Image Inpainting by Global Structure and Texture Propagation

HUANG, Ting

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of
Master of Philosophy
in
Information Engineering

September 2008

Abstract

Abstract of thesis entitled:

Image Inpainting by Global Structure and Texture Propagation

Submitted by HUANG, Ting

for the degree of Master of Philosophy

at the Chinese University of Hong Kong in 2008

Image inpainting is a technique to repair damaged images or modify images in a non-detectable form. It has a wide range of applications, and is currently an active and challenging research area of computer graphics and computer vision.

Structure information and texture information are the two critical ingredients of image inpainting. They are both required to produce a high-quality output. In the literature, many approaches are either purely texture-based or purely structure-based. Others deal with structures and textures using different schemes separately, but can't handle them both well. Considering these aspects, and motivated by the *exemplar-based* approaches, we propose a novel image inpainting algorithm, combining both structure and texture information under a unified framework, which automatically finds a balance between the structure and texture.

After removing objects from an image, our approach fills in the regions using patches taken from the known region. The filling process is formulated

as a discrete global optimization problem using Markov random fields (MRFs) whose energy function measures both structure and texture consistencies. Belief propagation (BP) is utilized to solve the minimization problem.

One big challenge in using standard BP is that its computational complexity is the square of the number of label candidates. For a typical 481×321 image, the number of label patches is at the order of 10^4. To reduce the large number of label candidates, we present a coarse-to-fine scheme where two BPs run with much smaller numbers of label candidates instead of one BP running with a large number of label candidates. Experimental results demonstrate the excellent performance of our algorithm over other related algorithms.

摘要

圖像修補 (Image Inpainting) 是計算機視覺和圖形學中一個非常活躍的研究領域，在圖像恢復和圖像修改的研究及應用中被廣泛採用，比如它可以用於恢復舊照片或被損照片。

近 些年來，圖像修補得到眾多研究人員的重視，各種相關算法不斷發展和進步。對於圖像修補，結構信息和紋理信息是兩個最重要的因素，因為人類視覺系統對於圖像 修補區域與周圍區域在紋理和結構上的一致性與連續性更加敏感和關注。因而，高質量的圖像修補算法必須充分利用和保持圖像的紋理和結構信息。在以前的眾多相 關算法中，大部分都只考慮了圖像的紋理信息（比如基於紋理合成的算法），有些則用不同的方式分別處理這兩種信息，但並不能很好地處理這兩者之間的關係。基 於此，我們提出了一種新的基於樣本的圖像修補算法，將結構信息和紋理信息放到統一的框架下處理，自動地平衡兩者之間的相對關係以實現修補後圖像紋理和結構 的連續性。

在我們的算法中，我們用圖像已知區域中的圖像片段(patch)來填補未知區域。整個填補過程是基於馬爾可夫隨機場 (Markov Random Fields) 全局優化模型來完成的。我們提出的優化目標函數同時包含了圖像的結構和紋理信息。這種結構紋理統一化的模型使我們的優化過程能同時實現結構 紋理一致性。很多優化算法可以求解基於馬爾可夫隨機場的目標函數。這裡，我們採用 Belief Propagation (BP) 來解決這個優化問題。

然而，大量的計算開銷是 BP 算法應用於實際問題的一個重要限制。其中備選樣本數的大小是最重要的影響因子，因為標準 BP 算法的算法複雜度正比於備 選樣本數的平方。在圖像修補中，對於一幅 481×321 的圖像，圖像樣本片段有上萬個，這使得標準 BP 算法不能直接運用到我們的模型上。因此，我們提出了 一個逐層細化 BP 算法。這個算法的核心思想就是利用多次小樣本 BP 代替一次大樣本 BP，從而極大地降低了算法的時間複雜度。我們在大量圖像數據上進行了測 試，實驗結果表明，我們算法明顯優於其他相關算法。

Acknowledgments

My first thanks go to my supervisors, Prof. Tang Xiaoou and Prof. Liu Jianzhuang. I'm greatly impressed by their passion for research and students. Xiouou has always been an enthusiastic advisor, providing encouragement, insight and a valuable big picture. He does not only care about my research but also my career and my future. His kind suggestions always guide me in the right direction. Jianzhuang shows me the real meaning of hardworking. He's always conscientious and energetic, and helps me grow up in research. I'm so lucky to have these excellent supervisors.

I am and will ever be proud that I am a member of MMlab. We form a real family here. I am greatly indebted to Xiaodai, Weige, Afeng, and Liu Ming. You're all great guys. I've been always enjoying the happy times here with you, doing sports, hiking, cooking, and shopping together. Moreover, you always stand by my side at any time whenever I need you. Thanks, my dear friends. You guys make my life here wonderful and colorful. Thanks Yueming for your help and kind suggestions. I also extend my thanks to the other MMlabers: Pengfei, Yingze, Chunjing, Zhenguo, Yiwen, Deli, Chen Mo, Dahua, Zhao Feng, Huanzi, Li Yun, Zhifeng, Xiaowei, Chen Yu, Kaiming... We are a big family here and I'll always remember the time with you.

I'd like to give my special thanks to my dear friends, Sky, Peizhi, Leo, Dolphin, Qi You, Xiaoqiang, Linxing, Xpp...

Last but not least, I thank my family, Mom and dad for giving my life in the first place, for educating me, for your unconditional supports. Ganma and

Gandie, you always stand by my side and treat me as your own son. My Uncle Liu, Aunt Wu and Aunt Fang, you always support me and encourage me to pursue my own life.

I love you all.

Contents

1 Introduction **1**

1.1 Related Area 2

1.2 Previous Work 4

1.3 Proposed Framework 7

1.4 Overview 8

2 Markov Random Fields and Optimization Schemes **9**

2.1 MRF Model 10

2.1.1 MAP Understanding 11

2.2 Belief Propagation Optimization Scheme 14

2.2.1 Max-Product BP on MRFs 14

2.2.2 Sum-Product BP on MRFs 15

3 Our Formulation **17**

3.1 An MRF Model 18

3.2 Coarse-to-Fine Optimization by BP 21

3.2.1 Coarse-Level Belief Propagation 23

3.2.2 Fine-Level Belief Propagation 24

3.2.3 Performance Enhancement 25

4 Experiments **27**

4.1 Comparison 27

4.2 Failure Case 32

5 Conclusion **35**

Bibliography **37**

List of Figures

1.1 Two examples of image inpainting from [4]: restoration of a damaged photograph and removal of superimposed text. 2

2.1 A figure from [39], describing a square lattice Markov Random Field. The filled-in circles represent the nodes with observed information y_i; the empty circles represent the "hidden" scene nodes with underlying information x_i. 10

3.1 Graph construction. The image is sparsely sampled with a horizontal spacing sh and a vertical spacing sv. The samples in the target region Ω construct the nodes of the graph G. The overlapping part (region 1) between x_i and the source region provides the data cost for x_i and the overlapping part (region 2) between x_i and x_j gives the consistency cost for (x_i, x_j). . . 19

