{X}$ is

$$q(\mathbf{X}|\mathbf{X}') = \prod_{<i,j>\in\mathcal{C}(\mathbf{X}'\to\mathbf{X})} (1-q_{ij}) \prod_{\text{cp}\in D(\mathbf{X},\mathbf{X}')} q(\mathbf{X}|\text{cp}) \sum_{\mathbf{U}'} \prod_{\text{cp}\in\text{CP}(\mathbf{U}'|\mathbf{X}')\backslash D(\mathbf{X},\mathbf{X}')} q(\mathbf{X}|\text{cp}) \prod_{<i,j>\in\mathbf{U}'_{\text{on}}} q_{ij} \prod_{<i,j>\in{}^{-}\mathbf{U}'_{\text{off}}} (1-q_{ij}) \tag{2.53}$$

Similarly to the proof of Theorem 4, there is a one-to-one correspondence between auxiliary variables $\mathbf{U}$ in state $\mathbf{X}$ and $\mathbf{U}'$ in state $\mathbf{X}'$ such that such that

$$\mathrm{CP}(\mathbf{U}|\mathbf{X}) = \mathrm{CP}(\mathbf{U}'|\mathbf{X}') \tag{2.54}$$

and

$$\mathbf{U}_{\mathrm{on}} = \mathbf{U}'_{\mathrm{on}}, {}^{-}\mathbf{U}_{\mathrm{off}} = {}^{-}\mathbf{U}'_{\mathrm{off}}. \tag{2.55}$$

Then the sums in eqs. 2.52 and 2.53 are equal, so we obtain, by cancellation

$$\frac{q(\mathbf{X}|\mathbf{X}')}{q(\mathbf{X}'|\mathbf{X})} = \frac{\prod_{<i,j>\in\mathcal{C}(\mathbf{X}\to\mathbf{X}')}(1-q_{ij})}{\prod_{<i,j>\in\mathcal{C}(\mathbf{X}'\to\mathbf{X})}(1-q_{ij})} \frac{\prod_{\mathrm{cp}\in D(\mathbf{X},\mathbf{X}')} q(\mathbf{X}'|\mathrm{cp})}{\prod_{\mathrm{cp}\in D(\mathbf{X},\mathbf{X}')} q(\mathbf{X}|\mathrm{cp})} \tag{2.56}$$

which, by applying the Metropolis acceptance eq. 2.35, gives the desired result. □

Observe that when $D = \{R\}$ is a single connected component, this reduces to Theorem 4.

It is worth mentioning that if we flip all connected components simultaneously, then the Markov transition graph of $\mathcal{K}(\mathbf{X}, \mathbf{X}')$ is fully connected, i.e.

$$\mathcal{K}(\mathbf{X}, \mathbf{X}') > 0, \quad \forall \mathbf{X}, \mathbf{X}' \in \Omega. \tag{2.57}$$

This means that the Markov chain can walk with non-zero probability between any two partitions in a single step.

2.3.3 The multi-cue Swendsen-Wang Cuts

Sometimes there are many cues which provide bottom-up information for the graph partitioning. For example there are different types of texture, intensity, motion cues. How can we combine all these cues while maintaining detailed balance?

Great help for answering this question comes from the following

Theorem 6. *Let $q_1, ..., q_n$ be Markov moves with transition kernels $K_1, ..., K_n$, such that all q_i observe detailed balance with respect to the same probability p. Let $\alpha_1, ..., \alpha_n \geq 0$ be such that $\alpha_1 + ... + \alpha_n = 1$. Then the Markov move q that at each step randomly selects an $i \in \{1, ..., n\}$ with probability α_i and executes q_i has transition kernel:*

$$K = \sum_{i=1}^{n} \alpha_i K_i \tag{2.58}$$

and also satisfies the detailed balance equation for p.

From this theorem, the answer to our question comes easily. We can construct multiple graphs, one for each cue, and this way have multiple types of SWC algorithms corresponding to these cues. Then the algorithms are used alternatively as in the following

Corollary 1. *Let $SW_1, ..., SW_n$ be a number of Swendsen-Wang Cuts algorithms working on the same nodes V and same posterior probability P, with adjacency graphs $G_1, ..., G_n$. Let $\alpha_1, ..., \alpha_n \geq 0$ be fixed numbers such that $\alpha_1 + ... + \alpha_n = 1$. Then the move consisting of randomly choosing an i with probability α_i and executing SW_i is reversible and ergodic.*

We can think of each SW_i as a hypothesis that is being tested in a reversible manner.

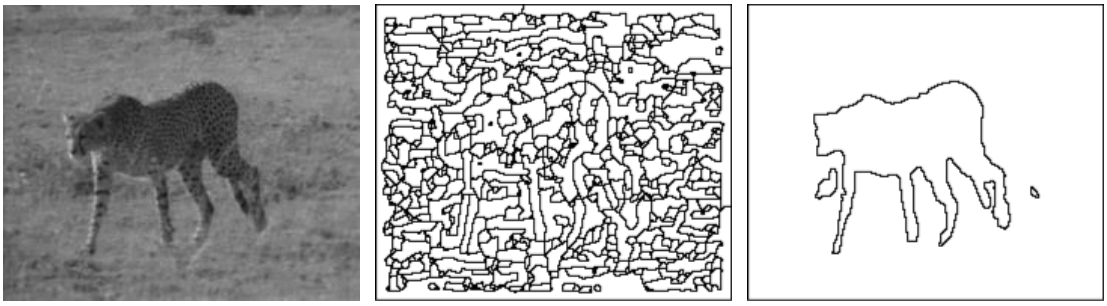
The only restriction in using the above results is that the α_i be fixed. We can still use, if possible, bottom-up information to select good values for α_i, resulting in an efficient visiting schedule of the different SW_i as long as the schedule is fixed a priori. This way we can have some hypotheses more likely than other, so they are tested more often. For each hypothesis, the algorithm will be efficient at

the places where that hypothesis is valid. By combining a good set of hypotheses, the algorithm will be efficient everywhere.

CHAPTER 3

Image Segmentation

Our first experiment tests the cluster sampling algorithm in an image segmentation task. The objective is to partition the image into a number of disjoint regions (as Figs. 2.3 and 2.5 have shown) so that each region has consistent intensity in the sense of fitting to some image models. The final result should optimize a Bayesian posterior probability $\pi(\mathbf{X}) \propto \mathcal{L}(\mathbf{I}|\mathbf{X})p_o(\mathbf{X})$.



a. input image b. atomic regions in $\mathbf{G}$ c. segmentation

Figure 3.1: Image segmentation as graph partition. a. Input image. b. Atomic regions by Canny edge detection followed by edge tracing and contour closing, each being a vertex in the graph $\mathbf{G}$. c. Segmentation result.

In this problem, $\mathbf{G}$ is an adjacency graph with vertices V being a set of atomic regions (see Figs.(2.3) and (3.1)), obtained by edge detection followed by edge tracing. Alternatively, the atomic regions could be computed using other methods, such as normalized cuts [48]. Usually $|V| = O(10^2)$. For each atomic region $v \in V$, we compute a 15-bin intensity histogram h normalized to 1. Then

the edge probability is calculated as

$$q_{ij} = p(\mu_e = \text{on}|\mathbf{I}(v_i), \mathbf{I}(v_j)) = \exp\{-\frac{1}{2}(KL(h_i||h_j) + KL(h_j||h_i))\}, \qquad (3.1)$$

where $KL()$ is the Kullback-Leibler divergence between the two histograms. Usually q_{ij} should be close to zero for $< i, j >$ crossing an object boundary. In our experiments, the edge probability leads to good clustering as Fig. 2.5 shows.

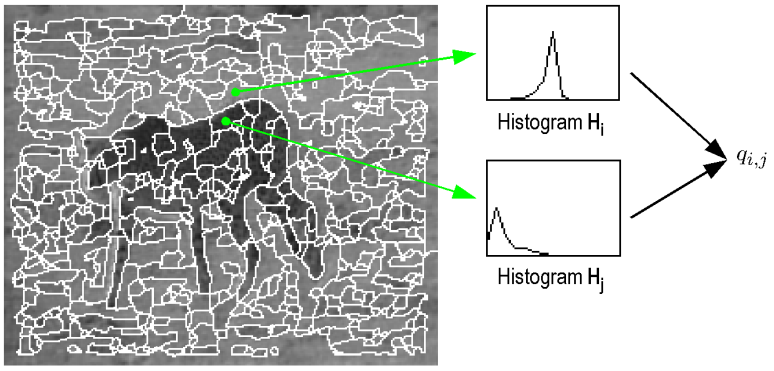


Figure 3.2: The edge weights q_{ij} are computed using the intensity histograms H_i, H_j of the atomic regions i, j.

Now we briefly define the target probability in this experiment. Let $\mathbf{X} = (V_1, ..., V_{\text{L}})$ be a coloring of the graph with the number of regions L being an unknown random variable, and the image intensities of the regions in each set V_k are consistent in the sense of fitting to a model θ_k. Different colors are assumed to be independent. Therefore, we have,

$$\pi(\mathbf{X}) = \pi(\mathbf{X}|\mathbf{I}) \propto \prod_{k=1}^{\text{L}}[\mathcal{L}(\mathbf{I}(V_k); \theta_k)p_o(\theta_k)]p_o(\mathbf{X}). \qquad (3.2)$$

We selected three types of simple models for the likelihood models to account for different image properties. The first model is a non-parametric histogram $\mathcal{H}$,

which in practice is represented by a vector of B-bins $(\mathcal{H}_1, ..., \mathcal{H}_B)$ normalized to 1. It accounts for cluttered objects, like vegetation.

$$\mathbf{I}(x, y;\, \theta_0) \sim \mathcal{H} \text{ iid},\ \forall (x, y) \in V_k. \tag{3.3}$$

The other two are regression models for the smooth change of intensities in the two-dimensional image plane (x, y), and the residues follow the empirical distribution $\mathcal{H}$ (i.e. the histogram).

$$\mathbf{I}(x, y;\, \theta_1) = \beta_0 + \beta_1 x + \beta_2 y + \mathcal{H} \text{ iid},\ \forall (x, y) \in V_k. \tag{3.4}$$

$$\mathbf{I}(x, y;\, \theta_2) = \beta_0 + \beta_1 x + \beta_2 y + \beta_3 x^2 + \beta_4 xy + \beta_5 y^2 + \mathcal{H} \text{ iid},\ \forall (x, y) \in V_k. \tag{3.5}$$

In all cases, the likelihood is expressed in terms of the entropy of the histogram $\mathcal{H}$

$$\mathcal{L}(\mathbf{I}(V_k); \theta_k) \propto \prod_{v \in V_k} \mathcal{H}(\mathbf{I}_v) = \prod_{j=1}^{B} \mathcal{H}_j^{n_j} = \exp(-|V_k| \text{entropy}(\mathcal{H})). \tag{3.6}$$

The model complexity is penalized by a prior probability $p_o(\theta_k)$ and the parameters θ in the above likelihoods are computed deterministically at each step as the best least square fit. The deterministic fitting could be replaced by the reversible jumps together with the flipping of color. This was done in [54] and is beyond the scope of our experiments.

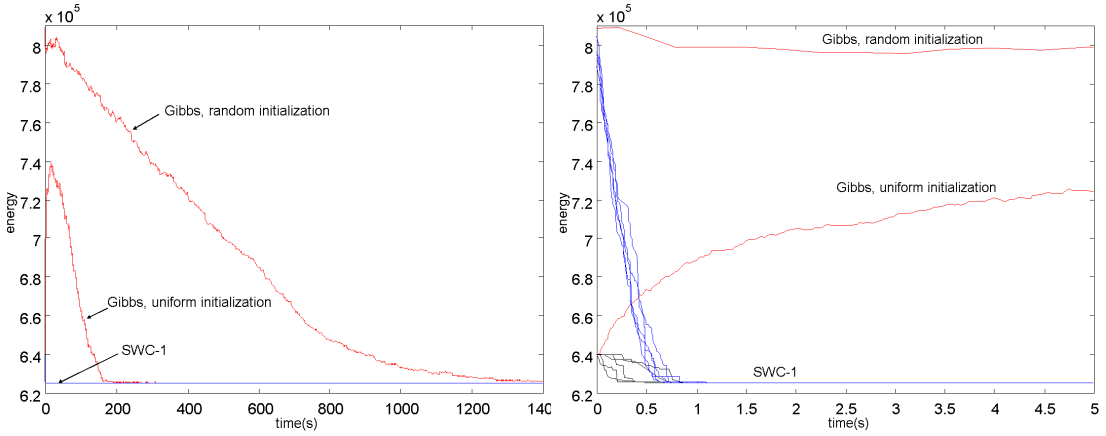
The prior model $p_o(\mathbf{X})$ encourages large and compact regions with a small number of colors, as it was suggested in [54]. Let $r_1, r_2, ..., r_m$, $m \geq \mathrm{L}$ be the connected components of all $V_k, k = 1, ..., \mathrm{L}$. Then the prior is

$$p_o(\mathbf{X}) \propto \exp\{-\alpha_0 \mathrm{L} - \alpha_1 m - \alpha_2 \sum_{k=1}^{m} \text{Area}(r_k)^{0.9}\}. \tag{3.7}$$

The last thing than needs to be designed for the SWC algorithm is the reassignment probability $q(l|R, \mathbf{X})$. We choose it again in terms of the KL divergence

between the intensity histogram of the cluster R and the intensity histogram of region V_l, as follows:

$$q(l|R,\mathbf{X}) \propto \begin{cases} 10e^{-KL(H(R),H(V_l))} & \text{if } R \text{ adjacent to } V_l \\ e^{-KL(H(R),H(V_l))} & \text{if } R \text{ not adjacent to } V_l \\ 0.1 & \text{if } l \notin L \text{ (new region)} \end{cases} \tag{3.8}$$



(a) convergence CPU time in seconds (b) Zoom-in view of the first 5 seconds.

Figure 3.3: The plot of $-\ln \pi(X)$ over computing time for both the Gibbs sampler and our algorithm for the horse image. Both algorithms are measured by the CPU time in seconds using a Pentium IV PC, so they are comparable. (a). Plot of the first $1,400$ seconds. The Gibbs sampler needs a high initial temperature and slow annealing step to achieve the same energy level. (b). The zoomed-in view of the first 5 seconds.

For the image segmentation example (horse) shown in Figs. 2.3 and 2.5, we compare the cluster sampling method with the single-site Gibbs sampler and the results are displayed in Fig. 3.3. Since our goal is to maximize the posterior probability $\pi(\mathbf{X})$, we must add an annealing scheme with a high initial temperature T_o and then decrease to a low temperature (0.05 in our experiments). We plot the

$-\ln \pi(\mathbf{X})$ over CPU time in seconds with a Pentium IV PC. The Gibbs sampler needs to raise the initial temperature high ($T_o \geq 100$)) and uses a slow annealing schedule to reach the same energy level as our algorithm. The cluster sampling method can run at low temperature. We usually raise the initial temperature to $T_o \leq 15$ and use a fast annealing scheme. Fig. 3.3.(a) plots the two algorithms at the first $1,400$ seconds, and Fig. 3.3.(b) is a zoomed-in view of the first 5 seconds.

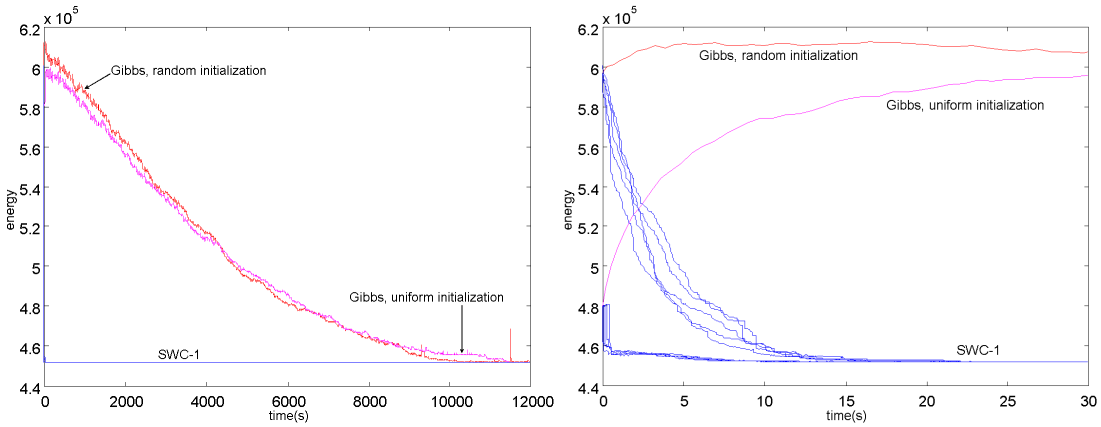


Figure 3.4: Convergence comparison between the clustering method and Gibbs sampler in CPU time (seconds) on the artificial image (circles, triangle and rectangles) in the first row of Fig.3.8. (left). The first 1,200 seconds. (right) Zoomed-in view of the first 30 seconds. The clustering algorithm is run 5 trials for both the random and uniform initializations.

We run the two algorithms with two initializations. One is a random labeling of the atomic regions and thus has higher initial energy $-\ln \pi(\mathbf{X})$, and the other initialization sets all vertices to the same color. The clustering methods are run five times on both cases. They all converged to one solution (see Fig.2.3.(c)) within 1 second, which is $O(10^2)$ times faster than the Gibbs sampler.

Fig. 3.8 shows five more images. Using the sample comparison method as in

the horse image, we plot $-\ln \pi(\mathbf{X})$ against running time in Figs. 3.4 and 3.5 for the images in the first and second row of Fig. 3.8 respectively. In experiments, we also compared the effect of the edge probabilities. The clustering algorithm are $O(100)$ times slower if we use a constant edge probability $\mu_{ij} = c \in (0, 1)$ as the original SW method does. For example the single-site Gibbs sampler is an example with $q_{ij} = 0$, $\forall\, i, j$.

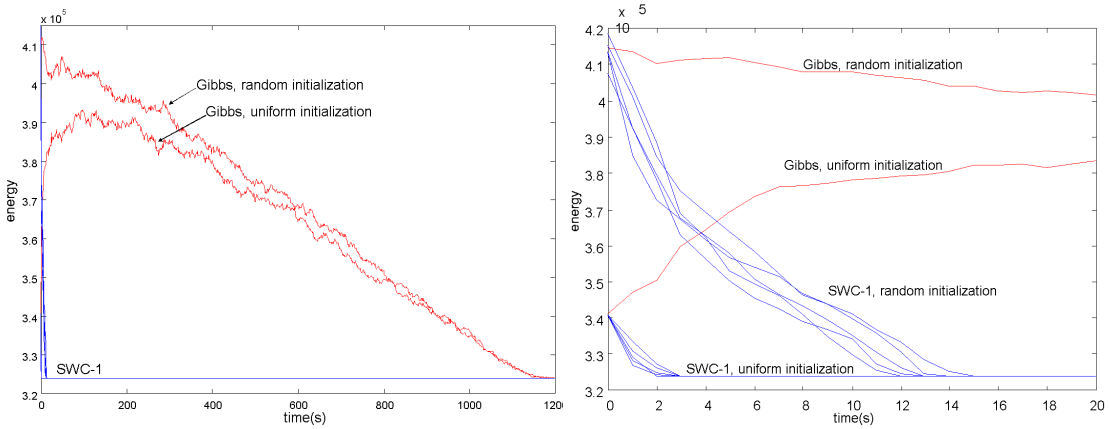


Figure 3.5: Convergence comparison between the clustering method and Gibbs sampler in CPU time (seconds) on the cheetah image. (Left) The first 1,200 seconds. (Right) Zoomed-in view of the first 20 seconds. The clustering algorithm is run 5 times for both the random and uniform initializations.

To study the effects of the discriminative probabilities q_e on convergence speed, we compare the performance of our algorithm with and without discriminative probabilities in Fig.3.6. We run the SWC-1 algorithm 3 times with all edges having the constant probability, $q_e = 0.2, 0.4, 0.6$ respectively (Note that the Gibbs sampler is equivalent to SWC with $q_e = 0$). The annealing schedules for these runs have to be slower, starting at higher temperature, to obtain the same final energy. Sometimes the algorithm cannot reach the same low energy as with discriminative models.