3.2 An example of the map of the confidence values. (a) The input "bungee" image. (b) The mask (in green) of the boy to be removed. (c) The confidence map in the mask, in which the brighter a pixel is, the larger the confidence value. 20

3.3 An example of our coarse-to-fine BP optimization. (a) The coarse-level result of the first BP step, with blurring effect. (b) The fine-level result of the second BP step, with detailed information filling in. 24

4.1 The comparative results on the "bungee" image. (a) The result of the greedy approach in [12]. (b) The result of BP without confidence and structure terms (i.e., C and E_2^s), as in [30]. (c) Our result. 28

4.2 More comparative results. Each column contains a set a experiment. From top to bottom, respectively, each row contains the original images, the masked images, the results obtained by the greedy algorithm in [12], the results of BP without confidence and structure terms (i.e., C and E_2^s) as in [30], and our results. (The results can be seen more clearly on the screen with enlarged images.) 29

4.3 The comparison with [34]. Each column contains a set a experiment. From top to bottom, respectively, each row contains the original images, the masked images, the results obtained by [34] with manual guidance, and our results. 31

4.4 A failure case of removing the bug. (a) original image; (b) our result; (c) the result obtained by the greedy algorithm in [12]; (d) the result of BP without confidence and structure terms (i.e., C and E_2^s) as in [30]. 32

4.5 More results by our algorithm. Each row contains a set of experiments. From left to right: original images, masked images, and the results. 33

4.6 More results by our algorithm. Each row contains a set of experiments. From left to right: original images, masked images, and the results. 34

Chapter 1

Introduction

Image inpainting [16, 35, 26, 4, 1, 11] (also known as image completion or filling-in), is the process of filling in the gap of missing data in a form that is non-detectable for an ordinary observer [4]. Its origin can be tracked back to as early as Renaissance, when medieval artwork started to be restored as "up-to-date". This practice is called retouching or inpainting.

Traditionally, inpainting has been done by professional artists. In photography and film industry, it is used to revert deterioration (e.g., cracks in photographs or scratches and dust spots in film), or to add or remove elements (e.g., removal of stamped date and red-eye from photographs, the infamous "airbrushing" of political enemies [26]).

Nowadays, digital technique is a widespread way of automatically performing inpainting. The applications are numerous, ranging from the restoration of damaged paintings and photographs to the removal/replacement of selected objects. Some examples are shown in Fig. 1.1. Image inpainting is currently an active and challenging research area of computer graphics and computer vision.

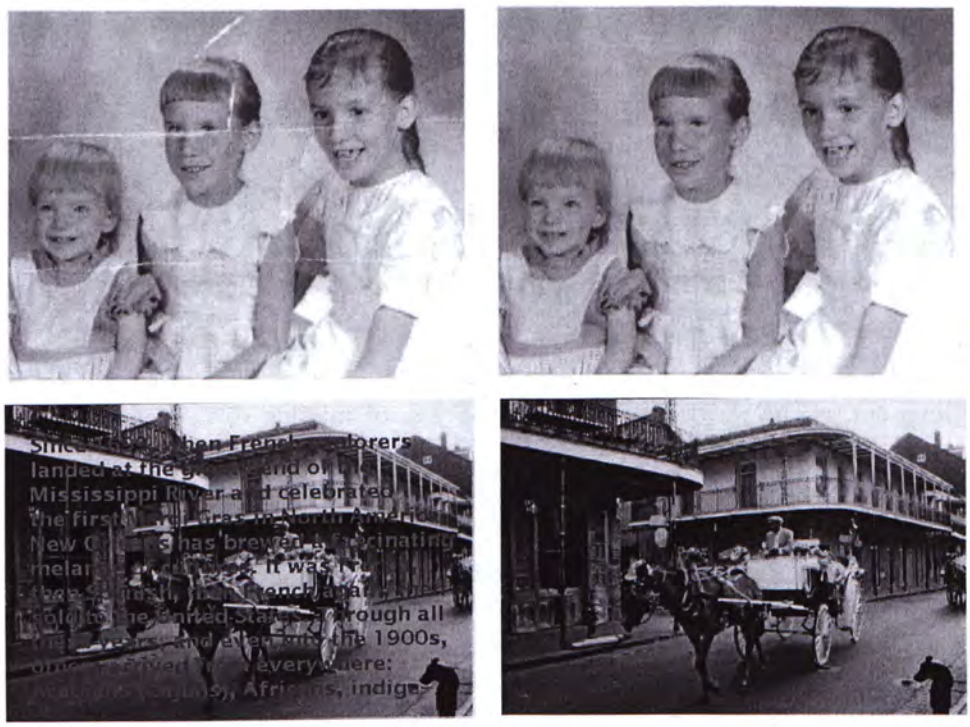

Figure 1.1: Two examples of image inpainting from [4]: restoration of a damaged photograph and removal of superimposed text.

1.1 Related Area

There're mainly four groups of works are related to image inpainting in the literature: film restoration, texture synthesis, disocclusion and contour completion.

The first one deals with the restoration of damaged films [27, 28, 25]. Its basic idea is to use motion estimation and the information from past/future frames to restore the current one, by copying the right pixels into the gap. This general approach cannot be directly used when handling still images. Moreover, it cannot deal with movies where the region to be filled is static with respect to its background (e.g., a logo on a shirt), or the region ranges a large number of frames, since the motion cannot be estimated and the past/future

frames do not provide new information for the loss data.

Another very active area related to the work here is texture synthesis [15, 14, 21, 22]. The basic idea is to select a texture and synthesize it to fill the region. The most notable approaches are based on Markov Random Fields [15, 36], which are widely used in many computer vision areas, including image inpainting. Texture synthesis is highly related to image inpainting. Actually, as we'll see later, texture is one of the two most important features for image inpainting. Texture synthesis approaches focus purely on texture, and often require the user to specify the texture to be copied. Thus, these methods cannot deal with images in which the region to be filled covers several different structures, but they do provide a helpful insight for image inpainting.

In the group of disocclusion algorithms, a pioneer work is described in [33]. Its basic idea is to connect T-junctions at the same gray-level with elastica minimizing curves. An extended work [31] proposed a general variational formulation for disocclusion. Since removing occluding objects can be considered as filling-in the occluded region, disocclusion is quite analogous to image inpainting. They both try to restore missing visual information for a 2D image, and mathematically, they can both be classified into the same category of inverse problems. The differences lie in both their goals and focuses.

Last, a number of fundamental works on the topic of edge and contour completion have been reported in the literature [19, 37], and they are quite related to image inpainting. Edge completion methods find the most likely smooth curves to connect edge elements, usually by minimizing a function based on the shape of the curve. The likelihood that any two elements are connected defines a field for each element, and completion is performed by optimizing over the whole fields. The completed contours partition the whole image into several regions, each with similar properties (e.g., texture) within itself. This reveals the underlined structure of the image, which is another critical ingredient for inpainting.

1.2 Previous Work

First, as stated above, the two most important features for image inpainting are structures and textures. Remember that the goal of inpainting is to produce a revised image in which the inpainted region is seamlessly merged into the image ("non-detectable for an typical observer"). Here, the structure plays a critical role for human vision. The inpainting process should fill in structures naturally, and merge different structures "seamlessly". On the other hand, the texture reflects details of an image, and represents a notable property within one structure. Highly consistent and detailed textures are required for filling in the region in a "non-detectable" form. As described above, contour completion and texture synthesis purely focus on one of these two aspects respectively. In the following, we'll review and evaluate inpainting algorithms in terms of these two aspects as well as computation efficiency.

In the literature, a number of inpainting algorithms focus on filling in the target region by propagating linear structures (called *isophotes* in inpainting) via diffusion. The terminology of *Image Inpainting* was first introduced by Bertalmio, Sapiro, Caselles, and Ballester [4]. Inspired by the traditional inpainting process of artists, they proposed a successful digital inpainting scheme, by propagating image Laplacians continuously in the level-lines (isophotes) directions. The algorithm introduces the importance of propagating both gray levels (photometry) and the gradient direction (geometry) of the image in a band surrounding the hole. It is based on partial derivation equation (PDE) and rooted in the Navior-Stokes equation in fluid dynamics [3]. A variational framework based on interpolation of the image gray levels and gradient directions is presented in [1]. The method in [11] called Curvature-Driven Diffusions (CCD) is proposed to handle curve structures by incorporating Euler's elastica. All these methods focus purely on structures and work at pixel level. They perform well for small or thin gap fillings, such

as speckles, scratches, and text overlays. They work convincingly as restoration algorithms in text removal and retouching old photos. However, they pay little attention to textures, and introduce blurring effects due to their diffusion process, which become noticeable when filling large regions.

Inspired by texture synthesis techniques [15, 14, 22], another group of algorithms perform inpainting based on pure texture synthesis. Amongst these algorithms, the *exemplar-based* techniques are most effective, in which one fills the unknown region by simply sampling and copying pixels or texture patterns from the source part of the image. Starting with the notable work in [15], this method has been mainly used for texture synthesis, until recently been extended to image inpainting as well [20, 2, 7].

Recent exemplar-based approaches also incorporate structure information. Jia et al. [24] fills the target region based on texture segmentation and a tensor-voting algorithm for smoothly linking structures across the region. This approach has a clear advantage that it connects curved structures by the explicit completion of texture contours, which is followed by a texture synthesis process. On the other hand, the algorithm requires an expensive segmentation step, and a hard decision about what makes a boundary between two textures. Our approach avoids both issues through global optimization.

Bertalmio et al. [5] decomposed the image into structure and texture components, one of which is processed by inpainting and the other by texture synthesis. The final result is the sum of the two completed components. This approach is still limited to small gaps, since the diffusion process causes blurring. In [13], the completion process is iteratively guided by the smooth approximation in a coarse-to-fine manner, interleaved by adaptive example fragments. It requires assistance from the user as the "points of interest" and may introduce blurring artifacts too. The long computational time is another limitation of [5] and [13] (as reported processing may take between 83 and 158 minutes on a 384×256 image).

In [12], a fast algorithm is presented to simultaneously propagate texture and structure in a greedy scheme, by placing emphasis on the order of image synthesis procedure. Like the method in [13], it fills in the region more effectively by processing synthesis with some automatic guidance. The guidance here is a computed confidence map, which determines the synthesis ordering and may improve the quality of completion by preserving some salient structures. On the other hand, as the confidence map is based on heuristics and ad hoc principles, it may not apply in general cases, and often produces artifacts. Moreover, once a missing block has been assigned some source patch, the block cannot change its appearance thereafter. This reveals the greediness of the approach, which again leads to visual inconsistencies.

Contrary to the greedy methods, recently some approaches formulate the image completion as discrete global optimization problems solved using Belief Propagation (BP) [34], [30]. In [34], user assistance is required to identity structure information in and surrounding the unknown region. The algorithm first fills in the area along the manually added structures. This process is called structure propagation. After structure propagation, the remaining unknown regions have been partitioned into several disjoint subregions by the user-specified curves, each with usually one known subregion around. These relationships between a pair of corresponding known/unknow subregions are critical for the next step - texture propagation. The approach produces quite convincing results, especially when the structure is rather complex. On the other hand, it's built upon user guidance and is not actually an automatic inpainting algorithm.

In [30], a recent exemplar-based approach is presented based on Markov Random Fields. Komodakis et al. formulated image completion as an optimization problem with an objective function. Image completion is automatically solved using an efficient BP, called Priority-BP. The computational complexity of standard BP is the square of the number of label candidates.

In order to reduce the intolerable computational cost of BP caused by the large number of labels, they proposed two cooperative optimization methods over standard BP, i.e.,"dynamic label pruning" and "priority-based message scheduling". The basic idea is to discard labels that are unlikely to be optimal, and use some greedy-like scheme to speed up BP's convergence. These methods highly reduced the computational cost, but are not proofed to find the optimal solution generated by the standard BP. Moreover, structure information is not included in their frame.

1.3 Proposed Framework

As stated above, the structure and texture are the two most critical ingredients of image inpainting. They are both required to produce a high-quality output. In the literature, many approaches are either purely texture-based or purely structure-based. Others deal with structures and textures using different schemes separately. Considering these aspects, and motivated by the *exemplar-based* approaches, which are the most efficient group of inpainting algorithms, we propose a novel image inpainting algorithm, combining both structure and texture information under a unified framework.

After removing objects from an image, our approach fills in the regions using patches taken from the known region. The filling process is formulated as a discrete global optimization problem using Markov random fields (MRFs), whose energy function measures both structure and texture consistency. Belief propagation (BP) is utilized to solve the minimization problem. One challenge in using standard BP is that its computational complexity is the square of the number of label candidates. For a typical 481×321 image, the number of label patches is at the order of 10^4. To handle the high computational complexity, we present a two-step coarse-to-fine scheme, where two BPs run with much smaller numbers of label candidates instead of one BP running with a large

number of label candidates. Experimental results demonstrate the excellent performance of our algorithm over other related algorithms, with reasonable time consumption.

1.4 Overview

The rest of this dissertation is organized as follows. Chapter 2 introduces Markov Random Fields model and Belief Propagation. The MAP understanding of MRF model and two standard BP algorithms are presented. Chapter 3 then introduces our formulation. We propose a unified framework for structure and texture information, and a coarse-to-fine BP approach is formulated to make our algorithm practical. Experimental results are presented in Chapter 4. We compare our results with two most recent related algorithms. The last chapter gives the conclusion and future work.

This dissertation is based on our publication of ACMM07' [23].