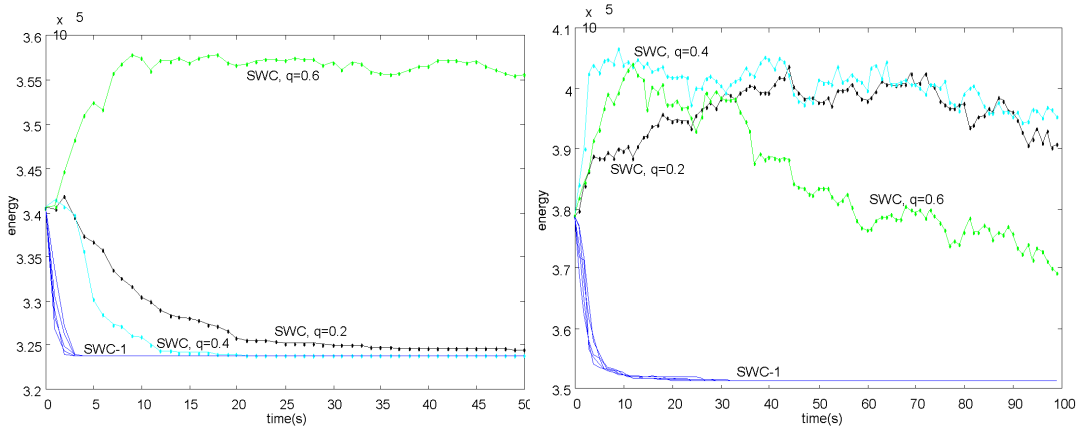


Figure 3.6: Evaluation of the effects of the discriminative probabilities q_ij. For comparison, the SWC-1 algorithm is compared with an SWC where the edge weights are fixed to values $q_ij = 0.2, 0.4, 0.6$ respectively, on the cheetah image 3.1 (left) and the airplane image from Figure 3.8 (right).

Fig. 3.6 displays the energy vs CPU time (in seconds) of the three runs and the SWC-1 on the cheetah (left) image of Figure 3.1 and airplane (right) image shown in Fig.3.8. The energies of the three SWC runs with constant edge probability $q_e = 0.2, 0.4, 0.6$ are shown in dotted lines, all three runs start from a uniform initialization. They are significantly slower than SWC-1. It is worth mentioning that these SWC runs without discriminative probabilities are not equivalent with the original SW algorithm because we work on a more general energy function, on which the original SW cannot be applied because the acceptance probability is not one.

Fig. 3.7 compares SWC-1 and SWC-3 on the second image in Figure 3.8. The plot displays the average of 100 runs of SWC-1 and SWC-3 respectively. SWC-1 is more effective than SWC-3 because of the computational overhead of each SWC-3 move, and that there is more data-driven information used in the SWC-1 than in SWC-3, existent in the design of the $q(l'|R, \mathbf{X})$.

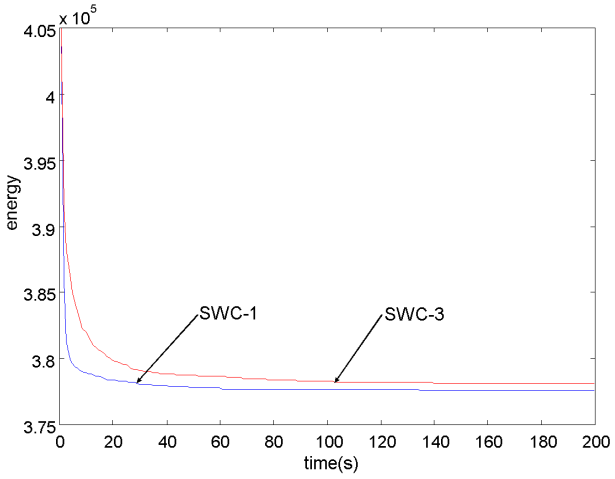


Figure 3.7: Comparison of SWC-1 and SWC-3 for the second image in Figure 3.8. Both plots are in CPU time. SWC-3 has more overhead at each step and is slower than SWC-1.

Compared with the DDMCMC algorithm from [54], our algorithm can speed it up by 20-40 times in CPU time. Our model fitting and switching steps are quite simple, but we observed that the full-featured model fitting and switching steps take much less time than the split-merge steps which are the focus of our algorithm. By incorporating full-featured model fitting and switching steps in our algorithm, it will remain 20-40 times faster than the DDMCMC[54].

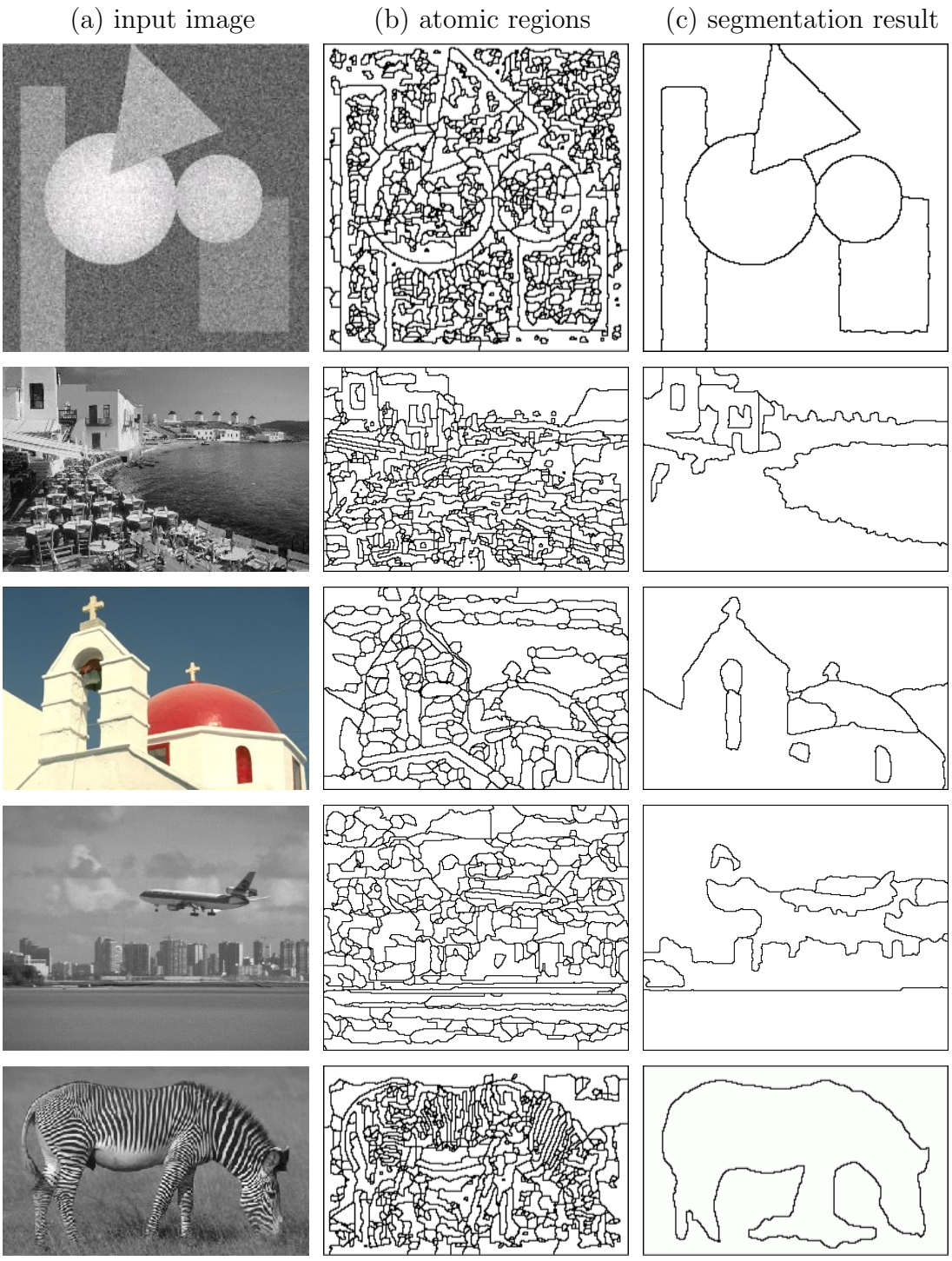


Figure 3.8: More results for image segmentation.

CHAPTER 4

Curve Grouping

In this experiment we are given an edge map with a number of n edgels. These edgels are obtained using the Canny edge detector followed by fitting long curves by many line segments. Usually we have $n \in [500 - 2000]$ short line segments (edgels of 3-6 pixels long) as vertices V in $\mathbf{G}$. We denote them by $v_i = (\mathbf{x}_i^s, \mathbf{x}_i^e), i = 1, 2, ..., N$ with $\mathbf{x}_i^s, \mathbf{x}_i^e$ being the starting and ending points.

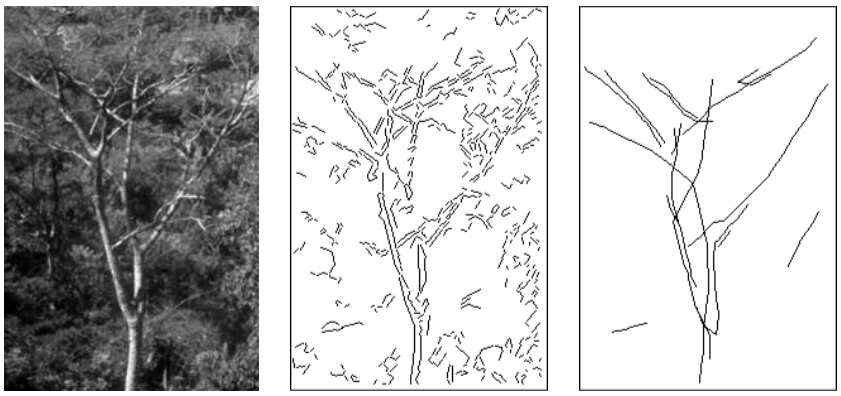


Figure 4.1: Perceptual grouping: input image, map of edgelets by Canny edge detection and a grouping result.

Our goal is to group these edgels into an unknown number n of subgraphs $V_i, i = 1, 2, ..., n$, each being a chain of edgels. By filling in the gaps between consecutive edgels in V_i we obtain a smooth and continuous curve Γ_i.

Now we choose the likelihood model. In discrete form, the edgel set V in $\mathbf{G}$ consists of pixels on the edges, denoted by

$$D^{\text{obs}} = \{(i,j) : (i,j) \text{ on } v \in V\}$$

The n continuous curves also contain a set of pixels on curves

$$D = \{(i,j) : (i,j) \text{ on } \Gamma_k, k = 1, 2, ..., n\}$$

We choose the likelihood to be

$$p(D^{\text{obs}}|W) \propto \prod_{(i,j)\in D^{obs}-D} p_0 \prod_{(i,j)\in D-D^{obs}} p_1 = e^{-\lambda_0|D^{obs}-D|-\lambda_1|D-D^{obs}|} \tag{4.1}$$

where $p_0 = e^{-\lambda_0} \in [0,1]$ is the probability for detecting a false edge, and penalizes removing too many edges. In contrast, $p_1 = e^{-\lambda_1}$ is the probability for missing an edge and penalizes the gaps in the curves.

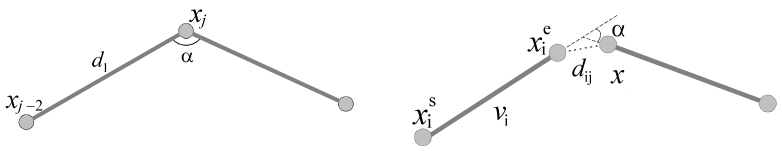
Each curve is then represented by a list of points $\Gamma_j = (\mathbf{x}_{j1}, \mathbf{x}_{j2}, ..., \mathbf{x}_{jN_j})$. The prior model for a curve group is

$$p(W) \propto \exp\{-\lambda n\} \prod_{i=1}^{n} p(\Gamma_i).$$

Each curve follows a 2nd order Markov chain model.

$$p(\Gamma_i) = p(\mathbf{x}_{i1}, \mathbf{x}_{i2}) \prod_{j=3}^{k} p(\mathbf{x}_j|\mathbf{x}_{j-1}, \mathbf{x}_{j-2}). \tag{4.2}$$

The probability $p(\mathbf{x}_{i1}, \mathbf{x}_{i2})$ is assumed uniform, while $p(\mathbf{x}_j|\mathbf{x}_{j-1}, \mathbf{x}_{j-2})$ is a two gram represented by a 2-way joint histogram. We compute it by supervised learning from a number of manually parsed images, e.g. from [36]. As Figure 4.2.a shows, we compute three variables: (1). distance $d_1 = |\mathbf{x}_{j-1} - \mathbf{x}_{j-2}|$, (2). distance $d_2 = |\mathbf{x}_j - \mathbf{x}_{j-1}|$, and (3). the angle α. There are 6 histograms, one for the



(a) (b)

Figure 4.2: (a) The joint histograms of $p(\mathbf{x}_j|\mathbf{x}_{j-1}, \mathbf{x}_{j-2})$ contain 6 bins for d_2 and 36 bins for α. There are 6 such histograms, for different values of d_1. (b) The SW edge strength between two nodes (edgels) is defined in terms of the gap and the angle between the edgels.

values of d_1 in each of the intervals $[0, 2), [2, 4), [4, 8), [8, 16), [16, 32), [32, 64]$. Each histogram has 6 bins for d_2, in the same range as d_1, and 36 bins for α, each of size 10^o. Thus we have 6 histograms with 6×36 bins each and we represent $p(\mathbf{x}_j|\mathbf{x}_{j-1}, \mathbf{x}_{j-2})$ by $p(d_2, \alpha|d_1)$. To avoid empty bins we will start with each bin having one sample in it.

This model depends on the ordering and orientation of the edgels in the curve. For each curve V_l, we resolve the ordering and orientation in a deterministic greedy way as follows.

1. we begin with the curve $V_l = \{v_1, .., v_k\}$ broken down into k curves $\gamma_i = v_i$.
2. find γ_i, γ_j and orientations such that the log likelihood ratio of the merged curve $\log p(\gamma_i \cup \gamma_j) - \log p(\gamma_i) - \log p(\gamma_j)$ is maximized.
3. merge γ_i and γ_j into $\gamma = \gamma_i \cup \gamma_j$. The numbers of curves is reduced by 1.
4. repeat steps 2,3 until the number of curves is 1.

To construct the SWC graph $\mathbf{G}$, we start with a complete graph on the edgels, and compute an edge strength for any pair $e = (v_i, v_j)$ (see Fig.4.2.b), based on the gap d_{ij} between the two edgels, and the two gram learned for the prior

$$q_e = 0.99 \cdot p(x_j^s | x_i^e, x_i^s) \cdot p(x_j^e | x_j^s, x_i^e) \cdot e^{-\lambda_1 * d_{ij}} \tag{4.3}$$

where λ_1 is the gap penalty used in the likelihood equation.

If $q_e < 0.01$ then edge e is removed. We assume this is a very safe threshold to reduce the graph complexity.

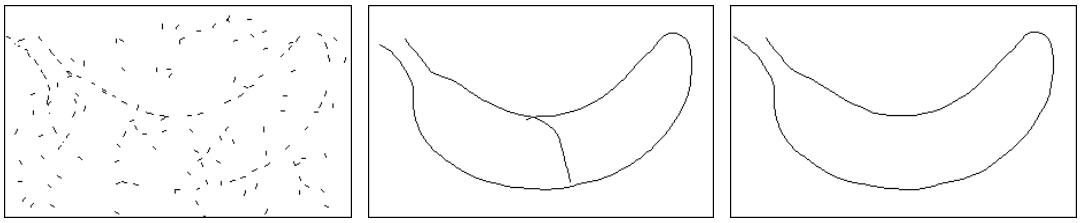


Figure 4.3: Curve grouping: input edgel map and two grouping results.

We display four examples obtained with SWC-1 in Fig. 4.1, 4.3, 4.4. The results are not ideal, mainly because of the simple curve model that we used. In future work, we should introduce more advanced curve models. In fact, most recently, the SWC method was applied to grouping parallel curves and trees and more advanced results are in a paper [55].

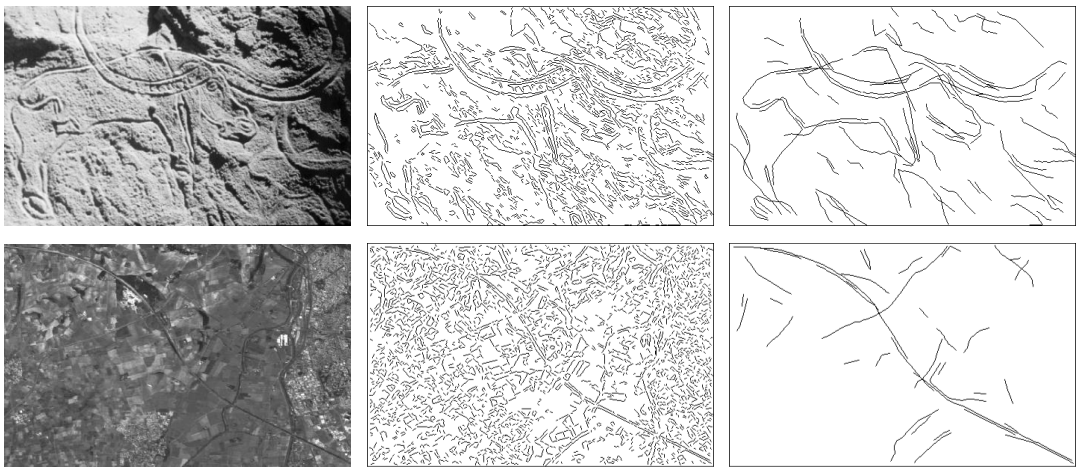


Figure 4.4: Curve grouping: input image, edgel map and grouping result.

CHAPTER 5

Hierarchic image-motion segmentation

5.1 Multi-grid and Multi-level cluster sampling

When the graph size $\mathbf{G}$ is big, for example, $|V| = O(10^4) \sim O(10^6)$ in image analysis, a clustering step has to flip many edges and is costly computationally. This section presents two strategies for improving the speed – the multi-grid and multi-level cluster sampling. Our methods are different from the multi-grid and multi-level samplings ideas in the statistical literature (see Gilks [19] and Liu [35]).

5.1.1 Rationale for multi-grid and multi-level cluster sampling

In multi-grid clustering sampling, we introduce an "attention window" Λ (see Figure reffig:multigrid) which may change location and size over time. The cluster sampling is limited to within the window at each step, and this is equivalent to sampling a conditional probability,

$$\mathbf{X}_\Lambda \sim \pi(\mathbf{X}_\Lambda | \mathbf{X}_{\bar{\Lambda}}). \tag{5.1}$$

The multi-level cluster sampling is motivated by the problem of hierarchic graph labeling. Figure 5.1 illustrates an example in motion segmentation. Suppose we are given two consecutive image frames in a video, and our goal consists of three parts: (i) calculate the planar velocity (i.e. optical flow) of the pixels

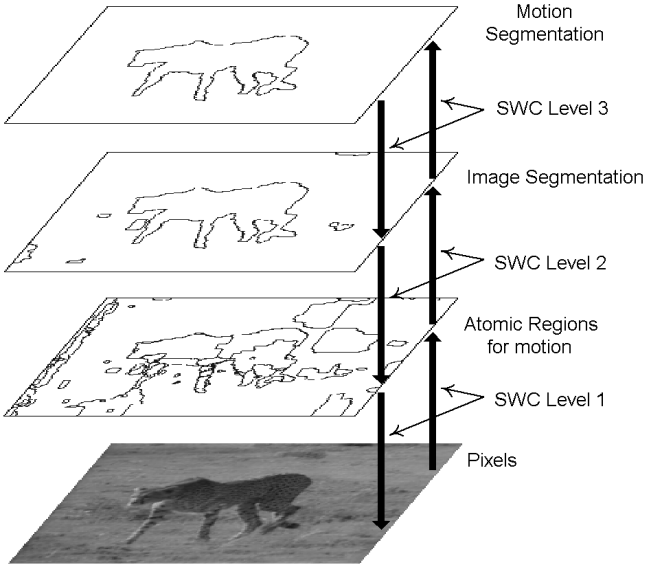


Figure 5.1: Cluster sampling on multi-level of graphs for motion segmentation. A connected component with the same color is frozen and collapsed into a single vertex in the level above.

in the second frame based on the displacement between pixels in two frames, (ii) segment (group) the pixels into regions of coherent intensities, and (iii) further group the regions into moving objects, such as the running cheetah and the grass background where each object should have both consistent intensity and motion velocity in the image plane.

This problem can be represented in a three-level labeling

$$\mathbf{X} = (\mathbf{X}^{(0)}, \mathbf{X}^{(1)}, \mathbf{X}^{(2)}). \tag{5.2}$$

and this label forms three levels of graph shown in Figure 5.1,

$$\{\mathbf{G}^{(s)} =< V^{(s)}, E^{(s)} > : \ s = 0, 1, 2\}. \tag{5.3}$$

$\mathbf{G}^{(0)}$ is the image lattice with each vertex being a pixel. The pixels are labeled by $\mathbf{X}^{(0)}$ according to their intensity and planar motion velocity, and thus grouped into a number of small regions of nearly constant intensity and velocity in $\mathbf{G}^{(1)}$.

The vertices in $\mathbf{G}^{(1)}$ are further labeled by $\mathbf{X}^{(1)}$ according to their intensities and grouped into a smaller graph $\mathbf{G}^{(2)}$, which is in turn labeled by $\mathbf{X}^{(2)}$. The vertice size has been reduced from $O(10^5)$ in $\mathbf{G}^{(0)}$ to $O(10^2)$ in $\mathbf{G}^{(1)}$ and to $O(10)$ in $\mathbf{G}^{(2)}$.