Chapter 2

Markov Random Fields and Optimization Schemes

Despite the recent success and rapid advances of the Markov random fields (MRFs) model, which provides a robust and unified framework for many early computer vision tasks, such as image restoration, image segmentation, stereo vision, image matting and image completion, MRF model draws much attention of many researchers. MRFs formulate the problem as an energy minimization problem [18] based on probability inference. This formulation is justified in terms of maximum a posteriori (MAP) estimation of a Markov random field in the Bayesian framework. The energy minimization solution is regarded as the realization of a Markov random field, as well as a configuration of the candidate labels. To the best of our knowledge, for the general case, this energy minimization problem in discrete domain is NP-hard. The major obstacle of the optimization is the large computational cost owing to the high dimensional computing space. Many approaches have been proposed to solve the MRFs energy optimization. Simulated annealing, popularized in [18], can minimize an arbitrary energy function but is very slow. Iterated conditional modes (ICM) introduced in [6], iterates greedily to a local minimum depending on initial values. Recently developed algorithms, including graph cuts [10, 29, 8] and

belief propagation [38, 39, 32], approximately solve MRFs with the consideration about tradeoffs between accuracy and efficiency. Graph cuts and belief propagation both make the problem in discrete sense, and thus usually can only find a locally optimal solution, depending on the initialization and the convergence criteria.

2.1 MRF Model

Markov Random Fields were fist introduced into vision in [18], and provide attractive theoretical models for many computer vision problems. In such problems, we usually want to infer an underlying representation of what is really out there, from the data we're given, i.e., the pixel values of an image.

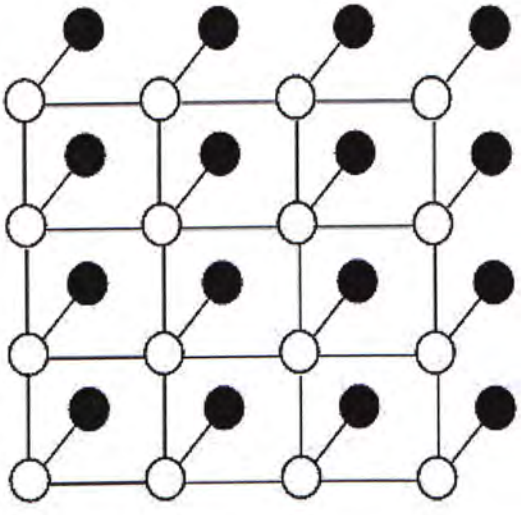

Figure 2.1: A figure from [39], describing a square lattice Markov Random Field. The filled-in circles represent the nodes with observed information y_i; the empty circles represent the "hidden" scene nodes with underlying information x_i.

Generally, assume that we observe some information about the image $Y =$

$[y_1, y_2, \ldots, y_n]^T$, and want to infer some other information about the underlying scene $X = [x_1, x_2, \ldots, x_n]^T$. Here, $P = \{1, 2, \ldots, n\}$ denotes the set of nodes, and each node i could represent a single pixel or a small patch of pixels. To infer X from Y, we further assume that there is some statistical dependency between X and Y at each node i, which could be described as a joint "evidence" function $\phi_i(x_i, y_i)$. Moreover, there must be some prior structure of X, otherwise the problem will be ill-posed. For an image, as the nodes are arranged in a two-dimensional grid, the scene information x_i at node i should be consistent with its neighboring variables x_j, as represented by a "compatibility" function $\psi_{ij}(x_i, x_j)$. Then the overall joint probability of an underlying scene x_i and an observed image could be:

$$P(X, Y) = \frac{1}{Z} \prod_{i \in P} \phi_i(x_i, y_i) \prod_{(i,j) \in \mathcal{N}} \psi_{ij}(x_i, x_j) \tag{2.1}$$

where Z is a normalization constant, and the product over (ij) is in neighborhood $\mathcal{N}$.

A graphical description is shown in Figure. 2.1.

2.1.1 MAP Understanding

The above equation (2.1) could be justified in terms of maximum a posteriori (MAP) estimation on the Bayesian framework. Using the same depiction, $Y = [y_1, y_2, \ldots, y_n]^T$ and $X = [x_1, x_2, \ldots, x_n]^T$ denote the known and underlying information respectively, which are both considered as random variables. The goal is to find an X that maximizes the posterior probability $P(X|Y)$ given the information Y, which results in the global optimal solution from statistic viewpoint. This definition is suitable for many computer vision tasks. In the following, we will use image denoising as an example to illustrate how to construct the probability in computer vision. In image denoising, Y is regarded as noised image intensities and X as restored pixel intensities. With the known

y_i configuration, our goal is to find a new label assignment x_i for each pixel i that satisfies some predefined criteria.

By using the Bayes' rule, we have

$$\begin{aligned} P(X|Y) &= P(Y|X)P(X)/P(Y) \\ &= P(Y|X)P(X), \end{aligned} \tag{2.2}$$

where $P(Y)$ can be removed since it is a constant with respect to X. Therefore, the MAP estimation is to find $\widehat{X}$ such that

$$\widehat{X} = \underset{X}{\operatorname{argmax}}\, P(Y|X)P(X). \tag{2.3}$$

As Y is already know, we'll drop y_i in the following equations. $P(Y|X)$ here is the likelihood function which can be represented by the joint "evidence" function $\phi_i(x_i)$, as

$$P(Y|X) \propto \prod_{i \in P} \phi_i(x_i) \tag{2.4}$$

In image denoising, assuming that the noise is an additive Gaussian white noise, the evidence function could be defined as

$$\phi_i(x_i) = \exp\big(-D_i(x_i)\big)$$

Here $D_i(x_i)$ is called the data penalty function, because it penalizes the inconsistency between output and observed data.

In the context of Markov assumption that each random variable x_i also depends on those in X that correspond to the neighbourhood nodes of node i, the prior probability $P(X)$ describes the structure of X, which can be represented by the "compatibility" function $\psi(x_i, x_j)$, as

$$P(X) \propto \prod_{(i,j) \in \mathcal{N}} \psi(x_i, x_j) \tag{2.5}$$

In the case of image denoising, we model the prior probability $P(X)$ by MRFs whose clique potentials involve pairs of neighboring pixels, which is

defined as

$$P(X) \propto \exp\big(-\sum_{(i,j)\in\mathcal{N}} S(x_i, x_j)\big),$$

where $\mathcal{N}(i)$ is the neighbourhood of i, and $s(x_i, x_j)$ is the clique potential, which is also commonly called smoothness penalty function to imposing the spatial homogeneous. Notice that $\psi(x_i, x_j) = \exp(-S(x_i, x_j))$. There have been some clique potential formats to enforce the intensity smoothness constraint for neighboring nodes, e.g., generalized Potts Model, $w_{ij}\cdot(1-\delta(f_i-f_j))$, ($\delta(\cdot)$ is the unit impulse function), and linear clique potential, $w_{ij}\cdot|f_i-f_j|$ [9]. The generalized Potts Model penalizes any neighbourhood pair of different labels while the penalization is independent on the difference. By contrary, some other costs of spatial incoherency in clique potential are proportional to the difference between neighbouring pixels' labels, such as quadratic label difference smoothness constraint and absolute label difference constraint. Meanwhile, to further avoid the outlier observed information which may leads to an unstable and incorrect result, data truncation and smoothness truncation terms are popularly used as well. That is to give a upper bound for each penalty term to prevent the cost being too large.