We should discuss more details in the next two subsections. In the rest of this subsection, we discuss the theoretical justifications for the multi-grid and multi-level cluster sampling.

The essence of the cluster sampling design is that its Markov chain kernel observes the detailed balance equations as a result of the Metropolis-Hastings design.

$$\pi(\mathbf{X})\mathcal{K}(\mathbf{X},\mathbf{Y}) = \pi(\mathbf{Y})\mathcal{K}(\mathbf{Y},\mathbf{X}),\ \forall \mathbf{X},\mathbf{Y}. \tag{5.4}$$

The detailed balance equation is a sufficient condition for $\mathcal{K}$ to satisfy the invariant condition,

$$\sum_{\mathbf{X}} \pi(\mathbf{X})\mathcal{K}(\mathbf{X},\mathbf{Y}) = \pi(\mathbf{Y}),\ \forall \mathbf{Y}. \tag{5.5}$$

In practice, one may design a set of Markov chain kernels, each corresponding to specific MCMC dynamics,

$$\Delta = \{\mathcal{K}_a, a \in \mathcal{A}\}, \tag{5.6}$$

The overall Markov chain kernel is a mixture of these dynamics with probability q_a,

$$\mathcal{K}(\mathbf{X},\mathbf{Y}) = \sum_{a\in\mathcal{A}} q_a \mathcal{K}_a(\mathbf{X},\mathbf{Y}),\ \ \forall \mathbf{X},\mathbf{Y}. \tag{5.7}$$

There are two basic design criteria for Δ, which are easily observed in the finite state space.

1. The Kernels in Δ are ergodic so that for any two points $\mathbf{X}$ and $\mathbf{Y}$, there is a path of finite length $(\mathbf{X}, \mathbf{X}_1, ..., \mathbf{X}_N, \mathbf{Y})$ between $\mathbf{X}$ and $\mathbf{Y}$ consisting of

the $N+1$ kernels $k(0), ..., k(N) \in \Delta$, with

$$\mathcal{K}_{k(0)}(\mathbf{X}, \mathbf{X}_1) \cdot \mathcal{K}_{k(1)}(\mathbf{X}_1, \mathbf{X}_2) \cdots \mathcal{K}_{k(N)}(\mathbf{X}_N, \mathbf{Y}) > 0.$$

2. Each sub-kernel observes the detailed balance equations, and thus the overall kernel satisfies them.

The multigrid and multi-level design in the next two subsections are ways for designing the sub-kernels that observe the detailed balance equations.

5.1.2 Multigrid cluster sampling

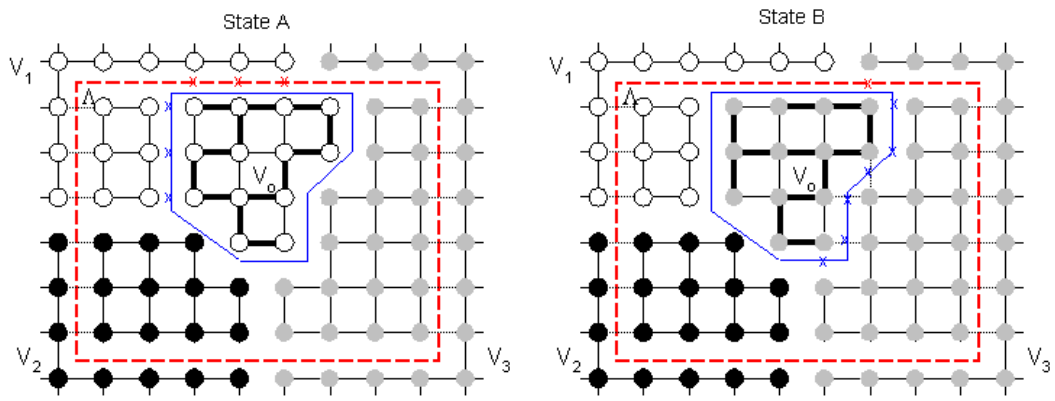


Figure 5.2: Multigrid flipping: computation is restricted to different "attention" windows Λ of various sizes, with the rest of the labels fixed.

Let Λ be an "attention window" on graph $\mathbf{G}$, and $\mathbf{X} = (V_1, V_2, ..., V_{\mathrm{L}})$ the current labeling state. Λ divides the vertices into two parts,

$$V = V_\Lambda \ \cup \ V_{\bar{\Lambda}}, \ \text{ and } \ \mathbf{X} = (\mathbf{X}_\Lambda, \mathbf{X}_{\bar{\lambda}}). \tag{5.8}$$

For example, Figure reffig:multigrid displays a rectangular window Λ (in red dashed) in a lattice $\mathbf{G}$. The window Λ cuts some edges within each subset $V_k, k =$

$1, 2, ..., \mathrm{L}$, and we denote them by,

$$\mathcal{C}(V_k, \Lambda) = \{< s, t >: s \in V_k \cap V_\Lambda,\ t \in V_k \cap V_{\bar{\Lambda}}\}.$$

In Figure 5.2 the window Λ intersects with three subsets V_1 (white), V_2 (black), and V_3 (grey), and all edges crossing the (red) rectangle window are cut.

Multi-grid Swendsen-Wang Cuts

1. Select an attention window $\Lambda \subset \mathbf{G}$.
2. Cluster the vertices within Λ and select connected component R.
3. Flip the label of R.
4. Accept the flipping with probability, using $\mathbf{X}_{\bar{\Lambda}}$ as boundary condition.

Following the proof of Theorem 4 in Section (2.2), we can derive the acceptance proposal probability to move from state $\mathbf{X}_A$ to state $\mathbf{X}_B$ within Λ.

Proposition 7. *The acceptance probability for proposing R as a candidate cluster within window Λ and moving from state $\mathbf{X}$ to state $\mathbf{X}'$ is*

$$\alpha(\mathbf{X} \to \mathbf{X}') = \min\{1, \frac{\prod_{<i,j> \in \mathcal{C}(R, V_{\ell'}) - \mathcal{C}(V_{\ell'}, \Lambda)} (1 - q_{ij})}{\prod_{<i,j> \in \mathcal{C}(R, V_\ell) - \mathcal{C}(V_\ell, \Lambda)} (1 - q_{ij})} \cdot \frac{q(\mathbf{X}_R = \ell | R, \mathbf{X}')}{q(\mathbf{X}_R = \ell' | R, \mathbf{X})} \cdot \frac{\pi(\mathbf{X}')}{\pi(\mathbf{X})}\}.$$

In Figure 5.2), we have $\mathbf{X} = \mathbf{X}_A$ and $\mathbf{X}' = \mathbf{X}_B$ $(\ell = 1, \ell' = 3)$.

The difference between this ratio and the ratio in Theorem 4 is that some edges in $\mathcal{C}(V_\ell, \Lambda) \cup \mathcal{C}(V_{\ell'}, \Lambda)$ no longer participate in the computation.

Proposition 8. *The Markov chain simulated by the multi-grid scheme has invariant probability $\pi(\mathbf{X}_\Lambda | \mathbf{X}_{\bar{\Lambda}})$ and its kernel $\mathcal{K}$ observes the detailed balance equation,*

$$\pi(\mathbf{X}_\Lambda | \mathbf{X}_{\bar{\Lambda}}) \mathcal{K}(\mathbf{X}_\Lambda, \mathbf{Y}_\Lambda) = \pi(\mathbf{Y}_\Lambda | \mathbf{X}_{\bar{\Lambda}}) \mathcal{K}(\mathbf{Y}_\Lambda, \mathbf{X}_\Lambda). \tag{5.9}$$

The proof is straightforward.

We will now prove that the multi-grid method is also invariant to teh full posterior probability. Let $p(x, y)$ be a two dimensional probability, and $\mathcal{K}$ be a Markov kernel sampling its conditional probability $p(x|y)$ (or $p(y|x)$). Thus it observes the detailed balance equations,

$$p(x|y)\mathcal{K}(x, x') \quad = \quad p(x'|y)\mathcal{K}_1(x', x), \ \forall x, x'. \tag{5.10}$$

Theorem 7. *In the above notation, $\mathcal{K}$ observes the general detailed balance equations after augmenting y*

$$p(x, y)\mathcal{K}((x, y), (x', y')) = p(x', y')\mathcal{K}((x', y'), (x, y)).$$

Proof. If $y = y'$, then it is straightforward. If $y \neq y'$ then $\mathcal{K}((x, y), (x', y')) = \mathcal{K}((x', y'), (x, y)) = 0$ because there is no way to go from state (x, y) to state (x', y'). □

The conclusion of this theorem is that an algorithm which is reversible when sampling from a conditional probability is also reversible for sampling the full probability. From here we obtain

Proposition 9. *Let $\pi(\mathbf{X})$ be a target probability defined on a graph $\mathbf{G} =< V, E >$ and $\Lambda \subset V$ a window. Then the kernel $\mathcal{K}_\Lambda$ of the multi-grid scheme for Λ observes the detailed balance equation with respect to $\pi(\mathbf{X})$,*

$$\pi(\mathbf{X})\mathcal{K}(\mathbf{X}, \mathbf{Y}) = \pi(\mathbf{Y})\mathcal{K}(\mathbf{Y}, \mathbf{X}). \tag{5.11}$$

5.1.3 Multi-level cluster sampling

Following the notations in Section (5.1.1), the problem is hierarchic labeling with $\mathbf{G} = (\mathbf{G}^{(0)}, \mathbf{G}^{(1)}, \mathbf{G}^{(2)})$ and $\mathbf{X} = (\mathbf{X}^{(0)}, \mathbf{X}^{(1)}, \mathbf{X}^{(2)})$. Each level of labeling $\mathbf{X}^{(s)}$ is

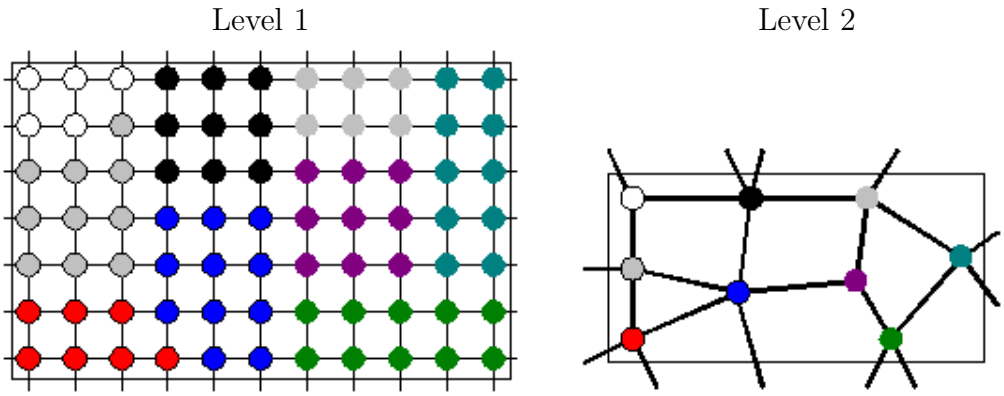


Figure 5.3: Multi-level cluster sampling. Computation is performed at different levels of granularity, where the connected components from the lower level collapse into vertices in the higher level.

equivalent to a partition of the lattice with connected components.

$$\mathrm{CP}(\mathbf{X}^{(s)}) = \{\mathrm{cp}_1^{(s)}, \mathrm{cp}_2^{(s)}, ..., \mathrm{cp}_{m^{(s)}}^{(s)}\},\ s = 0, 1, 2. \tag{5.12}$$

Note that vertices in each connected component have the same label and two disconnected components may share the same label.

Definition 2. *The hierarchic labels* $\mathbf{X} = (\mathbf{X}^{(0)}, \mathbf{X}^{(1)}, \mathbf{X}^{(2)})$ *are said to be* "nested" *if*

$$\forall \mathrm{cp}^{(s)} \in \mathrm{CP}(\mathbf{X}^{(s)}),\ \exists \mathrm{cp}^{(s+1)} \in \mathrm{CP}(\mathbf{X}^{(s+1)}) \text{ so that } \mathrm{cp}^{(s)} \subset \mathrm{cp}^{(s+1)}, \quad s = 0, 1.$$

A nested $\mathbf{X}$ has a tree structure for the levels of labels. A vertex in level $s+1$ has a number of children vertices in level s.

Multi-level Swendsen-Wang Cuts

1. Select a level s, usually in an increasing order.
2. Cluster the vertices in $\mathbf{G}^{(s)}$ and select a connected component R.

3. Flip the labeling of R.
4. Accept the flipping with probability, using the lower levels (denoted by $\mathbf{X}^{(<s)}$) as boundary condition.

Proposition 10. *Let $\pi(\mathbf{X})$ be a probability on the nested labeling $\mathbf{X} = (\mathbf{X}^{(0)}, \mathbf{X}^{(1)}, \mathbf{X}^{(2)})$, and let the kernels of the cluster sampling algorithm on the three levels of graphs be $\mathcal{K}^{(0)}$, $\mathcal{K}^{(1)}$ and $\mathcal{K}^{(2)}$ respectively. Then they observe the detailed balance equations with respect to the conditional probabilities,*

$$\pi(\mathbf{X}^{(s)}|\mathbf{X}^{(<s)})\mathcal{K}^{(s)}(\mathbf{X}^{(s)}, \mathbf{Y}^{(s)}) = \pi(\mathbf{Y}^{(s)}|\mathbf{X}^{(<s)})\mathcal{K}^{(s)}(\mathbf{Y}^{(s)}, \mathbf{X}^{(s)}),\ s = 0, 1, 2. \tag{5.13}$$

where $\mathbf{X}^{(<s)} = (\mathbf{X}^0, ..., \mathbf{X}^{s-1})$. Therefore the multi-level SWC observes the detailed balance equation (5.4).

5.2 Hierarchic motion segmentation

Now we report the experiments on motion analysis using multi-grid and multi-level SWC.

Let $\mathbf{I}_1, \mathbf{I}_2$ be two consecutive image frames in a video sequence, as Figure 5.4 illustrates. The images $\mathbf{I}_1$ and $\mathbf{I}_2$ are discretized into lattices Λ_1 and Λ_2 respectively. Due to motion occlusion, some points are visible in only one image, say the white areas ϕ_1 in $\mathbf{I}_1$ and ϕ_2 in $\mathbf{I}_2$, and are called "half-occluded" points. All other points, named $\rho_1 = \bar{\phi}_1 = \Lambda_1 \setminus \phi_1$ and $\rho_2 = \bar{\phi}_2 = \Lambda_2 \setminus \phi_2$ respectively, can be mapped between the two image frames. The mapping function is called the "optical flow" field,

$$(u, v) : \rho_2 \mapsto \rho_1 \tag{5.14}$$

For any point (x, y) in the first frame, $(u(x, y), v(x, y))$ is the displacement for the planar motion velocity. Usually one can assume that the intensity of a point

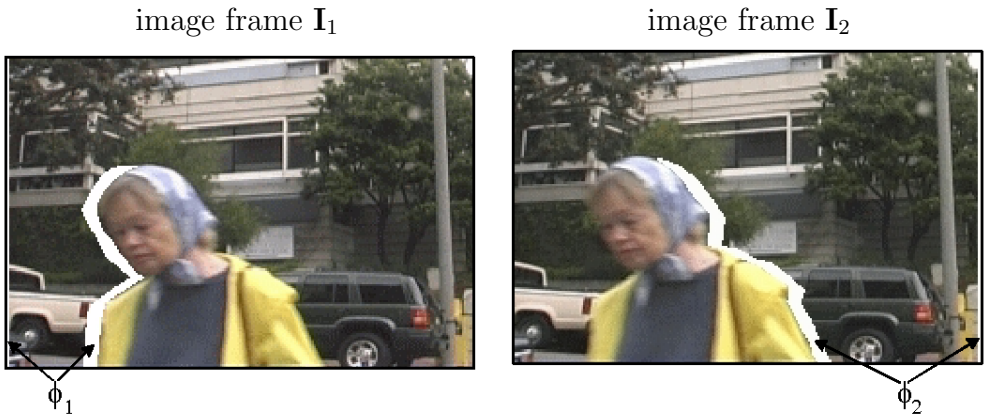


Figure 5.4: Two consecutive image frames with two moving objects, foreground and background respectively. The pixels in the areas ϕ_1 are not seen in $\mathbf{I}_2$ and reversely, the pixels in ϕ_2 are not seen in image $\mathbf{I}_1$; they are called "half-occluded" pixels. The other pixels can be mapped between the two frames.

will be constant (with stable illumination and Lambertian surfaces) between the two frames, and the residue is modeled by Gaussian noise $\mathbf{n} \sim \text{Gaussian}(0, \sigma_o^2)$. Let's take the second image as the reference frame,

$$\mathbf{I}_2(x, y) = \mathbf{I}_1(x - u(x, y), y - v(x, y)) + \mathbf{n}(x, y), \quad \forall\, (x, y) \in \rho_2. \tag{5.15}$$

In the motion analysis problem, we consider discrete pixels in the second image frame $\mathbf{G}^{(0)} = \Lambda_2$, and each pixel has three labels $x = (x^{(0)}, x^{(1)}, x^{(2)})$:

1. The velocity $x^{(0)} = (u, v)$ is discretized into $21 \times 9 = 189$ different planar velocities. We assume the maximum displacement in the lattice between two consecutive frames to be $-5 \leq u \leq 5, -2 \leq v \leq 2$ with $1/2$ pixel precision. That leads to 189 possible planar velocities. Then for pixels which do not have corresponding pixels in the first frame, i.e. pixels in ϕ_2, their velocities cannot be decided and are denoted by nil. They can be estimated based on context information on their intensity through image segmentation. Thus we have $x^{(0)} \in \{\text{nil}, 1, 2, ..., 189\}$ as its velocity label.

2. The intensity label $x^{(1)} \in \{1, 2, ..., \mathrm{L}^{(1)}\}$ for image segmentation. That is, the image lattice is partitioned into a number of regions with coherent intensities in terms of fitting to the three families of image models in Chapter 3.

3. The object label $x^{(2)} \in \{1, 2, ..., \mathrm{L}^{(2)}\}$. That is, the image lattice is partitioned into a number of $\mathrm{L}^{(2)}$ objects which have coherent intensity and motion.

To fix notation, we divide the image frames into two parts,

$$\mathbf{I}_1 = (\mathbf{I}_{1,\phi_1}, \mathbf{I}_{1,\bar{\phi}_1}), \quad \mathbf{I}_2 = (\mathbf{I}_{2,\phi_2}, \mathbf{I}_{2,\bar{\phi}_2})$$

The target probability is the Bayesian posterior,

$$\pi(\mathbf{X}) = \pi(\mathbf{X}^{(0)}, \mathbf{X}^{(1)}, \mathbf{X}^{(2)} | \mathbf{I}_1, \mathbf{I}_2) \propto \mathcal{L}(\mathbf{I}_{1,\bar{\phi}_1} | \mathbf{I}_{2,\bar{\phi}_2}, \mathbf{X}^{(0)}) \mathcal{L}(\mathbf{I}_2 | \mathbf{X}^{(1)}) \pi_o(\mathbf{X}). \quad (5.16)$$

The first likelihood is specified by the optical flow model,

$$\mathcal{L}(\mathbf{I}_{1,\bar{\phi}_1} | \mathbf{I}_{2,\bar{\phi}_2}, \mathbf{X}^{(0)}) = \prod_{(x,y) \in \bar{\phi}_2} \frac{1}{\sqrt{2\pi}\sigma_o} \exp\{-\frac{1}{2\sigma_o}(\mathbf{I}_2(x, y) - \mathbf{I}_1(x - u(x, y), y - v(x, y)))^2\}. \quad (5.17)$$

The second likelihood is the same as the image segmentation likelihood in Chapter 3.

The prior probability assumes piecewise coherent motion. That is, each moving object $o = 1, 2, ..., \mathrm{L}^{(2)}$ has a constant planar velocity $c_o \in \{1, 2, ..., 189\}$ plus a Markov model for the adjacent velocities. Also each object (and region) has a compact boundary.

$$\pi_o(\mathbf{X}) \propto \prod_{o=1}^{\mathrm{L}^{(2)}} \exp\{-\alpha \sum_{v,x^{(2)}(v)=o} |x^{(0)}(v) - c_o|^2 - \beta \sum_{v' \in \partial v} |x^{(0)}(v') - x^{(0)}(v)|\}$$
$$\prod_{l=1}^{\mathrm{L}^{(1)}} \exp\{-\gamma |\partial V_l^{(1)}|\} \prod_{i=1}^{\mathrm{L}^{(0)}} \exp\{-\delta |\partial V_i^{(0)}|\} \exp\{-\lambda_0 \mathrm{L}^{(0)} - \lambda_1 \mathrm{L}^{(1)} - \lambda_2 \mathrm{L}^{(2)}\} \tag{5.18}$$

Now we define the edge probability at the three levels of graph for the auxiliary variables.