From (2.2), (2.4), and (2.5), we can get:

$$P(X|Y) \propto \prod_{i\in P} \phi_i(x_i) \prod_{(i,j)\in\mathcal{N}} \psi(x_i, x_j) \tag{2.6}$$

Notice that this is consistent with equation (2.1). Similarly, in the case of image denoising, the probability becomes

$$P(X|Y) \propto \exp\big(-\sum_{i\in P} D_i(x_i)\big) \exp\big(-\sum_{(i,j)\in\mathcal{N}} S(x_i, x_j)\big) \tag{2.7}$$

Obviously, by taking a negative logarithm, maximizing the above MAP estimation is equivalent to minimizing the following energy function:

$$E(X) = \sum_{i\in P} D_i(x_i) + \sum_{(i,j)\in\mathcal{N}} S(x_i, x_j). \tag{2.8}$$

The first term on the right hand side is called the data term and the second one is the smoothness term. By minimizing the energy $E(X)$, that is, the minimal cost realization of a Markov random field, X tends to be consistent with the observed data Y as well as maintain self continuities.

2.2 Belief Propagation Optimization Scheme

Belief Propagation (BP) is an efficient algorithm to solve inference problems based on passing local messages. Inference problems come up in many different scientific fields, e.g. statistical physics, computer vision, error-correcting coding theory and AI. So it is not that surprising that a good algorithm to solve such problems has been repeatedly re-discovered. In fact, different methods such as the forward-backward algorithm, the Viterbi algorithm, iterative decoding algorithms for gallager codes and turbocodes, Pearl's belief propagation algorithm for Bayesian networks, the Kalman filter, and the transfer-matrix approach in physics, are all special cases of the BP algorithm discovered in different scientific communities. Recently, BP has also begun to be used as an "engine" for low-level computer vision problems [17]. In the following of this section, we review two variations of belief propagation on MRFs.

2.2.1 Max-Product BP on MRFs

The max-product BP algorithm is used to approximate the MAP estimation on MRF problems. Normally, the formulation is defined in terms of probability distributions, as to maximize the posterior probability $P(X|Y)$. An equivalent formulation can be defined by taking negative logarithm over the probability, where the max-product becomes a min-sum. This formulation is less sensitive to numerical computation. Moreover, it directly corresponds to the energy function definition in computer vision.

The BP algorithm introduces variables $m_{ij}(x_j)$, which can intuitively be

understood as a "message" from node i to node j about what state node j should be in. The message m_{ij} is a vector of the same dimensionality as x_j, with each component being proportional to how likely node i believes that node j has the corresponding state x_j. The algorithm passes messages around the graph defined by the neighboring relationships $\mathcal{N}$. Messages from all nodes are passed in parallel. When using negative log probabilities, all messages $m_{ij}^0(x_j)$ are initialized to zero. And at each iteration, new messages are updated in the following way:

$$m_{j\to i}^t(x_i) = \min_{x_j} \left[S(x_i, x_j) + D_j(x_j) + \sum_{s\in\mathcal{N}(j)\backslash i} m_{s\to j}^{t-1}(x_j) \right], \tag{2.9}$$

where $\mathcal{N}(j)\backslash i$ denotes the neighbors of node j excluding node i. After T iterations, a belief vector $b_i(x_i)$ is computed as:

$$b_i(x_i) = D_i(x_i) + \sum_{j\in\mathcal{N}(i)} m_{j\to i}^T(x_i). \tag{2.10}$$

In the BP algorithm, the belief $b_i(x_i)$ at a node i is proportional to the product of local evidence at that node. Finally, the label $\widehat{x}_i$ minimizing $b_i(x_i)$ is selected as the optimal label for node v_i. For a graph with N nodes and K label candidates, the running time for T iterations is $O(TNK^2)$. It takes $O(K^2)$ time to compute message for each node, and there are $O(N)$ nodes.

2.2.2 Sum-Product BP on MRFs

Rather than maximizing the posterior probability, another common formulation is to maximize the *marginal* probabilities, which selects the most probable label for each pixel instead of choosing an overall configuration. This scheme minimizes the number of pixels with incorrect labels, but the overall labeling might have small joint posterior probabilities.

The message update process is based on summation rather than minimum,

as in the following way:

$$m_{ij}(x_j) = \sum_{x_i} \psi_i(x_i)\phi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} m_{ki}(x_i) \tag{2.11}$$

The belief $b_i(x_i)$ is an approximation of the marginal probability at node i. It is computed in the same way as in the max-product algorithm:

$$b_i(x_i) = \psi_i(x_i) \prod_{j \in N(i)} m_{ji}(x_i) \tag{2.12}$$

Notice that we cannot transform this formulation into a negative log probability form. Thus it is less numerical stable, and less popular in computer vision problems, as we must transform the directly defined energy function into probabilities.

Chapter 3

Our Formulation

Structure and texture information are the two critical ingredients of image completion. They are both required to produce a high-quality output. In the literature, some of the works are either pure texture or pure structure. Others deal with structures and textures using different schemes. Considering these aspects, and motivated by the *exemplar-based* approaches, which is the most efficient group of inpainting algorithms, we propose a novel image completion algorithm, combining both structure and texture information under a unified framework.

The framework solve image completion in a global scheme. Similar to [30] and [34], we formulate the image completion problem as an exemplar-based graph labeling problem modeled by discrete MRFs. However, our energy function is different from those in [30] and [34]. The goal of using this global scheme is to put structure and texture under one unified framework.

Belief propagation (BP) is utilized to solve the minimization problem. One challenge in using standard BP is that its computational complexity is the square of the number of label candidates. For a typical 481×321 image, the number of label patches is the order of 10^4. To handle the high computational complexity to make BP practical, we present a two-step coarse-to-fine scheme, where two BPs run with much smaller numbers of label candidates instead of one BP running with a large number of label candidates.

3.1 An MRF Model

Let Π be the region of an input image I and Ω be the region marked to be filled, called the *target* region. We need to fill in Ω using the information from the *source* region $\Phi = \Pi - \Omega$.

We first sparsely sample the input image with a horizontal spacing sh and a vertical spacing sv. Let $P = \{p_1, p_2, ..., p_N\}$ be N sampled pixels in the target region. The process of image completion is to fill the target region by pasting some $w \times h$ patches taken from the source region to the locations centered at $p_i \in P$, $1 \leq i \leq N$. It is a process of propagating textures and structures from the source region to the target region. Denote v_i as the $w \times h$ region centered at the point p_i. Construct an undirected weighted graph $G = (V, E)$, where the node set $V = \{v_1, v_2, ..., v_N\}$ contains all the $w \times h$ rectangle regions centered at some points in P, and E is the set of edges connecting each node to its four neighbors. Fig. 3.1 demonstrates the process of graph construction.