At level $\mathbf{X}^{(0)}$, let (x, y) and (x', y') be two adjacent pixels, and (u, v) the common motion velocity of both pixels, The edge probability is defined as

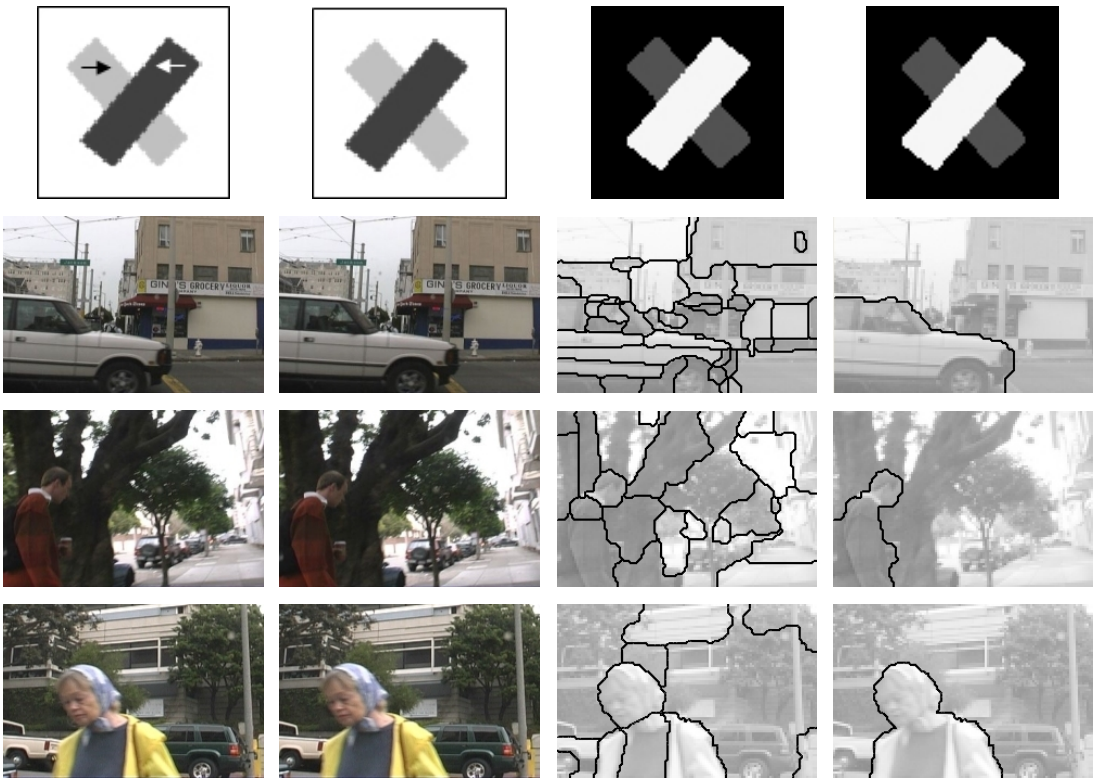
$$\begin{aligned} q^{(0)}(v, v') &= \min_{(u,v)} e^{-[|\mathbf{I}_2(x,y) - \mathbf{I}_1(x-u,y-v)| + |\mathbf{I}_2(x',y') - \mathbf{I}_1(x'-u,y'-v)|]/7} \\ &= -|\mathbf{I}_2(x, y) - \mathbf{I}_2(x', y')|/10\}. \end{aligned}$$

At the region level $\mathbf{X}^{(1)}$, the edge weights between two adjacent nodes v, v' (each being a set of pixels) are based on the KL divergence between their intensity histograms h_u, h_v, as in Chapter 3.

At the object level $\mathbf{X}^{(2)}$, the edge weights between two adjacent nodes v, v' (each being a set of pixels) are based on the KL divergence between their motion histograms $h_m(v), h_m(v')$. We maintain the histogram of the motion velocities in each object.

$$q^{(2)}(v, v') = \exp\{-\frac{1}{2}(KL(h_m(v)||h_m(v')) + KL(h_m(v')||h_m(v))\}. \tag{5.19}$$

We run the multi-grid and multi-level SW-cut on a number of synthetic and real world motion images. We show four results in Figure 5.5. The first image shows two moving rectangles where only the 8 corners provide reliable local velocity (aperture problem) and the image segmentation is instrumental in deriving



frame $\mathbf{I}_1$ frame $\mathbf{I}_2$ image segmentation motion segmentation

Figure 5.5: Hierarchical motion analysis. From left to right: first frame $\mathbf{I}_1$, second frame $\mathbf{I}_2$, image segmentation, motion segmentation. The image segmentation is the result at level $s = 1$ and the motion segmentation is the result at level $s = 2$. For the color images (the 3rd and 4th rows) we treated the three R,G, B color bands each as a grey image.

the right result. For the other three sequences, the algorithm obtains satisfactory results despite large motion and complex background. The cheetah image in Figure 5.1 is a fifth example.

For comparison of the different cluster sampling methods, we choose the image segmentation example – the cheetah image in Figure 3.8.

In Chapter 3, the pixels are grouped deterministically into atomic regions in

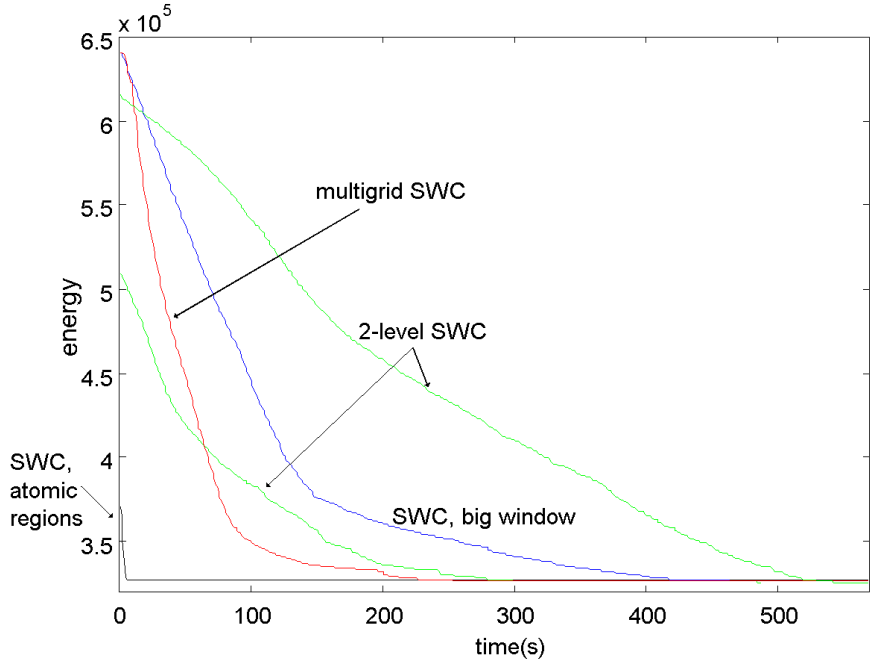


Figure 5.6: Convergence comparison of multigrid and multi-level SWC for the cheetah image in Figure 3.8. (see text for explanation)

a pre-processing stage. Now we perform the cluster sampling on two levels, the atomic regions being generated by the first level of the cluster sampling process, while the second level groups the atomic regions into intensity regions.

We plot in Figure 5.6 the $-\ln \pi(\mathbf{X})$ vs the CPU time for various methods. This figure should be compared with Figure 3.5. The multi-level cluster sampling was run in two initializations.

Firstly, the two level cluster sampling is much slower than the the one level clustering. The latter assumed deterministic atomic regions. But the two level cluster sampling can reach a deeper minimum as it has more flexibility in forming the atomic regions.

Secondly, the multi-grid method is the fastest among the methods that work directly on pixels.

Thirdly, the Gibbs sampler plotted in Figure 3.5 run on the deterministic atomic regions not the pixels. If it is running on the pixels, we cannot get it converge to the minimum in any reasonable amount of time.

CHAPTER 6

Stereo Matching

6.1 Comparison of SWC with Graph Cuts and Belief Propagation for stereo

Tappen [53] compared the performance of the Graph Cuts [9] and Belief Propagation [62] algorithms on stereo matching, using a model that can be used by both algorithms, model that has been presented in [46]. Using the same model, in this section we compare the performance of the SW Cuts with Graph Cuts and Loopy Belief Propagation.

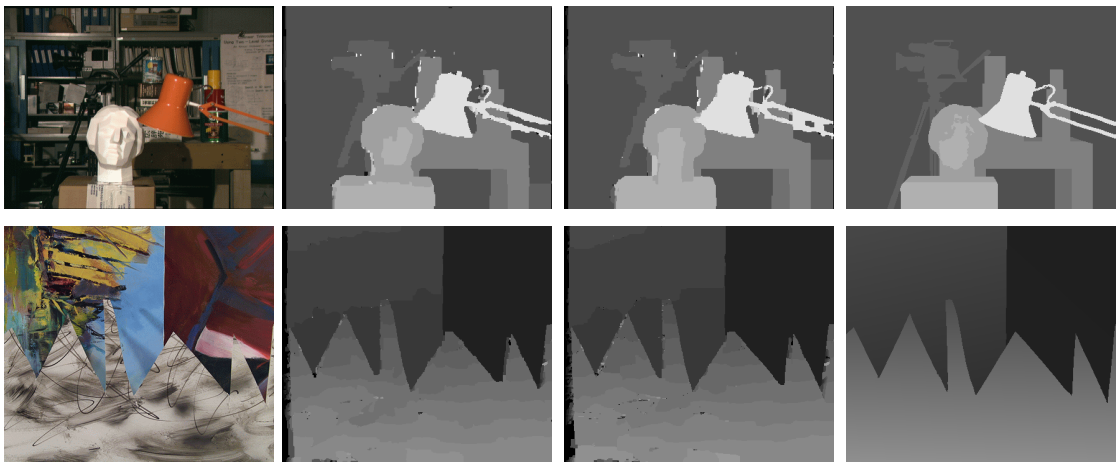
Given a pair of stereo images $\mathbf{I} = (\mathbf{I}_l, \mathbf{I}_r)$, we assign an integer disparity value (as color) $\mathbf{c}_v = d_v$ for every pixel v in the left image. The adjacency graph G_o is simply the lattice with 4-nearest neighbor connections. The energy used in the benchmark[46, 53] is a Potts model with external field,

$$\mathcal{E} = \sum_v D(d_v, v) + \sum_{<s,t>} \beta_{s,t} \mathbf{1}(d_s \neq d_t) \tag{6.1}$$

The external field (data) term measures the goodness of intensity match between the left and right images for a disparity d_v using the Birchfield-Tomasi matching cost [4],

$$D(d_v, v) = \min\{\min_{d_v - 1/2 \leq x \leq d_v + 1/2} |\mathbf{I}_l(v) - \mathbf{I}_r(v - x)|, 50\} \tag{6.2}$$

The coefficient in the prior term is made to be dependent on $< s, t >$ (inhomogeneous Potts model) $\beta_{s,t} = 20$ if $|\mathbf{I}_l(s) - \mathbf{I}_l(t)| > 8$, otherwise $\beta_{s,t} = 40$. This



a. Left image b. SWC-2 result c. Graph cuts result d. Manual (truth)

Figure 6.1: Stereo matching for the Tsukuba sequence (first row) and the Sawtooth sequence (second row).

energy has some shortcomings. (i). It is a first order Markov random field capable only of properly representing fronto-parallel planes. For example, the slanted planes in Fig. 6.1 (second row) are broken into many pieces. (ii) It does not treat half-occluded pixels explicitly and because of this, the ground truth has a much higher energy than the output of the algorithms (see Fig.6.2). We are forced to use this energy in order to compare with the graph cuts (and BP) as this is the type of energy that they can minimize. We compare the SWC-2 with the Graph Cuts implementations provided in the Scharstein and Szelisky's package [46] and Tappen's extension to Belief Propagation [53] available online.

For the stereo problem, we define discriminative probabilities on both vertices and edges to get better empirical results.

On each vertex (pixel) $v \in V$ we compute the vertex probability $q(d_v, v) \propto e^{-D(d_v,v)}$ normalized to 1 for $d_v \in \{0, ..., d_{\max}\}$. It measures how likely pixel v has disparity d_v based on local information. We compute a marginal probability

$q(d) = \frac{1}{|V|}\sum_v q(d,v)$ for each disparity level d.

For each edge $e =< s,t >$, we define an edge probability for any $d \in \{0,...,d_{\max}\}$,

$$q_e^d = 1 - e^{-\frac{20\beta_{s,t}}{3(D(s,d_s)+D(t,d_t))+10}}. \tag{6.3}$$

Thus we have $d_{\max}+1$ probabilities on each edge e, one for each disparity level. At each SWC-2 step, we first choose a disparity level d with probability $q(d)$, and then we use q_e^d as the edge probability for clustering the connected component R.

We found that most of the energy costs are contributed by the boundary pixels (due to the lack of half-occlusion treatment). Therefore, in SWC-2, a seed vertex v is chosen with equal probability either from the boundary pixels or by sampling from a goodness of fit probability $q(d_v,v)D(d_v,v)$ with d_v being the current assigned disparity at v. That is, we wish to choose more often those pixels v whose assigned disparity level d_v have a lower probability. Then we grow the component R as in SWC-2 from the seed v and propose to flip its label. The new disparity level d (or color) for R is chosen according to a probability

$$q(d|R,\pi) \propto e^{-\sum_{v\in R} D(d,v) - 0.7K\sum_{<s,t>,s\in R}\beta_{s,t}\mathbf{1}(d\neq d_t)}. \tag{6.4}$$

Fig. 6.2 compares the energy curves against CPU time in seconds for the SWC (two runs with different annealing schedules), graph cuts[9], and belief propagation (synchronous and accelerated)[53] We initialized the system with an SWC-1 version working on atomic regions which decreased the energy from about $5,000,000$ to about $650,000$ in less than 30 seconds (included in the plot). Then the SWC-2 version working on the pixel lattice provided the final result. The final energy obtained with SWC-2 was within 1% of the final energy of the Graph Cuts algorithm for the Tsukuba sequence and within less then 2% for the other sequences. All parameters were kept the same in all experiments.

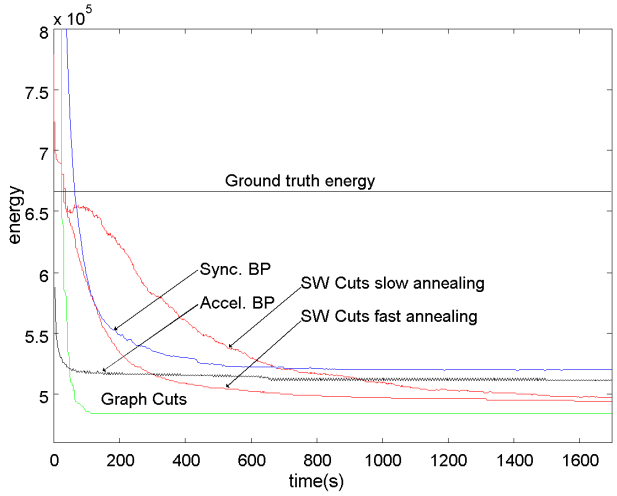
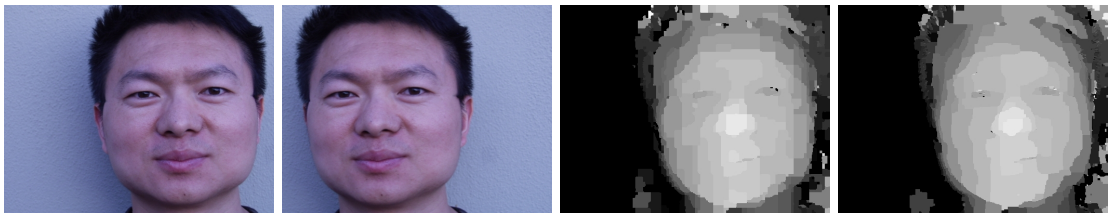


Figure 6.2: Performance comparison of SWC with Graph Cuts and Belief Propagation for the Tsukuba sequence. The curves plot the energy over CPU time in seconds.

The energy level is not a good indicator of the quality of results as the ground truth results have higher energy than all algorithms. The experiments show that the SWC reaches lower energy than belief propagation but it is slower than Graph cuts.



a. Left image b. Right image c. SWC result d. Graph cut result

Figure 6.3: The simple Potts model does not allow a faithful reconstruction of a face.

The Potts model is not suitable for 3d reconstruction of free-form surfaces such as faces, as shown in Figure 6.3. Models with second order priors and sub-pixel accuracy, computed using PDE methods [13] are better choices in this case.

6.2 SWC with generative models for stereo

In this section, we release ourselves from the simple energy model in eqn.(6.1), and adopt generative models with piecewise planar surfaces, in a Bayesian framework.

The SWC graph $\mathbf{G}$ has atomic regions as nodes. The atomic regions are obtained by intersecting a rectangular grid of 6×6 squares with an edge map obtained by Canny edge detection. For each atomic region r_i, we compute the matching cost for all d in a discrete set $\{0, ..., d_{max}\}$ of possible disparities.

$$C(r_i, d) = \sum_{v \in r_i} D(d, v) \tag{6.5}$$

where $D(d, v)$ is the Birchfield-Tomasi cost from equation 6.2.

Our SWC algorithm will assign a label to each atomic region. In the previous section, this label represented a unique disparity. In the generative setup of this section, each label represents a plane in the disparity space, which does not necessarily has to be fronto-parallel, as in the previous section. Observe that in the case of a perspective projection camera, planar regions in the disparity space correspond to planar regions in the scene and vice versa. Therefore, each label $\ell \in \mathrm{L}$, corresponds to a planar model $\theta_\ell = (a_\ell, b_\ell, c_\ell)$ so that

$$d(v) = a_\ell x + b_\ell y + c_\ell, \ \forall \ v = (x, y) \in V_\ell \tag{6.6}$$

Instead of computing this disparity all the time, we will assume that the disparity is relatively constant in each atomic region r_i, which is a valid assumption since the atomic regions have a small size. Then we approximate the disparity of all pixels in r_i with the disparity of the center $c_i = (x_i, y_i)$ of the region r_i.

$$d(v) = d(c_i) = a_\ell x_i + b_\ell y_i + c_\ell, \ \forall \ v \in r_i \tag{6.7}$$

where $\ell = l(r_i)$. This approximation will greatly reduce the computation time.

Thus the variables that have to be inferred are

$$W = \{l(r_i), \forall r_i\} \cup \theta_\ell, \forall \ell \in \mathrm{L} \tag{6.8}$$

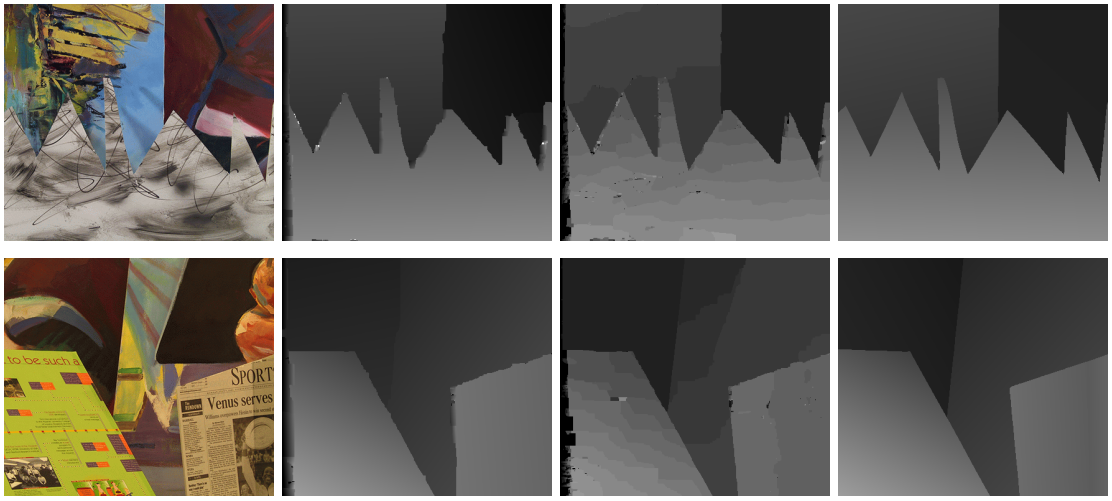
Our Bayesian probabilistic model is

$$p(W|\mathbf{I}) = p(\mathbf{I}|W)p(W) \tag{6.9}$$

The likelihood is given in terms of the model parameters and the matching cost

$$p(\mathbf{I}|W) \propto \exp[-\sum_i C(r_i, a_{l(r_i)}x_i + b_{l(r_i)}y_i + c_{l(r_i)})] \tag{6.10}$$

while the prior is the same as in the previous section.



a. Left image b. SWC result c. Graph cuts d. Manual (truth)

Figure 6.4: Using a Bayesian formulation with generative models fitting piecewise planar surfaces, our algorithm obtains much better results for the same set of stereo images. The running time is also reduced to 2 minutes in a PC.