Let $\mathcal{L} = \{l_1, l_2, ..., l_K\}$ be the set of label candidates containing all the $w \times h$ patches taken from the source region. Then the labeling problem is to find the best label configuration $X = \{x_1, x_2, ..., x_N\}$ to minimize an energy function defined later in this section, where $x_i \in \mathcal{L}$, $1 \leq i \leq N$ and $x_i = l_k$ means that the label (patch) for node v_i is l_k.

Before giving the energy function in our algorithm, let us consider the *confidence term*. In [12], a confidence term is proposed to determine the filling priority. They propagate the confidence of the synthesized pixel values in a manner similar to the propagation of information in inpainting. Our approach defines another confidence term, which is based on the distance from an unknown pixel to the source region. It reflects the importance of the nodes in our propagation process. Since image completion is to propagate textures and structures in the source region to the target region, the pixels in the target region closer to the source region should be more important in the propagation

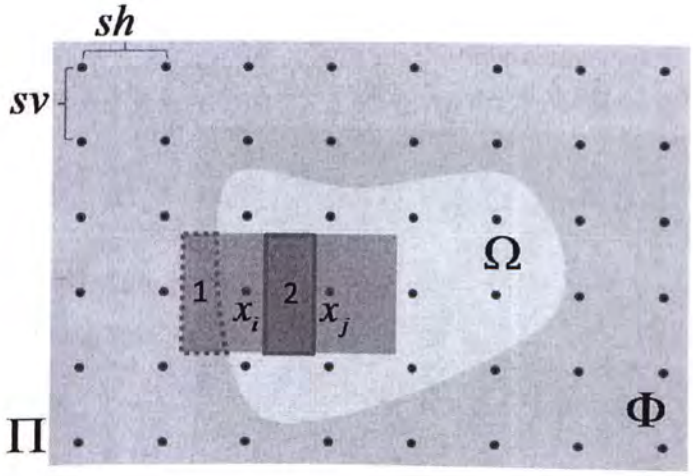

Figure 3.1: Graph construction. The image is sparsely sampled with a horizontal spacing sh and a vertical spacing sv. The samples in the target region Ω construct the nodes of the graph G. The overlapping part (region 1) between x_i and the source region provides the data cost for x_i and the overlapping part (region 2) between x_i and x_j gives the consistency cost for (x_i, x_j).

process, thus have larger confidence values. Fig. 3.2 shows an example of the confidence value distribution.

Based on the MRF model, our energy function is defined in the following form

$$E(X) = \sum_{v_i \in V} E_1(x_i) + \sum_{(v_i, v_j) \in E} E_2(x_i, x_j), \tag{3.1}$$

where $E_1(x_i)$ is the data cost for label x_i and $E_2(x_i, x_j)$ is the consistency cost for (x_i, x_j) (see Fig. 3.1). Actually, this energy function is where we combine both structure information and texture information together. By minimizing the energy, the structure and texture are both considered and also collaborate with each other to help the propagation process. MRFs and the energy defined here constitute a unified framework. Detailed definition of E_1 and E_2 are discussed bellow. In [30] and [34], the energy functions are defined in a similar form to (3.1). However, their definitions of these E_1 and E_2 are different from ours, which result in a different scheme to handle structure information. The

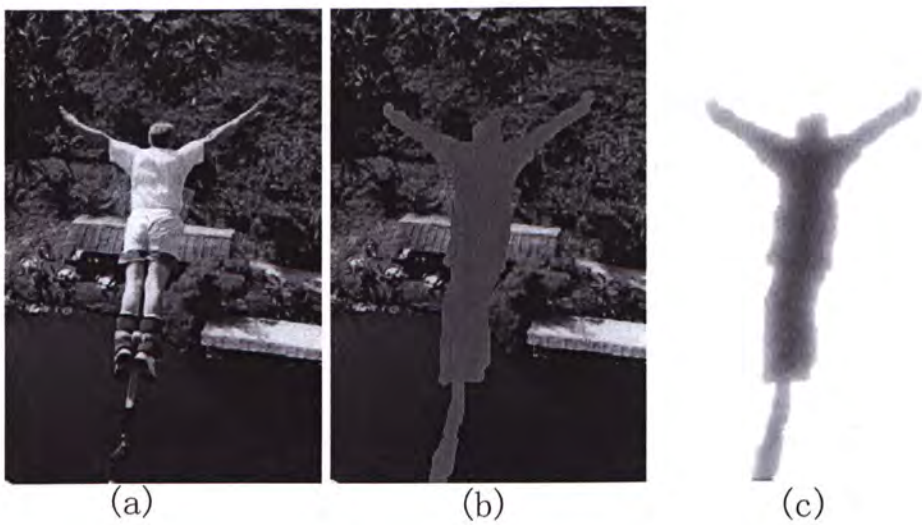

Figure 3.2: An example of the map of the confidence values. (a) The input "bungee" image. (b) The mask (in green) of the boy to be removed. (c) The confidence map in the mask, in which the brighter a pixel is, the larger the confidence value.

differences will be explained in the following contents.

The data cost term for label x_i is defined as:

$$E_1(x_i) = C(i) \cdot d(x_i, \Phi), \tag{3.2}$$

where $C(i)$ is the confidence value for node v_i and $d(x_i, \Phi)$ constrains the synthesized patch x_i to match well with the source region Φ when x_i and Φ overlap. $d(x_i, \Phi)$ is calculated as the sum of the squared differences (SSD) of pixel colors in the overlapping part between x_i and Φ (e.g., region 1 surrounded by the red dashed curve in Fig. 3.1). When x_i and Φ do not overlap, $E_1(x_i) = 0$.

In [34], the cost term is linearly composed of the structure and texture components, which are used to propagate structure and texture information, respectively. The structure information is some curves drawn interactively by the user. The texture component is defined the same as $d(x_i, \Phi)$ in (3.2) but without $C(i)$. The cost term used in [30] is also the same as $d(x_i, \Phi)$ without $C(i)$.

The consistency cost term $E_2(x_i, x_j)$ in our algorithm is defined as

$$E_2(x_i, x_j) = \left[\frac{C(i) + C(j)}{2}\right] \left[\lambda_1 E_2^t(x_i, x_j) + \lambda_2 E_2^s(x_i, x_j)\right], \tag{3.3}$$

where $C(i)$ and $C(j)$ are the confidence values of nodes v_i and v_j, respectively, $E_2^t(x_i, x_j)$ is a term used to enforce consistency for texture propagation, and $E_2^s(x_i, x_j)$ for structure propagation. λ_1 and λ_2 are two factors to balance E_1, E_2^t, and E_2^s. Here $E_2^t(x_i, x_j)$ is computed by

$$E_2^t(x_i, x_j) = d(x_i, x_j), \tag{3.4}$$

where $d(x_i, x_j)$ is the SSD in the overlapping part between x_i and x_j (e.g., region 2 surrounded by the red solid curve in Fig. 3.1). $E_2^s(x_i, x_j)$ is computed by

$$E_2^s(x_i, x_j) = d_{gx}^2(x_i, x_j) + d_{gy}^2(x_i, x_j), \tag{3.5}$$

where $d_{gx}(x_i, x_j)$ and $d_{gy}(x_i, x_j)$ are the gradient differences between x_i and x_j in x and y directions, respectively. The maximum gradient of the pixels in a patch is used to denote the gradient of the patch, which roughly describes the structure of the patch. The constraint of gradient consistency propagates the structure information.