The SWC edge weights between the atomic regions are defined in terms of the smallest matching cost for fitting both atomic regions by the same disparity

$$q_{ij} = \exp\{-\frac{1}{2}\min_{d}[C(r_i, d) + C(r_j, d)]/(|r_i| + |r_j|)\},\ \forall\ < i, j > \in E. \quad (6.11)$$

At random steps of the algorithm, the models for the regions involved in reassignment are updated by least square fitting of proposals from the textured atomic regions (with high saliency to one disparity only) in each region.

Our algorithm runs in 2 minutes and obtains the much better results shown in Figure 6.4 which are closer to the ground truth. We run the SWC-1 algorithm on the atomic regions and then run the boundary diffusion[63] on pixels for a few steps to smooth the object boundary.

This method applies for simple scenes where the surfaces can be described by parametric models. It will not be as efficient for free-form surfaces such as clothes or faces. For these, PDE optimization techniques [13] could be used.

6.3 Incorporating visual knowledge in stereo

Stereo matching is an intensively studied problem, and recently there is major interest in studying it using effective algorithms such as BP[49] and graph cuts[9] based on Markov random field models and splines [33]. However, the representations (models) used in these methods are still very limited. As a result, they often produce unsatisfactory results when the images have textureless surfaces, such as indoor walls, or when the images have curve structures which do not fit to the pixel-based MRF representation. Fig. 6.21 displays two such results using the state-of-the-art graph cuts algorithm. For such images, one could see that just matching pixels using Markov Random Field priors is not enough. It is desirable to incorporate some visual knowledge into the surface representation. In the stereo literature, the 3D reconstruction of curves was studied in [64], but the

depth was obtained only at the location of curve boundaries and the algorithm does not provide a dense depth estimation.

In what follows, we present a two-layer generative model that incorporates generic middle-level visual knowledge for dense stereo reconstruction. The overall dataflow of the algorithm is illustrated in Fig. 6.5. Given a pair of stereo images, we first compute a primal sketch representation[24] which decomposes the image into two layers. (i) A structural layer for object boundaries and high intensity contrast represented by a 2D sketch graph, and (ii) a structureless layer represented by Markov random field on pixels. The sketch graph in the structural layer consists of a number of isolated points, line segments, and junctions which are considered vertices of different degrees of connection.

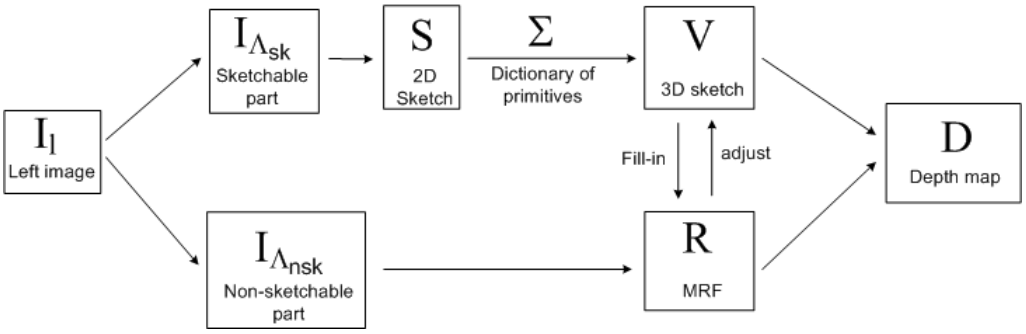


Figure 6.5: The flow diagram of our algorithm.

We then study the 3D structures for these points, line segments, and junctions and develop a dictionary for different configurations. The boundary primitives correspond to places where the depth map is not smooth, namely the boundaries between objects in the scene (first order discontinuities) and the places where the surface normal experiences large changes in direction (second order discontinuities). The curve primitives describe thin curves of different intensity from the background, and usually represent wire-like 3D objects such as thin branches of a tree or electric cables, etc. The point primitives represent feature points in the

image that have reliable depth information. The valid combinations of these 3D primitives is summarized in a dictionary of junctions. Figs. 6.8 and 6.10 shows the dictionaries of line segments and junctions respectively. Each is a 3D surface primitive specified by a number of variables. The number of variables is reduced for degenerated (accidental) cases.

There are three reasons for us to study these primitives. Firstly, the primitive representation reduces the number of variables (dimension reduction). For example, a boundary primitive may have 11 pixels in width and 30 pixels in length, and is described by 4-7 variables which can be more reliably estimated from data. A similar idea was pursued in motion boundary estimation in[5]. Secondly, these sketches are the most informative parts in the images, and it is computationally more effective to compute their 3D depth early and then propagate the surface information to the textureless layers. Thirdly, the dictionary of junction primitives limit the search space by ruling out invalid configurations, like in line drawing interpretation and perceptual organization[45].

We adopt a probability model in a Bayesian framework, where the likelihood is described in terms of the matching cost of the primitives to images, while the prior has terms for continuity and consistency between the primitives, and a Markov Random Field that is used to fill in the depth information in the structureless areas. This Markov Random Field together with the labeling of the edges can be thought of as a Mixed Markov Model [15], in which the neighborhood structure of the MRF depends upon the types of the primitives, and changes dynamically during the computation.

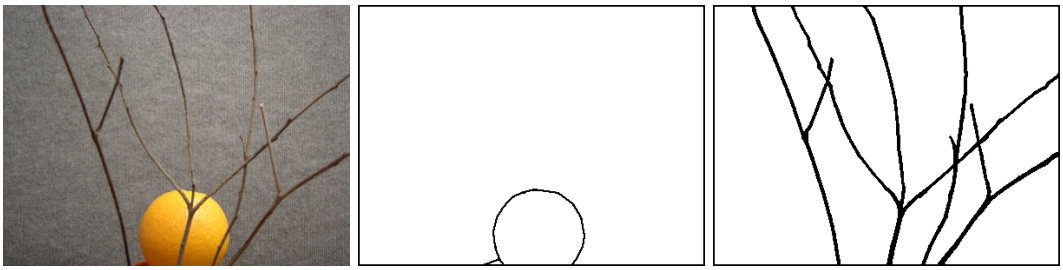
The inference algorithm simultaneously finds the types of the 3D primitives, their parameters and the full depth map. To make-up for the slowdown given by the long range interactions between the primitives through the MRF, the

algorithm makes use of data driven techniques to propose local changes (updates) in the structureless areas.

6.3.1 A two layer representation

Given a stereo pair I_l, I_r of images, we are required to find the depth of all pixels in I_l. Assuming that the camera parameters are known, this is equivalent to finding for each pixel, the horizontal disparity that matches it to a corresponding pixel in I_r. Let D be the disparity map that needs to be inferred and Λ be the pixel lattice.

We assume the disparity map D is generally continuous and differentiable, with the exception of a number of curves Λ_{sk} where the continuity or differentiability assumption does not hold. These curves are augmented with disparity values and are considered to form a 3D sketch D_s that acts as boundary conditions for the Markov Random Field modeling the disparity on $\Lambda_{nsk} = \Lambda \setminus \Lambda_{sk}$.



(a) (b) (c)

Figure 6.6: Our algorithm starts from a two layer sketch representation. (a) input image, (b) region layer, (c) curve layer.

6.3.2 The sketch layer – from 2D to 3D

We assume that the places where the disparity is discontinuous or non-differentiable are among the places of intensity discontinuity. The intensity discontinuities are given in the form of a sketch S consisting of a region layer S_R and a curve layer S_C, as illustrated in figure 6.6. The curve layer is assumed to occlude the region layer. These sketches can be obtained as in [24, 56]. The sketch edges are approximated with line segments $S = \{s_i, i = 1, .., n_e\}$. The segments that originated from the region layer $s_i \in S_R$ will be named *edge segments* while the segments originating from the curve layer $s_i \in S_C$ will be named *curve segments*.

Each edge segment $s_i \in S_R$ from the region layer is assigned two 5 pixel wide *edge regions* l_i, r_i, on the left respectively on the right of s_i, as shown in Figure 6.7, left. Each curve segment $s_j \in S_C$ is assigned a *curve region* r_j along the segment, of width equal to the width of the curve, as shown in Figure 6.7, right. Denote the pixels covered by the edge and curve regions by Λ_R, Λ_C respectively.

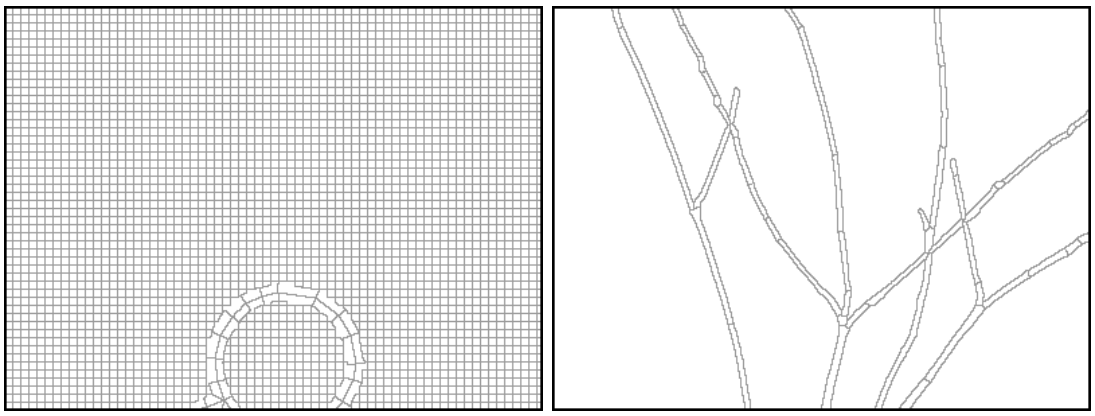


Figure 6.7: Division of the image in figure 6.6 into sketch primitives and 6x6 pixel square regions. Region layer (left) and curve layer (right).

Because away from the places of discontinuity, the surfaces are in general very smooth and to reduce the dimensionality of the problem, the pixels of $\Lambda \setminus \Lambda_R$ are

grouped into *square regions* of size 6×6 pixels, by intersecting $\Lambda \setminus \Lambda_R$ with a 6×6 rectangular grid. Small regions at the boundary between the edge regions and the rectangular grid are merged to the edge regions. This way, the entire lattice Λ is divided into atomic regions that either are along the sketch S_C, or are on the rectangular grid, as shown in Figure 6.7. This structure allows the use of the thin plate spline model for the MRF and also enables implementation of good boundary conditions by the 3D primitives.

Then all line segments $s_i \in S$ are augmented with parameters to become 3D sketch primitives, as shown in figure 6.8. Depending on the type of segments they originated from, there are *boundary primitives* and *curve primitives*.

Let

$$V_1 = \{\pi_i = (s_i, [l_i, o_i^l], r_i, o_i^r, t_i, p_i, d_i, w_i[, f_i]), i = 1, .., n_e\} \tag{6.12}$$

be the set of all primitives, where the parameters in brackets might be missing, depending on the primitive type. The variables of each primitive are:

1. the edge segment $s_i \in S_R$ or curve segment $s_i \in S_C$
2. the left and right regions l_i, r_i in case of an edge segment, or the curve as a region r_i in case of a curve segment.
3. an occlusion label o_i^l, o_i^r for each of the regions l_i, r_i, representing whether the region is occluded (value 0) or not (value 1).
4. the label $t_i = t(\pi_i) \in \{1, .., 8\}$ indexing the type of the primitive from the *primitive dictionary* with the restriction that edge segments $s_i \in S_R$ can only be assigned types from $\{1, .., 6\}$ while curve segments $s_i \in S_C$ can only be assigned types from $\{1, 7, 8\}$. These types are illustrated in Figure 6.8.
    - Type 1 represents edges or curves that are on the surface of the objects.

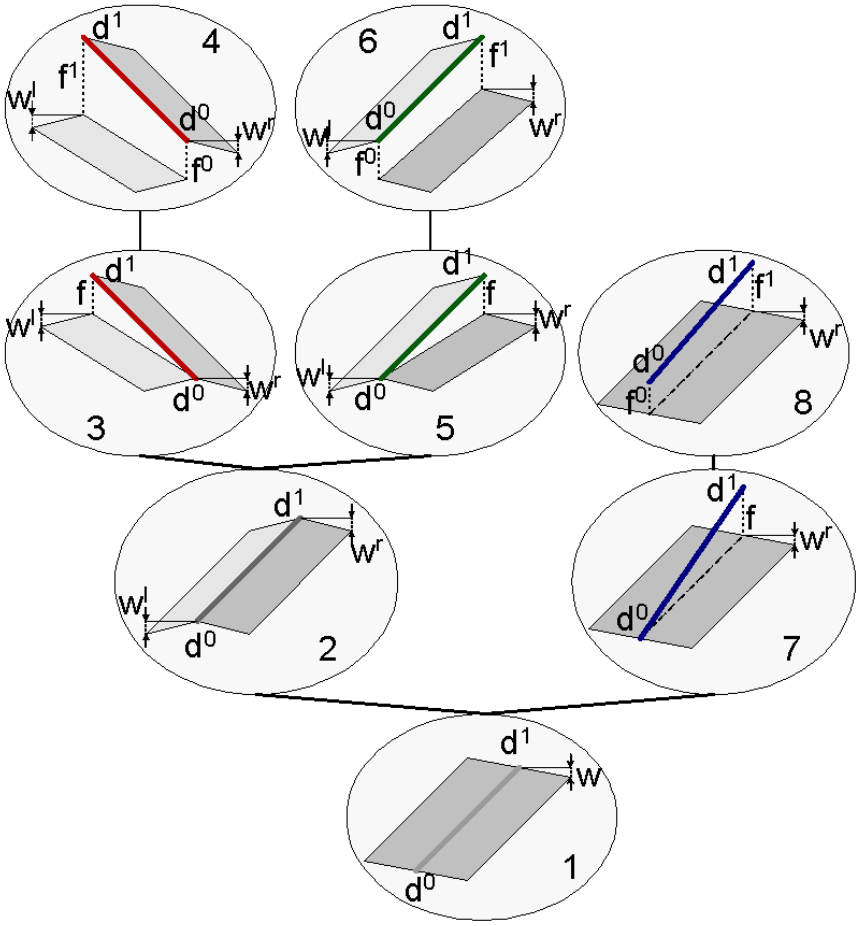


Figure 6.8: Each sketch segment is augmented to a primitive from the following dictionary, order by generality.

- Type 2 represents first order discontinuities, i.e. places where the surface is continuous but the normal is discontinuous.
- Types 3, 4, 5, 6 represent occluding edges where the occluded surface is on the left (types 3, 4) or on the right (types 5, 6) of the edge.
- Types 7, 8 represent 3D curves, either connected with one end to the surface behind, or totally disconnected.

5. a label p_i specifying whether this primitive is a control point (value 1) of

the thin plate spline or not (value 0). All horizontal edges have $p_i = 0$ at all times.

6. the disparities $d_i = d(\pi_i) = (d_i^0, d_i^1)$ at the endpoints of the segment or the disparity $d_i = d(\pi_i)$ at the center of the segment if the segment is short (less than 6 pixels long).

7. the left and right control arm $w_i = w(\pi_i) = (w_i^l, w_i^r)$ representing the slope of the disparity map D in the direction perpendicular to the line segment.

8. for types 3-6, the disparity $f_i = f(\pi_i) = (f_i^0, f_i^1)$ of the occluded surface at the ends of the segment, or the disparity $f_i = f(\pi_i)$ at the center of the edge segment if the segment is short (less than 6 pixels long).

Each of the regions l_i, r_i of the primitive $\pi_i = (s_i, [l_i, o_i^l], r_i, o_i^r, t_i, p_i, d_i, w_i[, f_i])$ is assigned a matching cost

$$c(r_i, d) = \begin{cases} 0 & \text{if } r_i \text{ intersects the curve sketch } S_C \\ \sum_{v \in r_i} |I_l(v) - I_r(v - d_v(\pi_i))| & \text{if } o_i^r = 1 \\ \alpha & \text{else} \end{cases} \tag{6.13}$$

where for each pixel $v \in r_i$, the disparity $d_v(\pi_i)$ is the linear interpolation based on the parameter d representing the disparity at the ends of the region, in the assumption that $w = 0$. Then the matching cost of the primitive π_i is

$$c(\pi_i) = c(r_i, [l_i], t_i, d_i, [f_i]) = \begin{cases} c(r_i, d_i) & \text{if } t_i = 7, 8, 1(\text{curve}) \\ c(l_i, d_i) + c(r_i, d_i) & \text{if } t_i = 2, 1(\text{region}) \\ c(l_i, f_i) + c(r_i, d_i) & \text{if } t_i = 3, 4 \\ c(l_i, d_i) + c(r_i, f_i) & \text{if } t_i = 5, 6 \end{cases} \tag{6.14}$$

The primitives form a graph by the natural adjacency relation between the underlying edge segments, as it can be seen in Figure 6.9.

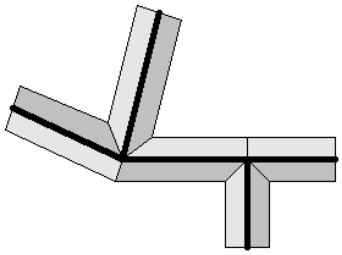


Figure 6.9: A set of edge segments (black) and their 3D primitives (gray). The primitives form a graph by the adjacency of the segments.

To increase the model accuracy, the junction points of two or more primitives are modeled. Similar to [45] we will have certain types of possible junctions depending on the degree (number of primitives) of the junction, as mentioned below and illustrated in Fig. 6.10.

- junctions of 2 boundary primitives have three main types: Surface junctions, beginning of occlusion and occlusion junctions.
- junctions of 3 boundary primitives have three main types: Surface junctions, Y-junctions and T-junctions.
- junctions of 4 or more boundary primitives are accidental and are assumed to be all surface junctions.
- we assume there are no junctions between one or two curve primitives and one boundary primitive
- junctions of 1 curve primitive with two boundary primitives have three main types: curve beginning, Y-junctions and T-junctions.

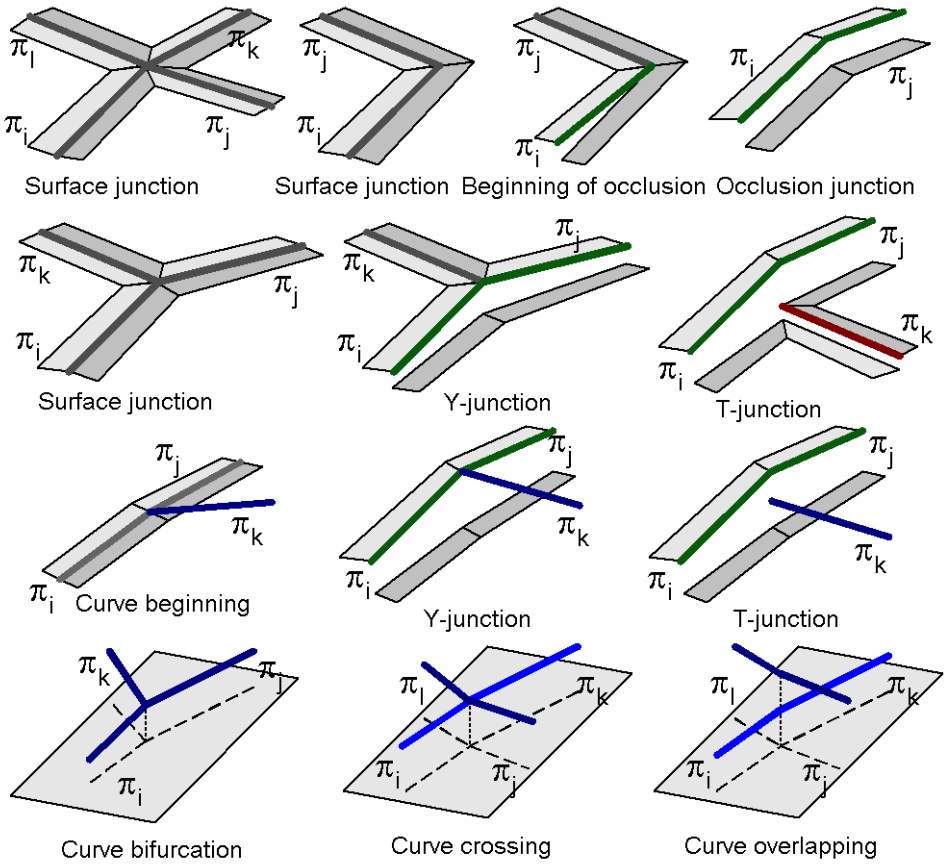


Figure 6.10: These are the main types of junctions between boundary and curve primitives.

- junctions of 2 curve primitives have only one type.
- junctions of 3 curve primitives have only one type, namely bifurcation.
- junctions of 4 curve primitives have two types, namely curve crossing of curve overlapping. In both cases, the opposite primitives can be seen as part of the same 3D curve.

Let $J = \{\phi_i = (t, k, \pi_{i_1}, ..., \pi_{i_k}), \pi_{i_1}, ..., \pi_{i_k} \in V_1, i = 1, ..., n_J\}$ be the set of junctions, each containing the list of primitives that are adjacent to it. The variable

t is the junction type and restricts the types of the primitives $\pi_{i_1}, ..., \pi_{i_k}$ to be compatible to it.

Each junction $\phi_i \in J$ imposes a prior model that depends on the junction type, and the types and directions of the 3D primitives $\pi_{i_1}, ..., \pi_{i_k}$ meeting in this junction. This prior is composed of a 3D geometric prior on the primitives and a 2D occurrence prior of each particular junction type.

Thus

$$P(\phi) \propto P(\pi_{i_1}, ..., \pi_{i_k}|t, \phi^{2D})P(\phi^{2D}, t) = P(\phi^{3D}|t, \phi^{2D})P(t|\phi^{2D}) \tag{6.15}$$

since the 2d geometry ϕ^{2D} of the junction is fixed.

We will now discuss $P(\phi^{3D}|t, \phi^{2D})$ for each junction type.

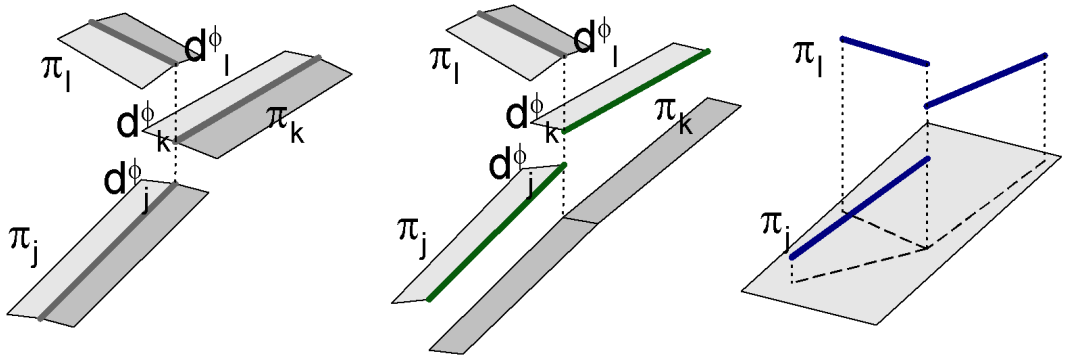


Figure 6.11: The prior of the junction between 3 or more boundary primitives and the curve bifurcation or crossing encourages 3D continuity of the primitives.

1) All the surface junctions of 3 or more boundary primitives and the curve bifurcation or crossing have a prior that prefers the same disparity for all primitives meeting at this junction.

$$P(\phi^{3D}|t, \phi^{2D}) = \frac{1}{Z_1} \exp(-\beta_c \sum_{\pi_j, \pi_k \in \phi} |d_j^\phi - d_k^\phi|^2) \tag{6.16}$$

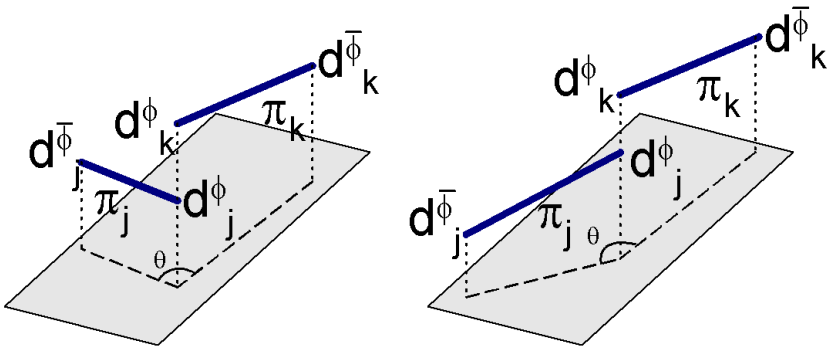


Figure 6.12: The prior of the junction between two boundary or curve primitives depends on the angle θ between the primitives.

where $d_j = (d_j^{\phi}, d_j^{\bar{\phi}})$ is the disparity of the primitive π_j, d_j^{ϕ} being the disparity at the junction ϕ endpoint.

2) The prior of junctions of two boundary or two curve primitives depends on the angle θ between the primitives at the junction. First, define the curve continuity measure

$$s(\pi_j, \pi_k) = |d_j^{\bar{\phi}} - 2d_j^{\phi} + d_k^{\bar{\phi}}|^2 + |d_j^{\bar{\phi}} - 2d_k^{\phi} + d_k^{\bar{\phi}}|^2 \tag{6.17}$$

Then the prior is

$$P(\phi^{3D}|t, \phi^{2D}) = \frac{1}{Z_2} \begin{cases} \exp(-\beta_c |d_j^{\phi} - d_k^{\phi}|^2) & \text{if } |\theta - \pi| > \pi/6 \\ \exp[-\beta_c |d_j^{\phi} - d_k^{\phi}|^2 - \beta_s s(\pi_j, \pi_k)] & \text{else} \end{cases} \tag{6.18}$$

as shown in Figure 6.12.

3) For the curve overlapping junction involving four curve primitives, the prior is defined in terms of the continuity of each pair of opposite curves.

$$P(\phi^{3D}|t, \phi^{2D}) = \frac{1}{Z_3} \exp[-\beta_c(|d_i^{\phi} - d_k^{\phi}|^2 + |d_j^{\phi} - d_l^{\phi}|^2) - \beta_s(s(\pi_i, \pi_k) + s(\pi_j, \pi_l))] \tag{6.19}$$

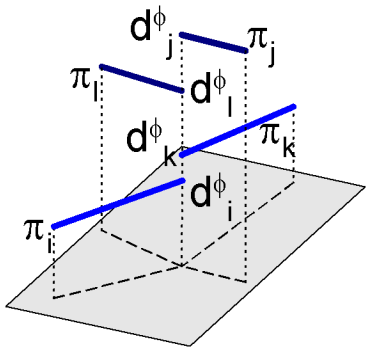


Figure 6.13: The prior of the curve overlapping junction encourages continuity of each pair of opposite curves.

as shown in Figure 6.13.

4) For the Y-junctions of 3 boundary primitives and for the curve beginning, the prior encourages all three primitives to be adjacent, and the primitives π_i, π_j (refer to Figure 6.10) to have a good continuation as in case 2).

$$P(\phi^{3D}|t, \phi^{2D}) = \frac{1}{Z_4} \exp[-\beta_c \sum_{\pi_u, \pi_v \in \phi} |d_v^{\phi} - d_u^{\phi}|^2 - \beta_s s(\pi_i, \pi_j)] \tag{6.20}$$

5) For the T-junctions, the prior encourages continuity of the occluding edge.

$$P(\phi^{3D}|t, \phi^{2D}) = \frac{1}{Z_5} \exp[-\beta_c |d_i^{\phi} - d_j^{\phi}|^2 - \beta_s s(\pi_i, \pi_j)] \tag{6.21}$$

Since the disparity space of each primitive is discretized, the normalizing constant for each junction can be computed effectively.

The prior $P(t|\phi^{2D})$ can be learned from hand labeled data, independently for each degree (number of primitives) k of the junction.

Based on the matching cost, a saliency map

$$\psi_{\pi_i}(d, [f]) = \exp(-c(r_i, [l_i], t_i, d, [f])/10) \tag{6.22}$$

towards all possible values of d, f is computed for each primitive $\pi_i \in V_1$. This information will be used to find the disparities d_i of the sketch primitives.

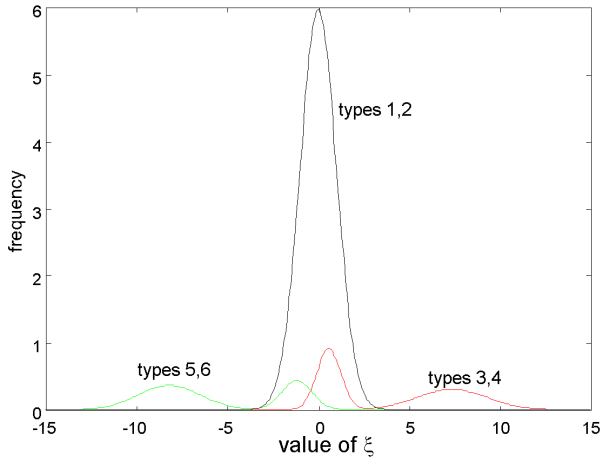


Figure 6.14: The frequency of the feature ξ as fitted from hand labeled data.

We also compute a saliency map towards the three main types of boundary primitives, namely surface (types $1, 2$), occluding left (types $3, 4$), occluding right (types $5, 6$), based on the feature

$$\xi(\pi_i) = \frac{\min_d c(l_i, d)}{|l_i|} - \frac{\min_d [c(l_i, d) + c(r_i, d)]}{|l_i| + |r_i|} \tag{6.23}$$

which measures how well both wings of the primitive fit the same disparity, as compared to the left wing alone.

From hand labeled data, we obtained histograms H_{12}, H_{34}, H_{56} of the values of ξ for each of the three main types of boundary primitives. We fit these histograms with gaussians to even out the small amount of training data and eliminate the need for histogram bins. The fitted gaussians and their relative importance are displayed in Figure 6.14. From here we obtain a likelihood $L_\pi(t)$ towards the

three main types of boundary primitives.

$$L_\pi(t) = \begin{cases} 60e^{-\xi^2/2.} \text{ if } \ t = 1, 2 \\ 4.4e^{-(\xi+1.18)^2/1.42} + 3.67e^{-(\xi+8.21)^2/6} \text{ if } \ t = 3, 4 \\ 9.18e^{-(\xi-0.58)^2/0.87} + 3.06e^{-(\xi-7.36)^2/7.5} \text{ if } \ t = 5, 6 \end{cases} \tag{6.24}$$

Using the intensity-driven likelihood for the boundary primitives, we construct a likelihood, driven simultaneously by the image intensity and the geometry (relative position of primitives), for each junction $\phi = \{\pi_1, ..., \pi_k\}$:

$$L_\phi(t) = P(\phi)L_{\pi_1}(t_1)...L_{\pi_k}(t_k) \tag{6.25}$$

6.3.3 The free-form layer

The primitives $\pi \in V_1$ discussed in the previous section are elongated primitives corresponding to line segments, so they can be considered of dimension 1. Other sketch primitives that are involved in the free form layer are the zero dimensional primitives corresponding to feature points with reliable disparity information, i.e. *point primitives*. These primitives are a subset of the rectangular atomic regions, and together with the one dimensional boundary primitives are the control points of the thin plate spline. The curve primitives are not involved in the MRF computation.

Let R be the set of all rectangular atomic regions.

For each region $r \in R$, we compute a saliency map

$$\rho_r(d) \propto \exp(-\sum_{v \in r} |I_l(v) - I_r(v - d))|/10) \tag{6.26}$$

to all possible disparities $d \in [d_{\min}, d_{\max}]$

Then the square regions

$$R = \{r_i = (d_i, o_i, p_i, \mu_i, \sigma_i^2), i = 1, .., n_r\} \tag{6.27}$$

have the following parameters:

1. the disparity $d_i = d(r_i)$ of the center of the region.

2. a label o_i specifying whether this region is occluded (value 0) or not (value 1).

3. a label $p_i = p(r_i) \in \{0, 1\}$ representing whether the region is a point primitive (i.e. control point for the thin plate spline) or not.

4. the mean μ_i and variance σ_i^2 of the saliency map ρ_{r_i}.

All regions (edge regions, curve regions and square regions) will have their occlusion label deterministically assigned based on the disparities of the boundary and curve primitives. For example, for an occlusion primitive π_i of type 4, the left region l_i and other regions horizontally to the left of the edge at horizontal distance less than the disparity difference between the right and left wings of π_i will be labeled as occluded.

The matching cost for each region $r_i \in R$ is

$$c(r_i) = \begin{cases} \alpha & \text{if } o_i = 0 \\ \sum_{v \in r_i} |I_l(v) - I_r(v - d_i))| & \text{if } o_i = 1 \end{cases} \tag{6.28}$$

The set of point primitives is denoted by

$$V_0 = \{r_i \in R, s_i = 1\}. \tag{6.29}$$

In Figure 6.15 we present the labeled graph, i.e. primitive types (middle), and the point and boundary primitives that act as control points for the Λ_{nsk} part (right). The depth and disparity maps obtained this way are shown in Figure 6.22. Observe that the horizontal edges are not control points.

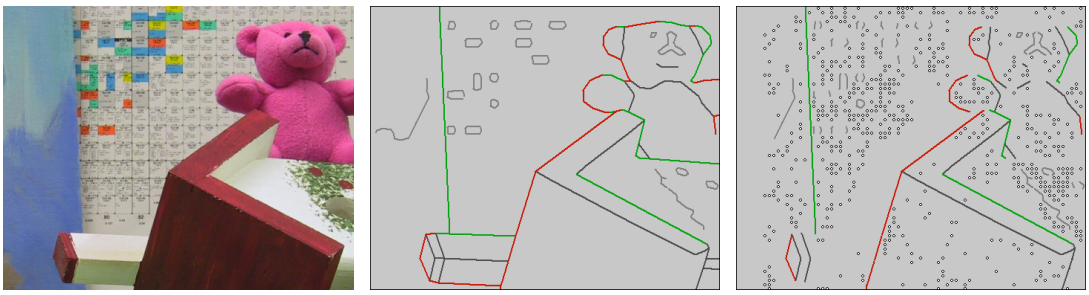


Figure 6.15: Left image of a stereo sequence, the graph labeling and the control points (point and boundary primitives) of the thin plate spline.

The dense disparity map D is obtained from V_1 and R by interpolation. By using the boundary primitives to model the places of discontinuity, the obtained disparity map has crisp discontinuities at the object boundaries and is smooth everywhere else, as shown in Figure 6.22.

6.3.4 Bayesian formulation

We formulate our model using the Bayes rule:

$$P(V_1, R|I_l, I_r) = P(I_l|I_r, V_1, R)P(R - V_0|V_0, V_1)P(V_0, V_1) \tag{6.30}$$

The likelihood $P(I_l|I_r, V_1, R)$ is expressed in terms of the likelihood $L_{\pi_i}(t_i)$ and matching cost $c(r_j)$ of the sketch primitives.

$$P(I_l|I_r, V_1, R) \propto \prod_{i=1}^{n_e} L_{\pi_i}(t_i) \exp[-\sum_{r_j \in R} c(r_j)] \tag{6.31}$$

The prior

$$P(R - V_0|V_0, V_1) \propto \exp[-E_c(R) - \beta_b E_b(R, V_1)] \tag{6.32}$$

is defined in terms of the energy of the soft control points:

$$E_c(R) = \sum_{r_j \in V_0} (d_j - \mu_j)/2\sigma_j^2 \tag{6.33}$$

and the thin plate bending energy:

$$E_b(R, V_1) = \sum_{(x,y)\in G} [d_{xx}^2(x,y) + 2d_{xy}(x,y)^2 + d_{yy}^2(x,y)] \tag{6.34}$$

which is computed on a 6×6 grid G containing the centers of all the square regions and neighboring grid points on the boundary primitives. For example, if the point $(x, y) \in G$ is the center of $r_j \in R$ and $r_N, r_{NW}, r_W, r_{SW}, r_S, r_{SE}, r_E, r_{NE}$ are the 8 neighbors of r_j, then

$$\begin{aligned} d_{xx}(x,y) &= d_W - 2d_j + d_E \\ d_{yy}(x,y) &= d_N - 2d_j + d_S \\ d_{xy}(x,y) &= (d_{NE} + d_{SW} - d_{NW} - d_{SE})/4 \end{aligned}$$

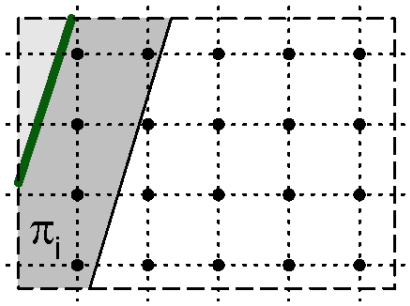


Figure 6.16: The region not covered by boundary primitives has a thin plate spline prior, computed on a rectangular grid that intersects the wings (atomic regions) of the primitives.

Similar terms in the bending energy $E_b(R, V_1)$ can be written for cases where one or many of the neighbors are boundary primitives. However, there are no terms involving the left and right atomic regions $l_i, r_i \in \pi_i$ belonging to the same edge primitive π_i.

The prior $P(V_0, V_1) = P(V_0)P(V_1)$ assumes a uniform prior on V_0 while $P(V_1)$ is defined in terms of the junction priors $P(\phi_i)$ defined in section 6.3.2.

$$P(V_1) = \prod_{\phi_i \in J} P(\phi_i) \tag{6.35}$$

6.3.5 The inference algorithm

In our problem formulation, there are two types of variables, discrete and continuous. The discrete variables are

$$\Delta = V_1^d \cup R^d \tag{6.36}$$

consisting of $V_1^d = \{(t(\pi), o^l(\pi), o^r(\pi), p(\pi)), \forall\, \pi \in V_1\}$ and $R^d = \{(s(r), o(r), p(r)), \forall\, r \in R\}$. All other variables are continuous variables, namely $V_1^c = V_1 \setminus V_1^d$ and $R^c = R - R^d$, and can be divided into the boundary conditions

$$\Gamma = V_0^c \cup \{d(\pi), \forall\, \pi \in V_1, p(\pi) = 1\} \tag{6.37}$$

and the fill-in variables

$$\begin{aligned} \Psi = \{([w(\pi)], [f(\pi)]), \forall\, \pi \in V_1\} \cup \\ \{d(\pi), \forall\, \pi \in V_1, p(\pi) = 0\} \cup R^c - V_0^c. \end{aligned} \tag{6.38}$$

The posterior probability can then be written as

$$p(V_1, R | I_l, I_r) = p(\Delta, \Gamma, \Psi | I_l, I_r) \tag{6.39}$$

In a MAP formulation, our algorithm needs to perform the following three tasks:

1. Reconstruct the 3D sketch to infer the parameters Γ of the primitives.

2. Label the primitive graph to infer the discrete parameters Δ, i.e. associate the primitives with the appropriate types. This represents the detection of surface boundaries and of the feature points of the image.

3. Perform "fill in" of the remaining parts of the image, using the MRF and Γ, Δ as boundary conditions, to infer Ψ and obtain a dense disparity map D.

The algorithm will proceed as follows. In an initialization phase, the first two steps will be performed to compute an approximate initial solution. Then steps 2) and 3) will be performed to obtain the final result.

6.3.5.1 Initialization

Initializing the system purely based on the local depth ψ_π and likelihood $L_\pi(t)$ information existent at the primitives $\pi \in V_1$ results in an inconsistent initial solution which is valid only at places with reliable local depth information, as shown in Figure 6.17.

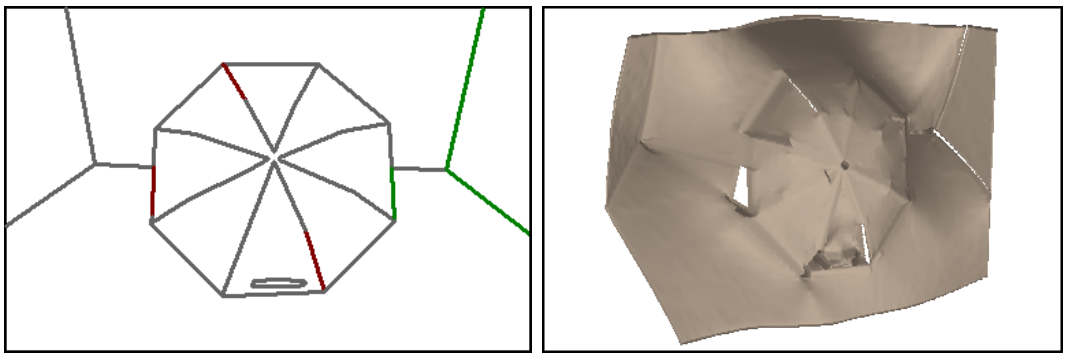


Figure 6.17: An initialization purely based on local information is not satisfactory.

A major improvement can be achieved by using the junction prior $P(\phi)$ that has been defined in Section 6.3.2, which provides a way to propagate depth information quickly along the edges of the sketch, from the places where it is available.

This is why we use an approximation of the posterior probability that only takes into account the matching cost of the edge regions $\pi_i \in V_1$ and the junction prior.

$$P(V_1|I_l, I_r) \propto \exp[-\sum_{i=1}^{n_e} c(\pi_i)] \prod_{\phi_i \in J} P(\phi_i) \tag{6.40}$$

At this stage, the variables that highly depend on the thin plate spline prior will be assigned some default values. Thus, the wing parameters $w_i, \forall \pi_i \in V_1$ will be assigned value 0 (i.e. all wings will be horizontal), while the occlusion labels o_i will be assigned value 1 (unoccluded).

The initialization algorithm alternates the following MCMC steps:

- a single node move that changes one variable d_i at a time.
- a move that simultaneously shifts all d_i at the same junction ϕ by the same value. This move is capable of adjusting the disparity of primitives at a junction at times when changing the disparity of only one primitive will be rejected because of the continuity prior.
- a labeling move as described in the MCMC algorithm section 6.3.5.3, which proposes a new labeling for a set of primitives and junctions. The move is accepted using the Metropolis-Hastings method based on the posterior probability from Eq. (6.40).