In [34] and [30], only SSD values are used in the consistency cost term. This is sufficient for the algorithm in [34] since the manually added structure information is incorporated in its cost term. However, for [30], there is no component used for structure propagation.

3.2 Coarse-to-Fine Optimization by BP

Minimizing (3.1) is an NP-hard problem. BP can find a local optimum for such an MRF energy function, in polynomial time. As stated in Chapter 2, the max-product and sum-product are two typical BP algorithms on MRFs.

In our work, the max-product algorithm is used since it is less sensitive to numerical inaccuracy. Conventionally, the max-product algorithm is defined in terms of probability distributions. By taking the negative logarithm of the probabilities, the max-product problem becomes an equivalent min-sum problem, which handles messages directly based on the energy function.

The max-product BP works iteratively by passing messages along the graph. At each iteration t, the message passing from v_j to v_i for label x_i is computed in the following way

$$m_{j\to i}^{t}(x_i) = \min_{x_j} \left[E_2(x_i, x_j) + E_1(x_j) + \sum_{s\in\mathcal{N}(j)\backslash i} m_{s\to j}^{t-1}(x_j) \right], \tag{3.6}$$

where $\mathcal{N}(j)\backslash i$ denotes the neighbors of v_j excluding v_i. Intuitively, $m_{j\to i}^{t}(x_i)$ indicates how likely node v_j believes that node v_i has label x_i. After T iterations, a belief value for each node v_i with label x_i is computed by

$$b_i(x_i) = E_1(x_i) + \sum_{j\in\mathcal{N}(i)} m_{j\to i}^{T}(x_i). \tag{3.7}$$

Finally, the label $\widehat{x}_i$ minimizing $b_i(x_i)$ is selected as the optimal label for node v_i. For a graph with N nodes and K label candidates, the running time for T iterations is $O(TNK^2)$.

In our image completion approach, the main problem with such a standard BP algorithm is that the number of label candidates K is too large to be used in practice. For instance, there are normally more than 10000 patches for a typical 256×256 image, taking a BP hours to run. Moreover, the huge number of labels also implies that for any pair of adjacent nodes v_i and v_j, their matrix of pairwise consistency cost $E_2(x_i, x_j)$ is so large that it cannot fit into memory and therefore cannot be pre-computed. Thus, this matrix must be re-calculated each time node v_i sends messages to node v_j, meaning that K^2 times of E_2 calculations are required for each message update. This is intolerable and makes standard BP impractical. In [30], they encountered the same

problem. In order to reduce the intolerable computational cost of BP caused by the large number of labels, they proposed two cooperative optimization methods over standard BP, i.e.,"dynamic label pruning" and "priority-based message scheduling". The basic idea is to discard labels that are unlike to be optimal, and use some greedy-like scheme to speed up BP's convergence. These methods highly reduced the computational cost. However, as they drop out a large number of labels for each node, and update messages in a greedy way rather than the standard way, some important information might be lost in the process, and the result might be quite different from the result obtained from a standard BP.

3.2.1 Coarse-Level Belief Propagation

In our work, to handle the high computational complexity to make BP practical, we propose a coarse-to-fine scheme. We maintain the diversity of label candidates to keep important structure and texture information. The main idea of this scheme is to perform BP twice with K_1 and K_2 label candidates instead of performing BP once with K candidates, where K_1 and K_2 are much smaller than K. The steps of our coarse-to-fine algorithm are described as follows.

First, we use the K-means algorithm to classify all the image patches in $\mathcal{L}$ into K_1 clusters, denoted as $S_1, S_2, ..., S_{K_1}$. Then $\mathcal{L} = \{S_1, S_2, ..., S_{K_1}\}$. Take the K_1 cluster centers as the label candidates $\mathcal{L}^1 = \{c_1, c_2, ..., c_{K_1}\}$ and minimize (3.1) using the standard max-product BP with $\mathcal{L}^1$ to find the best label configuration $X_1 = \{x_1^1, x_2^1, ..., x_N^1\}$, where $x_i^1 \in \mathcal{L}^1$, $1 \leq i \leq N$ and $x_i^1 = c_{k_1}$ means that the best label for node v_i is c_{k_1} after the first BP. As demonstrated in Fig. 3.3(a), this optimization step fills the target region in a coarse level with blurring effect, since the label candidates are centers of each cluster.

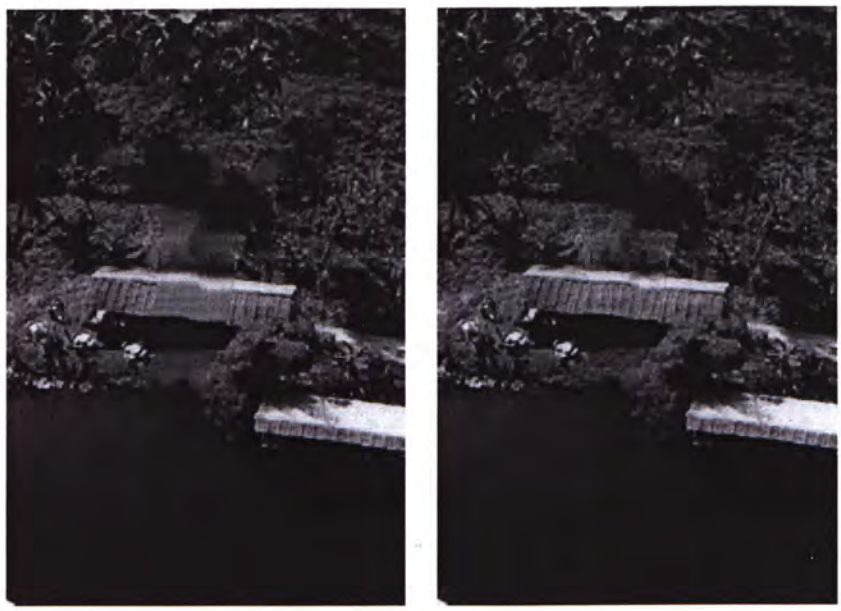

Figure 3.3: An example of our coarse-to-fine BP optimization. (a) The coarse-level result of the first BP step, with blurring effect. (b) The fine-level result of the second BP step, with detailed information filling in.