The algorithm is run for $10|V_1|$ steps and obtains the initialization result shown in Figure 6.18 in about 10 seconds. The initialization algorithm is very fast because the fill-in of the interior pixels is not performed, eliminating the expensive MRF computation.

The 3D reconstruction of the curve primitives is performed separately in a similar manner. The labeling move is much simpler, since the curve primitives

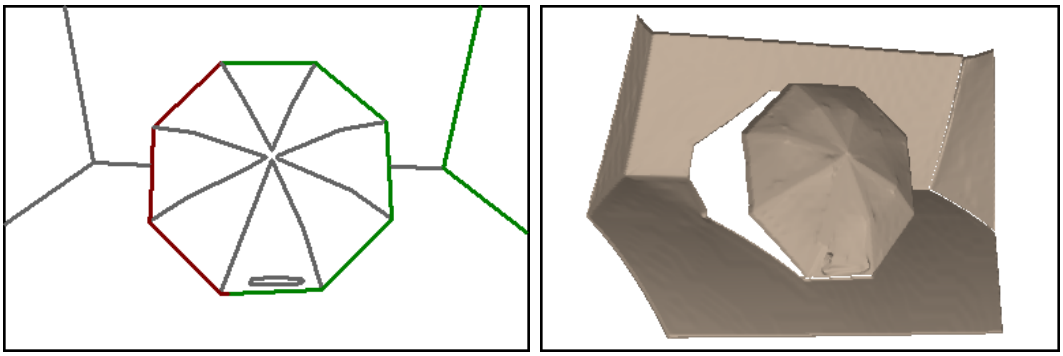


Figure 6.18: By propagating the junction priors along the sketch, a much better initialization can be quickly obtained.

basically accept two labels, surface/non-surface.

6.3.5.2 Updating the fill-in variables Ψ

Observe that in our formulation of the energy, if Δ, Γ are fixed, the conditional $-\log(P(\Psi|\Delta,\Gamma))$ is a quadratic function in all the variables Ψ, so it can be minimized analytically. This implies that Ψ can be regarded as a function on Δ, Γ, i.e. $\Psi = \Psi(\Delta, \Gamma)$. This restricts the problem to maximizing the probability $P(\Delta, \Gamma, \Psi(\Delta, \Gamma)|I_l, I_r)$, of much smaller dimensionality.

However, minimizing $-\log(P(\Psi|\Delta,\Gamma))$ analytically involves inverting a matrix of size $n \times n$, where $n = |\Psi|$. This can be computationally expensive if all the variables of Ψ are updated at the same time, since Ψ is on the order of $|\Psi| \sim 4000$. But since inside each of the regions bounded by the control point sketch primitives, the variables depend only on the control points inside and on the boundary of this region, the computation can be localized to each of these regions independently, as shown in Figure 6.19, and the computation demand will be much lower.

Observe that the update can affect some non-control point edges, such as the

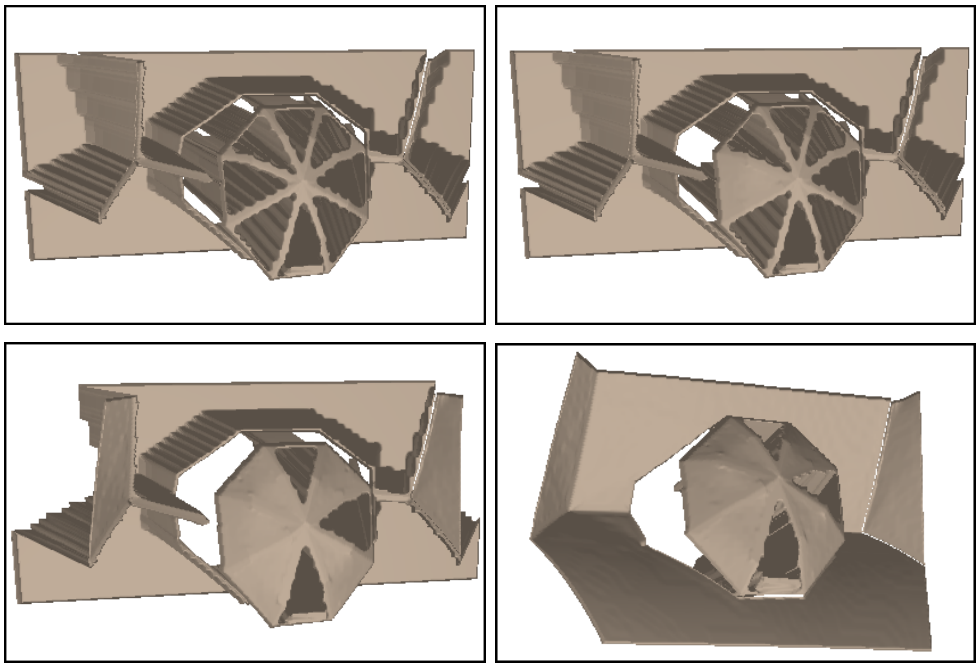


Figure 6.19: The fill-in can be restricted to the connected components bounded by control point boundary primitives. In a few steps, the initial 3D reconstruction before graph labeling is obtained. Shown are the 3D reconstructions after 0,1,4,5 connected components have been updated. The horizontal edges change the disparity at the same time with the interior, because they are not control points.

horizontal edges from Figure 6.19.

For each such connected component C, we define relative labels l_C of the edges adjacent to C that only take into account the side of the edge that belongs to C. For example, an occluding edge type 4 and an edge of type 1 will have the same label relative to the component containing the atomic region on the right of the edge. Using these relative labels, we reduce the computation expense, b defining the energy of the region

$$E(C, l_C) = \sum_{l(\pi)\in C} c(l(\pi)) + \sum_{r(\pi)\in C} c(r(\pi)) + \sum_{r\in C\cap R} c(r) + E_c(C) + \mu_b E_b(C) \quad (6.41)$$

This energy will be memorized for each pair C, l_C, so that, during the MCMC

optimization described below, if the same labeling combination occurs for this region, the energy computation will be readily available without performing another MRF reconstruction.

The full posterior probability can be recovered from the energy of the regions and the junction prior:

$$P(V_1, R|I_l, I_r) \propto \exp[-\sum_C E(C, l_C)] \prod_{\phi \in J} P(\phi) \tag{6.42}$$

6.3.5.3 The MCMC optimization algorithm

After the initialization, the 3D sketch variables $\Gamma = \Gamma_0$ will be fixed. The algorithm will only update the primitive types Δ and the fill-in variables Ψ.

To maximize $P(\Delta, \Gamma_0, \Psi(\Delta, \Gamma_0)|I_l, I_r)$ we will use a Markov chain Monte Carlo algorithm that will sample $P(\Delta, \Gamma_0, \Psi(\Delta, \Gamma_0)|I_l, I_r)$, and this way obtain the most probable solutions.

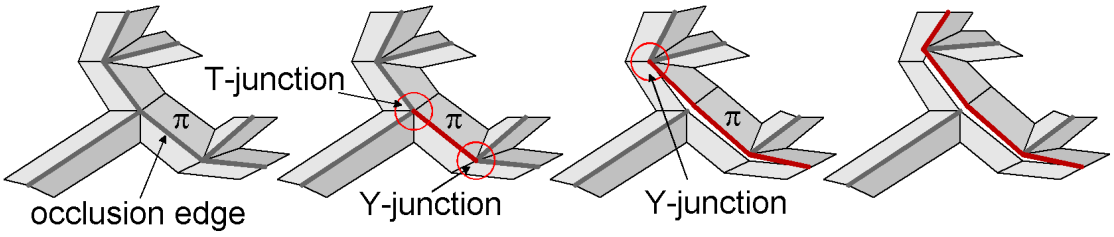


Figure 6.20: Each graph labeling move changes the types of a set of primitives in a consistent manner. First a primitive π is chosen and its type is sampled from the likelihood $L_\pi(t)$, then the adjacent junctions change their type conditional on the chosen type of π, which in turn determine the types of the other primitives of the junctions, etc. The labeling move is accepted based on the Metropolis-Hastings method. Illustrated is the left side of the umbrella image.

At each step, the algorithm proposes, as shown in Figure 6.20, new types for

a set of primitives N and junctions J in one move described below:

1. a random non-horizontal primitive π is chosen and N is initialized $N = \{\pi\}$ while J is initialized with the two junctions ϕ_1, ϕ_2 adjacent to π, $J = \{\phi_1, \phi_2\}$.

2. the primitive type $t(\pi)$ is sampled from the local likelihood $L_\pi(t)$.

3. conditional on the primitive type $t(\pi)$, the type of each of the two junctions $\phi \in J$ is sampled from $L_\phi(t)$. This determines the types of all primitives of

$$N_n = \{\pi' \notin N, \pi' \sim \phi \text{ for some } \phi \in J\}, \tag{6.43}$$

   where $\pi \sim \phi$ means π is adjacent to ϕ.

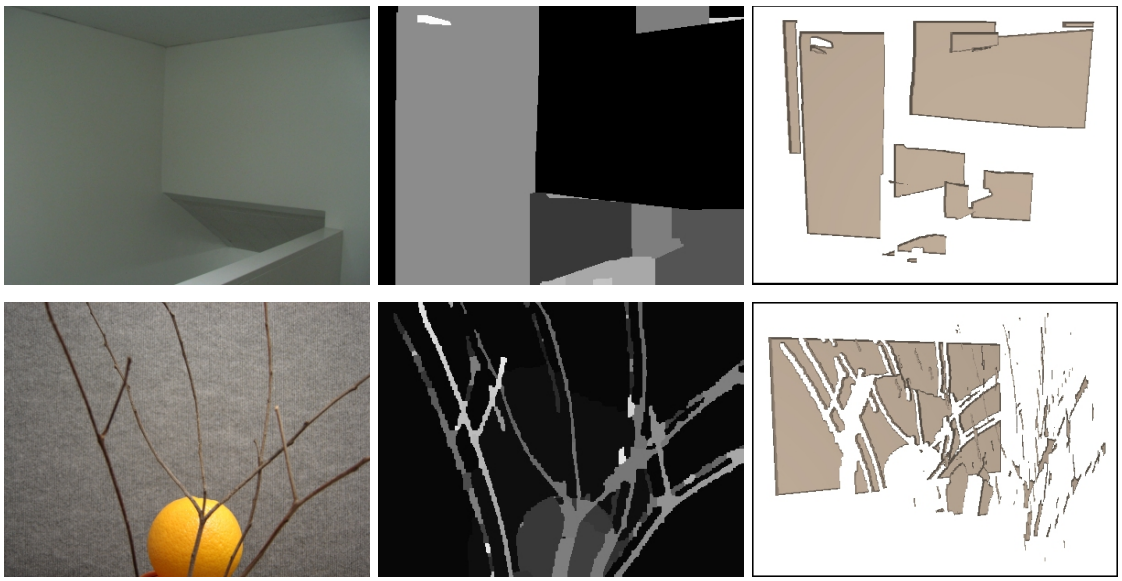
4. N is updated $N \leftarrow N \cup N_n$.

5. The junctions adjacent all primitives $\pi \in N_n$ are added to J as follows. Initialize $J_n = \emptyset$. For each $\pi \in N_n$, we pick the adjacent junction $\phi \notin J$. If the primitive had its type changed, then $J_n \leftarrow J_n \cup \{\phi\}$. If the primitive and has the same type as before this move, then $J_n \leftarrow J_n \cup \{\phi\}$ with probability 0.5. Set $J \leftarrow J \cup J_n$.

6. for each $\pi \in N_n$ and each $\phi \in J_n, \pi \sim \phi$, repeat steps 3-5.

After each proposal, the fill-in variables $\Psi(\Delta, \Gamma)$ are updated for the connected components C for which it is necessary.

The labeling move is accepted based on the full posterior probability, computed using eq. (6.42).



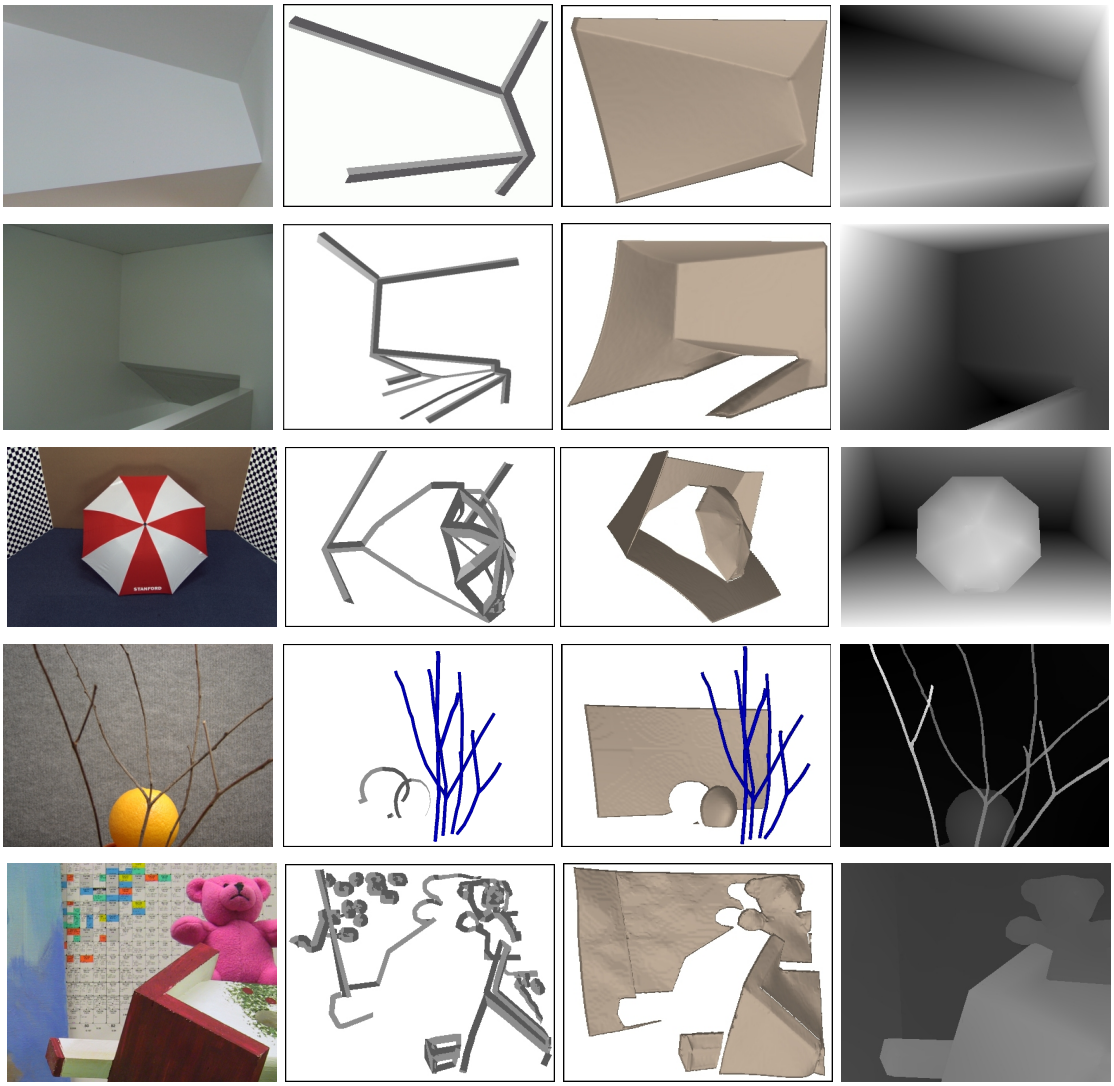
(a) (b) (c)

Figure 6.21: Two comparison examples using the **graph cuts algorithm** on scenes containing textureless surface and curve structures. (a) left image, (b) disparity map, (c) 3d map. Our results are shown in Fig.6.22.

6.3.6 Experimental results

The experiments are presented in Fig.6.22 where five typical images for stereo matching are shown. The first two have textureless surfaces and the most information is from the surface boundaries. The fourth image has curves (twigs). For these three images, it is not a surprise to see that the graph cut method with simple MRF models on pixels produce unsatisfactory results (see Fig.6.21 for comparison).

The third and fifth images are from [33] and [47] respectively and have free-form surfaces with or without textures. We have also shown the interactions of the two layers in Fig.6.19 and the effects of sketch labeling in Fig. 6.17 and 6.18.



(a) (b) (c) (d)

Figure 6.22: Results obtained using our method. (a) left image of the stereo pair, (b) 3D sketch using the primitives, (c) 3D depth map, (d) disparity map.

CHAPTER 7

Conclusions

In this thesis, we present a generic inference algorithm for sampling arbitrary probabilities or energy functions on general graphs by extending the SW method from physics and the Gibbs sampler (SWC-3). Our method extends the SW method from the Metropolis-Hastings perspective and it is thus different from other interpretations in the literature[11, 26]. In fact, there were some early attempts for applying SW to image analysis[26, 7] using a partial decoupling concept.

The speed of the SW-cut method depends on the discriminative probabilities on the edges and vertices. Such probabilities also make a theoretical analysis of convergence difficult. In ongoing projects, we are studying ways for bounding the SW-cut convergence with "external field" (data) and for diagnosing exact sampling using recent advanced techniques. We are also incorporating the SW-Cuts into the DDMCMC framework for image parsing.

APPENDIX A

Proof of Theorem 4

Consider a reversible jump between two states $\mathbf{X}$ and $\mathbf{X}'$ which differ only in the labeling of R,

$$\mathbf{X}_R = \ell \neq \ell' = \mathbf{X}'_R, \;\; \mathbf{X}_{\bar{R}} = \mathbf{X}_{\bar{R}}. \tag{A.1}$$

As we have already mentioned, the acceptance probability is computed by the Metropolis-Hastings method

$$\alpha(\mathbf{X} \to \mathbf{X}') = \min\{1, \frac{q(\mathbf{X}' \to \mathbf{X})}{q(\mathbf{X} \to \mathbf{X}')} \cdot \frac{\pi(\mathbf{X}')}{\pi(\mathbf{X})}\}. \tag{A.2}$$

We will now compute the proposal probabilities $q(\mathbf{X} \to \mathbf{X}')$ and $q(\mathbf{X}' \to \mathbf{X})$.

First we consider the canonical case when there is a unique path for moving between states $\mathbf{X}$ and $\mathbf{X}'$ in one step – choosing R and changing its color from ℓ to ℓ'.

Then the computation of the probability $q(\mathbf{X} \to \mathbf{X}')$ follows the description of the algorithm.

Let $\mathbf{U}|\mathbf{X}$ and $\mathbf{U}'|\mathbf{X}'$ be the auxiliary variables following the Bernoulli probabilities in the flipping step. They lead to two sets of connected components $\mathrm{CP}(\mathbf{U}|\mathbf{X})$ and $\mathrm{CP}(\mathbf{U}'|\mathbf{X}')$ respectively. We divide $\mathbf{U}$ into two sets for the "on" and "off" edges respectively,

$$\mathbf{U} = \mathbf{U}_{\mathrm{on}} \cap \mathbf{U}_{\mathrm{off}}. \tag{A.3}$$

$$\mathbf{U}_{\text{on}} = \{\mu_{ij} \,:\, \mu_{ij} = 1\}, \;\; \mathbf{U}_{\text{off}} = \{\mu_{ij} \,:\, \mu_{ij} = 0\}.$$

We are only interested in the configurations $\mathbf{U}$ (and thus CP's) which yield the connected component R. We collect all such $\mathbf{U}$ given $\mathbf{X}$ in the set,

$$\Psi(R|\mathbf{X}) = \{\mathbf{U} \,:\, R \in \text{CP}(\mathbf{U}|\mathbf{X})\}. \tag{A.4}$$

In order for R to be a connected component in $\mathbf{X}$, all edges between R and $V_\ell \backslash R$ must be cut (turned off), otherwise R is connected to other vertices in V_ℓ and cannot be a connected component. So, we denote the remaining "off" edges by ${}^{-}\mathbf{U}_{\text{off}}$,

$$\mathbf{U}_{\text{off}} = \mathcal{C}(R, V_\ell) \cup {}^{-}\mathbf{U}_{\text{off}}, \quad \forall\, \mathbf{U} \in \Psi(R|\mathbf{X}). \tag{A.5}$$

Similarly, we collect all $\mathbf{U}'$ in state $\mathbf{X}'$ which produce the connected component R,

$$\Psi(R|\mathbf{X}') = \{\mathbf{U}' \,:\, R \in \text{CP}(\mathbf{U}'|\mathbf{X}')\}. \tag{A.6}$$

In order for R to be a connected component in $\mathbf{U}'|\mathbf{X}'$, the clustering step must cut all the edges between R and $V_{\ell'}$. Thus we have

$$\mathbf{U}' = \mathbf{U}'_{\text{on}} \cap \mathbf{U}'_{\text{off}} \tag{A.7}$$

with

$$\mathbf{U}'_{\text{off}} = \mathcal{C}(R, V_{\ell'}) \cup {}^{-}\mathbf{U}'_{\text{off}}, \quad \forall\, \mathbf{U}' \in \Psi(R|\mathbf{X}'). \tag{A.8}$$

We are now ready to compute $q(\mathbf{X} \to \mathbf{X}')$. Suppose that we choose $R \in \text{CP}$ with probability $q(R|\text{CP})$. Since for all configurations $\Psi(R|\mathbf{X})$, the probability to change the label of R to ℓ' has the same value $q(\mathbf{X}_R = \ell'|R, \mathbf{X})$, we have

$$q(\mathbf{X} \to \mathbf{X}') = q(R|\mathbf{X}) q(\mathbf{X}_R = \ell'|R, \mathbf{X}) \tag{A.9}$$

The two factors correspond to the clustering and flipping steps, respectively.

Then the probability $q(R|\mathbf{X})$ of choosing R at state $\mathbf{X}$ is the sum over all possible $\mathbf{U} \in \Psi(R|\mathbf{X})$ of the probability of choosing $\mathbf{U} \in \Psi(R|\mathbf{X})$ times the probability of choosing R from $\mathrm{CP}(\mathbf{U}|\mathbf{X})$,

$$q(R|\mathbf{X}) = \sum_{\mathbf{U}\in\Psi(R|\mathbf{X})} [q(R|\mathrm{CP}(\mathbf{U}|\mathbf{X})) \prod_{<i,j>\in\mathbf{U}_{\mathrm{on}}} q_{ij} \prod_{<i,j>\in^{-}\mathbf{U}_{\mathrm{off}}} (1-q_{ij})] \prod_{<i,j>\in\mathcal{C}(R,V_\ell)} (1-q_{ij}). \tag{A.10}$$

Similarly, the probability for choosing $R \subseteq V_{\ell'}$ at state $\mathbf{X}'$ is

$$q(R|\mathbf{X}') = \sum_{\mathbf{U}'\in\Psi(R|\mathbf{X}')} [q(R|\mathrm{CP}(\mathbf{U}'|\mathbf{X}')) \prod_{<i,j>\in\mathbf{U}'_{\mathrm{on}}} q_{ij} \prod_{<i,j>\in^{-}\mathbf{U}'_{\mathrm{off}}} (1-q_{ij})] \prod_{<i,j>\in\mathcal{C}(R,V_{\ell'})} (1-q_{ij}). \tag{A.11}$$

We see that these proposal probabilities are very hard to compute, because of the exponential number of combinations $\mathbf{U} \in \Psi(R|\mathbf{X})$ which produce the same connected component R. In what follows, we will show how the ratio of the proposal probabilities can be simplified to obtain the desired equation (2.36).

We obtain

$$\frac{q(\mathbf{X}' \to \mathbf{X})}{q(\mathbf{X} \to \mathbf{X}')} = \frac{q(R|\mathbf{X}')}{q(R|\mathbf{X})} \cdot \frac{q(\mathbf{X}_R = \ell|R, \mathbf{X}')}{q(\mathbf{X}_R = \ell'|R, \mathbf{X})}. \tag{A.12}$$

Dividing eqn. (A.10) by eqn. (A.11), we obtain the ratio

$$\frac{q(R|\mathbf{X})}{q(R|\mathbf{X}')} = \frac{\prod_{<i,j>\in\mathcal{C}(R,V_\ell)} (1-q_{ij}) \sum_{\mathbf{U}\in\Psi(R|\mathbf{X})} [q(R|\mathrm{CP}(\mathbf{U}|\mathbf{X})) \prod_{<i,j>\in\mathbf{U}_{\mathrm{on}}} q_{ij} \prod_{<i,j>\in^{-}\mathbf{U}_{\mathrm{off}}} (1-q_{ij})]}{\prod_{<i,j>\in\mathcal{C}(R,V_{\ell'})} (1-q_{ij}) \sum_{\mathbf{U}'\in\Psi(R|\mathbf{X}')} [q(R|\mathrm{CP}(\mathbf{U}'|\mathbf{X}')) \prod_{<i,j>\in\mathbf{U}'_{\mathrm{on}}} q_{ij} \prod_{<i,j>\in^{-}\mathbf{U}'_{\mathrm{off}}} (1-q_{ij})]} \tag{A.13}$$

The sums in the numerator and denominator of the above equation are equal because of the following

Observation 1. *For any* $\mathbf{U} \in \Psi(R|\mathbf{X})$*, there exists exactly one* $\mathbf{U}' \in \Psi(R|\mathbf{X}')$ *such that*

$$\mathrm{CP}(\mathbf{U}|\mathbf{X}) = \mathrm{CP}(\mathbf{U}'|\mathbf{X}') \tag{A.14}$$

and

$$\mathbf{U}_{\text{on}} = \mathbf{U}'_{\text{on}}, ^{-}\, \mathbf{U}_{\text{off}} =^{-} \mathbf{U}'_{\text{off}}. \tag{A.15}$$

That is, $\mathbf{U}$ *and* $\mathbf{U}'$ *differ only in the cuts* $\mathcal{C}(R, V_\ell)$ *and* $\mathcal{C}(R, V_{\ell'})$.

The cancelation of the sums in equation (A.13) gives

$$\frac{q(R|\mathbf{X})}{q(R|\mathbf{X}')} = \frac{\prod_{<i,j>\in\mathcal{C}(R,V_\ell)}(1-q_{ij})}{\prod_{<i,j>\in\mathcal{C}(R,V_{\ell'})}(1-q_{ij})}. \tag{A.16}$$

Note that the proof holds for arbitrary design of q_{ij}, arbitrary design of $q(R|\text{CP}(\mathbf{U}|\mathbf{X}))$ on arbitrary graphs.

We will now consider the split and merge cases (see Section 2.2.2) which have two possible paths between the states $\mathbf{X}$ and $\mathbf{X}'$.

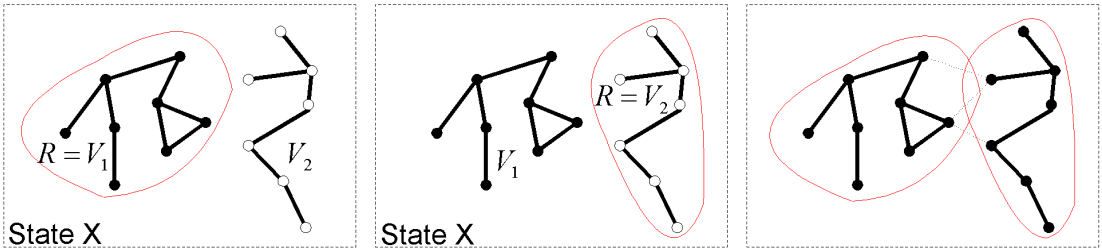


Figure A.1: State $\mathbf{X}$ has two subgraphs V_1 and V_2 which are merged in state $\mathbf{X}'$. There are two paths between $\mathbf{X}$ and $\mathbf{X}'$. One is to choose $R = V_1$ and the other is to choose $R = V_2$.

Without loss of generality, we can assume the states are $\mathbf{X} = (V_1, V_2, V_3, ..., V_n)$ and $\mathbf{X}' = (V_{1+2}, V_3, V_4, ..., V_n)$ with $V_{1+2} = V_1 \cup V_2$. The proposal probability $q(\mathbf{X} \to \mathbf{X}')$ is the sum of proposal probabilities in the two paths.

- Path 1: Choose $R = V_1$ in $\mathbf{X}$ and merge it to V_2 (i.e. choosing $\mathbf{X}_R = 2$) to reach $\mathbf{X}'$, and conversely, choose $R = V_1 \subset V_{1+2}$ in $\mathbf{X}'$ and split it to a new color V_1 (i.e. choosing $\mathbf{X}_R = 1$) and the rest $V_{1+2} \backslash V_1$ is named V_2.

- Path 2: Choose $R = V_2$ in $\mathbf{X}$ and merge it to V_1 (i.e. choosing $\mathbf{X}_R = 1$) to reach $\mathbf{X}'$, and reversely, Choose $R = V_2 \subset V_{1+2}$ in $\mathbf{X}'$ and split it to a new color V_2 and the rest $V_{1+2} \backslash V_2$ is named V_1.

We can summarize these paths into

$$\frac{q(\mathbf{X} \to \mathbf{X}')}{q(\mathbf{X}' \to \mathbf{X})} = \frac{q(R = V_1|\mathbf{X})q(\mathbf{X}_R = 2|R = V_1, \mathbf{X}) + q(R = V_2|\mathbf{X})q(\mathbf{X}_R = 1|R = V_2, \mathbf{X})}{q(R = V_1|\mathbf{X}')q(\mathbf{X}_R = 1|R = V_1, \mathbf{X}') + q(R = V_2|\mathbf{X}')q(\mathbf{X}_R = 2|R = V_2, \mathbf{X}')}. \tag{A.17}$$

Then we have the following two observations:

Firstly, from eq (A.13), we know,

$$\frac{q(R = V_1|\mathbf{X})}{q(R = V_1|\mathbf{X}')} = \frac{1}{\prod_{<i,j>\in\mathcal{C}(V_1,V_2)}(1 - q_{ij})} = \frac{q(R = V_2|\mathbf{X})}{q(R = V_2|\mathbf{X}')} \tag{A.18}$$

Secondly, once R is selected from $\mathbf{X}$ (or $\mathbf{X}'$), its new label follows a label proposal probability which depends on the partition of all other vertices $V \backslash R$ which are the same for both $\mathbf{X}$ and $\mathbf{X}'$. Note that all permutations of the labelings are considered equivalent. Therefore we have

$$\frac{q(\mathbf{X}_R = 2|R = V_1, \mathbf{X})}{q(\mathbf{X}_R = 1|R = V_1, \mathbf{X}')} = \frac{q(\mathbf{X}_R = 1|R = V_2, \mathbf{X})}{q(\mathbf{X}_R = 2|R = V_2, \mathbf{X}')}. \tag{A.19}$$

Therefore, we can write the ratio in both paths as $\frac{q(\mathbf{X}_R=\ell'|R,\mathbf{X})}{q(\mathbf{X}_R=\ell|R,\mathbf{X}')}$. Plug eqns. (A.18) and (A.19) in eq. (A.17), we have the result.

The split case is the reverse of the merge case and thus both cases are proven in the above discussion.

References

[1] A. Barbu and S.C. Zhu, "Graph partition by Swendsen-Wang cuts", *Proc. Int'l Conf. on Computer Vision*, Nice, France, 2003.

[2] A. Barbu and S.C. Zhu, "Multigrid and multi-level Swendsen-Wang cuts for hierarchic graph partition", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Washington DC, 2004.

[3] P. Belhumeur, "A Bayesian Approach to Binocular Stereopsis" *IJCV* 19(3)237-260, 1996

[4] S. Birchfield and C. Tomasi, "A Pixel Dissimilarity Measure that is Insensitive To Image Sampling", *IEEE Trans. on PAMI*, **20**(4):401-406, April 1998

[5] M. Black and D.J. Fleet, "Probabilistic detection and tracking of motion boundaries", *IJCV* 38(3) 231-245, 2000

[6] Black, M.J. and Jepson, A.D. (1996). "Estimating optical flow in segmented images using variable order parametric

[7] S. A. Barker, A. C. Kokaram, and P. J. Rayner. "Unsupervised segmentation of images", *SPIE Conf. on Bayesian Inference for Inverse Problems*, pp.200-211, July 1998.

[8] A. Blake, A. Zisserman, "Visual Reconstruction", em MIT Press, 1987

[9] Y. Boykov, O. Veksler, and R. Zabih. "Fast approximate energy minimization via graph cuts". *IEEE Trans. on PAMI* vol. 23, no. 11, pp 1222-1239, 2001

[10] C. Cooper, A. Frieze. "Mixing properties of the Swendsen-Wang process in classes of graphs", *Random Structures and Algorithms* vol. 15, no. 3-4, pp. 242-261, 1999.

[11] R.G. Edwards, A.D. Sokal, "Generalization of the Fortuin-Kasteleyn-Swendsen-Wang representation and Monte Carlo algorithm", *Phys. Rev. Lett.*, 38, pp 2009-2012,1988

[12] P. Diaconis and L. Saloff-Coste, "What do we know about the Metropolis Algorithm?", *J.of Computer and System Sciences* **57**, 20-36.

[13] O. Faugeras, J. Gomes, and R. Keriven, "Variational principles in computational stereo" *Geometric Level Set Methods in Imaging, Vision and Graphics* S. Osher and N. Paragios, editors, Springer-Verlag, 2003.

[14] C. Fox, G.K. Nicholls, "Exact MAP States and Expectations from Perfect Sampling",*Proc. of 20th Int'l Workshop on Bayesian Inference and Maximum Entropy Methods*, France, 2000.

[15] A. Fridman, "Mixed Markov Models", *Proc Nat Acad Sci*, 100 (14) 8092-8096, 2003

[16] Y. Gdalyahu, D. Weinshall, M. Werman, "Self-organization in vision: stochastic clustering for image segmentation, perceptual grouping and image database", *IEEE. Trans. on PAMI*, vol. 23, no.10, pp.1053-1074, 2001.

[17] S. Geman, D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images", *IEEE Trans. on PAMI*, vol. 6, pp. 721-741, 1984.

[18] C.J. Geyer and E.A. Thompson, "Annealing Markov chain Monte Carlo with applications to ancestral inference", *J. Am. Statist. Assoc.* **90** 909-920.

[19] Gilks, W.R. and Roberts, G. O. "Strategies for improving MCMC", in (Gilks, W.R. eds) *Markov Chain Monte Carlo in Practice*, Chapman & Hall/CRC .

[20] J. Goodman and A.D. Sokal, "Multigrid Monte Carlo method. Conceptual foundations", *Physical Review D* **40**, 2035-2072.

[21] V. Gore and A. Sinclair, "The Swendsen-Wang process does not always mix rapidly", *J. Stat. Phys.*, 97(1-2):67-86, 1999.

[22] Gore, V. and Jerrum, M (1997). "The Swendsen-Wang process does not always mix rapidly", *Proc. 29th ACM Symp. on Theory of Computing* 674-681.

[23] P. J. Green, "Reversible jump Markov chain Monte Carlo computation and Bayesian model determination", *Biometrika*, vol. 82, 711-732, 1995.

[24] C. Guo, S.C. Zhu and Y.N. Wu, "A Mathematical Theory of Primal Sketch and Sketchability", *ICCV* 2003.

[25] W.K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications", *Biometrika*, vol. 57, pp.97-109, 1970.

[26] D. Higdon, "Auxiliary variable methods for Markov chain Monte Carlo with applications", *J. Am. Statist. Assoc.* **93**, 585-595.

[27] T. Hofmann, J.M. Buhmann, "Pairwise data clustering by deterministic annealing", *IEEE Trans. on PAMI*, vol. 19, no. 1, pp. 1-14, 1997.

[28] M. Huber. "A bounding chain for Swendsen-Wang." *Random Structures and Algorithms*, vol 22, no 1,pp 43-59, 2002

[29] E. Ising, "Beitrag zur theorie des ferromagnetismus", *Zeitschrift für Physik*, 31, pp253-258, 1925.

[30] A.K. Jain, R. Dubes, *Algorithms for Clustering Data*, Prentice Hall, Englewood Cliffs, NJ, 1988.

[31] S. Kirkpatrick, C. Gelatt, M. Vecchi, "Optimization by simulated annealing", *Science*, 220(4598), pp.671-680, 1983.

[32] V. Kolmogorov, R. Zabih, "What energy functions can be minimized via graph cuts?", *Proc. ECCV*, pp. 65-81. vol. 3, Copenhagen, Denmark, 2002.

[33] M. Lin and C. Tomasi, "Surfaces with occlusions from layered stereo". *IEEE Trans. on PAMI*, 26 (8), 710–717, 2004

[34] Liu, J.S. and Wu, Y.N. (1999). "Parameter expansion scheme for data augmentation", *J. Am. Statist. Assoc.* **94**.

[35] Liu, J.S. "Monte Carlo strategies in scientific computing", Springer, NY.

[36] D. Martin, C. Fowlkes, D. Tal, J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics", *Proc. of Int'l Conf. on Comp. Vision*, Vancouver, Canada, July, 2001.

[37] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, E. Teller,"Equations of the state calculations by fast computing machines", *J. Chem. Physics*, 21, pp.1087-1091,1953

[38] J.Pearl, "Reverend Bayes on inference engines: A distributed hierarchical approach", *Proc. Conf. Nat. Conf. AI* Pittsburgh, PA, pp. 133-136, 1982.

[39] J.Pearl, "Fusion, propagation, and structuring in belief networks", *Artif. Intell.*, **29**, pp. 241-288, 1986

[40] R.B. Potts, "Some generalized order-disorder transformations", *Proceedings of the Cambridge Philosophic Society*, 48, pp.106-109, 1953.

[41] J.G. Propp, D.B. Wilson. "Exact Sampling with Coupled Markov Chains and Applications to Statistical Mechanics" *Random Structures and Algorithms*, vol. 9 no. 1-2, pp. 223-252, 1996.

[42] J. Puzicha, T. Hofmann, J.M. Buhmann, "A theory of proximity based clustering: structure detection by Optimization", *Pattern Recognition*, **33**, no.4, pp.617-634, 1999.

[43] A. Rosenfeld, R. Hummel and S. Zucker, "Scene labeling by relaxation operations",*IEEE Transactions on Systems, Man and Cybernetics* **6**,pp 420-433.

[44] S. Roy, I. Cox, "A maximum-flow formulation of the n-camera stereo correspondence problem", *Proc. ICCV*, Bombay, India, 1998.

[45] E. Saund, "Perceptual organization of occluding contours of opaque surfaces", *Computer Vision and Image Understanding* 1999 October; 76 (1): 70-82.

[46] D. Scharstein and R. Szeliski. "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms", *IJCV*, 2002

[47] D. Scharstein and R. Szeliski, "High-accuracy stereo depth maps using structured light", *CVPR* 2003

[48] J. Shi, J. Malik, "Normalized cuts and image segmentation", *IEEE Transactions on PAMI*, **22** no 8, pp. 888-905, 2000.

[49] J. Sun, H. Shum, and N.N. Zheng, "Stereo matching using belief propagation", *IEEE PAMI*, vol. 25, no. 7, 2003.

[50] R.H. Swendsen and J.S. Wang, "Nonuniversal critical dynamics in Monte Carlo simulations", *Physical Review Letters*, **58** no. 2, pp.86-88, 1987.

[51] Tanner, M. A. and Wong, W.H. (1987), "The calculation of posterior distributions by data augmentation (with discussion)", *J. Amer. Stat. Assoc.*, 82(398):528-540.

[52] H. Tao, H. S. Sawhney and R. Kumar, "A global matching framework for stereo computation", *ICCV*, 2001

[53] M. F. Tappen and W. T. Freeman, "Comparison of Graph Cuts with Belief Propagation for Stereo, using Identical MRF Parameters" *ICCV* 2003

[54] Z.W. Tu and S. C. Zhu, "Image segmentation by data-driven Markov chain Monte Carlo", *IEEE Trans. on PAMI*, **24**, no. 5, 2002.

[55] Z.W. Tu and S.C. Zhu, "Parsing images into region, curves, and curve processes", *Submitting to IJCV*, Short version appeared in *Proc. of ECCV*, 2002.

[56] Z. Tu and S.C. Zhu, "Parsing Images into Regions, Curves, and Curve Groups", IJCV(under review), http://www.stat.ucla.edu/∼ztu/.

[57] J. Wang, et al. "Relationship between ventral stream for object recognition and dorsal stream for spatial vision: an fMRI and ERP study", *Human Brain Mapping*, 8, pp.170-181, 1999.

[58] D.M. Waltz, "Generating semantic descriptions from drawings of scenes with shadows",*The psychology of computer vision*, pp 19-92, New York, 1972.

[59] G. Winkler, *Image Analysis, Random Fields and Dynamic Monte Carlo Methods*, page 171-173, Springer-Verlag, 1995.

[60] U. Wolff, "Collective Monte Carlo updating for spin systems", *Physical Review Letters*, vol. 62, no. 4, pp. 361-364, 1989.

[61] Z. Wu and R. Leahy, "An optimal graph theoretic approach to data clustering: theory and application to image segmentation", *IEEE Trans. on PAMI*, **15**, pp.1101-1113, 1993.

[62] J.S. Yedidia, W.T. Freeman, and Y. Weiss, "Generalized belief propagation", *MERL Report TR-2000-26*, 2000.

[63] S.C. Zhu and A.L. Yuille, "Region competition: unifying snake/balloon, region growing and Bayes/MDL/energy for multi-band image segmentation", *IEEE Trans. on PAMI*, **18**, no. 9, pp.884-900, 1996.

[64] G. Li and S. W. Zucker, "A Differential Geometrical Model for Contour-Based Stereo Correspondence" *IEEE Workshop on Variational, Geometric, and Level Set Methods in Computer Vision*, Nice, 2003.