3.2.2 Fine-Level Belief Propagation

After the coarse result is obtained, the second BP is used to refine the result. Suppose that after the first BP, the best label for node v_i is $x_i^1 = c_{k_1}$. Then the label candidates for node v_i is $\mathcal{L}_i^2 = S_{k_1}$, where S_{k_1} is the k_1th cluster obtained by the K-means algorithm. A constant K_2 is used to limit the maximum label number for $\mathcal{L}_i^2$: if the patch number of S_{k_1} is larger than K_2, then K_2 patches are randomly selected from S_{k_1} as the label candidates in $\mathcal{L}_i^2$. Using such different label candidate sets for different nodes, the second BP runs to find the best label configuration, which gives the final result of our image completion algorithm. Actually in this step, the total number of label candidates is still K. What we do here is using the result from the coarse level as a constraint on each node, i.e., the label candidates for node v_i can only be chosen from the cluster whose center is x_i^1. In Fig. 3.3 an example is shown. Compared

with the coarse-level result, the second BP uses the original patches from the source region, producing better result with detailed texture and structure information.

3.2.3 Performance Enhancement

By using a two-step BP, the most critical benefit is that it makes our algorithm practical. Suppose $K = 10000$, $K_1 = K_2 = 100$, and all the BP algorithms take the same number of iterations to stop. Then our two-step BP algorithm is about 5000 times faster than the standard BP algorithm. Actually, it reduces the computational complexity from $O(TNK^2)$ to $O(TNK)$ by setting K_1 and K_2 appropriately.

Moreover, the scheme significantly reduces the computational cost in another way. In the normal case, where $K > 20000$, the consistency cost $E_2(x_i, x_j)$ can hardly fit into memory, as it takes at least 3GB. Thus, whenever updating the message $m_{j\to i}^t(x_i)$ in (3.6), $E_2(x_i, x_j)$ needs to be recalculated. For standard BP algorithm, the total complexity of recalculating $E_2(x_i, x_j)$ is $O(TNK^2P)$, where P (around 100) is the number of pixels in a patch. It is P times the complexity of the main algorithm, and makes the un-optimized approach terribly unpractical. By adopting the coarse-to-fine scheme, the memory required by $E_2(x_i, x_j)$ is significantly reduced to less than 200MB. So the consistency cost can be calculated beforehand and fetched from memory whenever needed.

According to the above two reasons, our two-step BP scheme is critical to make the algorithm practical. The standard BP would unbearably take weeks for a typical image, while our optimized algorithm only needs 1 minutes.

Obviously, such a coarse-to-fine BP scheme leads to a result different from that obtained with the standard BP. This is the tradeoff for speeding up. As we maintain all the label candidates from the source region, we preserve the

label diversity and keep important structure and texture information as much as possible, which promises good performance. Our experiments show that this scheme achieves satisfactory results.

This approach is similar to the pyramid-based method in computer vision. The coarser level provides candidate constraints for the next level, and the finer level result shows more detailed information. Obviously, we may use multi-level models. The model with more than two levels increases the algorithm's complexity, and might not perform better than the two-level procedure for our inpainting problem. Last but not least, such a coarse-to-fine BP procedure is not only limited to our specific problem. It can also be used to speed up other MRF based applications in computer vision and computer graphics.

Chapter 4

Experiments

4.1 Comparison

In this chapter, we perform a number of experiments and comparisons. To demonstrate the performance of our algorithm, we test it on real images. Most recent related algorithms are published in [12], [34], and [30]. We do further comparison with [34], even though it is based upon manually added structure guidance and not belongs to automatic inpainting algorithms.

Fig. 4.1 gives the comparative results on the "bungee" image in Fig. 3.2(a), where we want to remove the boy. Fig. 4.1(a) is obtained by the algorithm in [12], which propagate texture and structure in a greedy scheme by using some automatic guidance. As the guidance is based on heuristics and ad hoc principles, the algorithm does not apply in general cases, and often produces some obvious artifacts, as shown here in Fig. 4.1(a). Another drawback of [12] is that, once a missing block has been assigned some source patch, the block cannot change its appearance thereafter. This reveals the greediness of the approach, which again leads to visual inconsistencies. Later we will show more comparison results.

Fig. 4.1(b) is the result by our BP algorithm with the confidence term $C(i)$ and the structure term E_2^s removed from the energy function in (3.1). This turns out to be the same energy function as in [30]. As stated before,

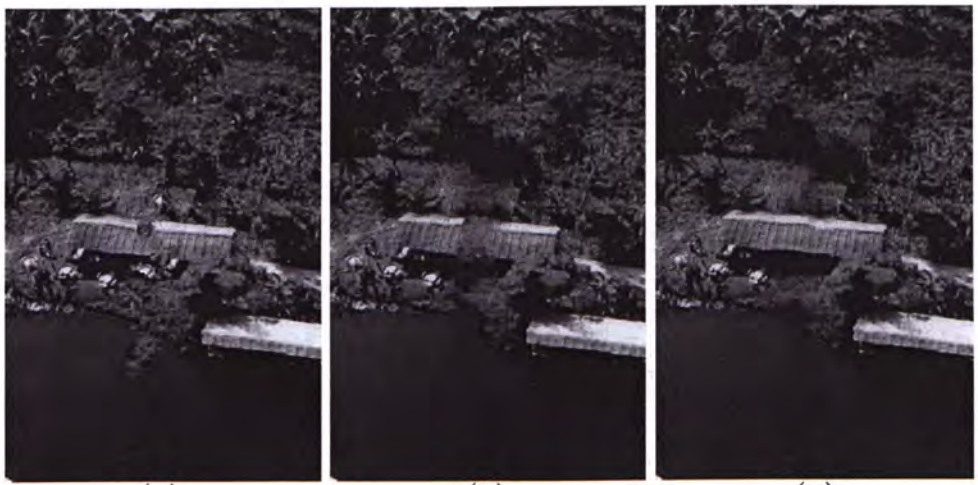

Figure 4.1: The comparative results on the "bungee" image. (a) The result of the greedy approach in [12]. (b) The result of BP without confidence and structure terms (i.e., C and E_2^s), as in [30]. (c) Our result.

the algorithm in [30] use two cooperative optimization methods over standard BP. Their basic idea is to discard labels that are unlike to be optimal, and use some greedy-like scheme to speed up BP's convergence. This approach greatly reduces the computational costs, but might lose important information by discarding most of their labels. From Fig. 4.1(b) we can see that some strong structures (e.g., the roof) cannot be propagated well, due to the loss of structure information and confidence term. Fig. 4.1(c) shows our result. We successfully propagates the structure and texture, and produces a quite satisfying image in which the filling in is non-detectable. The boundaries different regions are smoothly connected, and the detailed texture information is correctly filled in.

More comparative results are shown below in Fig. 4.2. Each column contains a set a experiment. From top to bottom, respectively, each row contains the original images, the masked images, the results obtained by the greedy algorithm in [12], the results of BP without confidence and structure terms (i.e.,

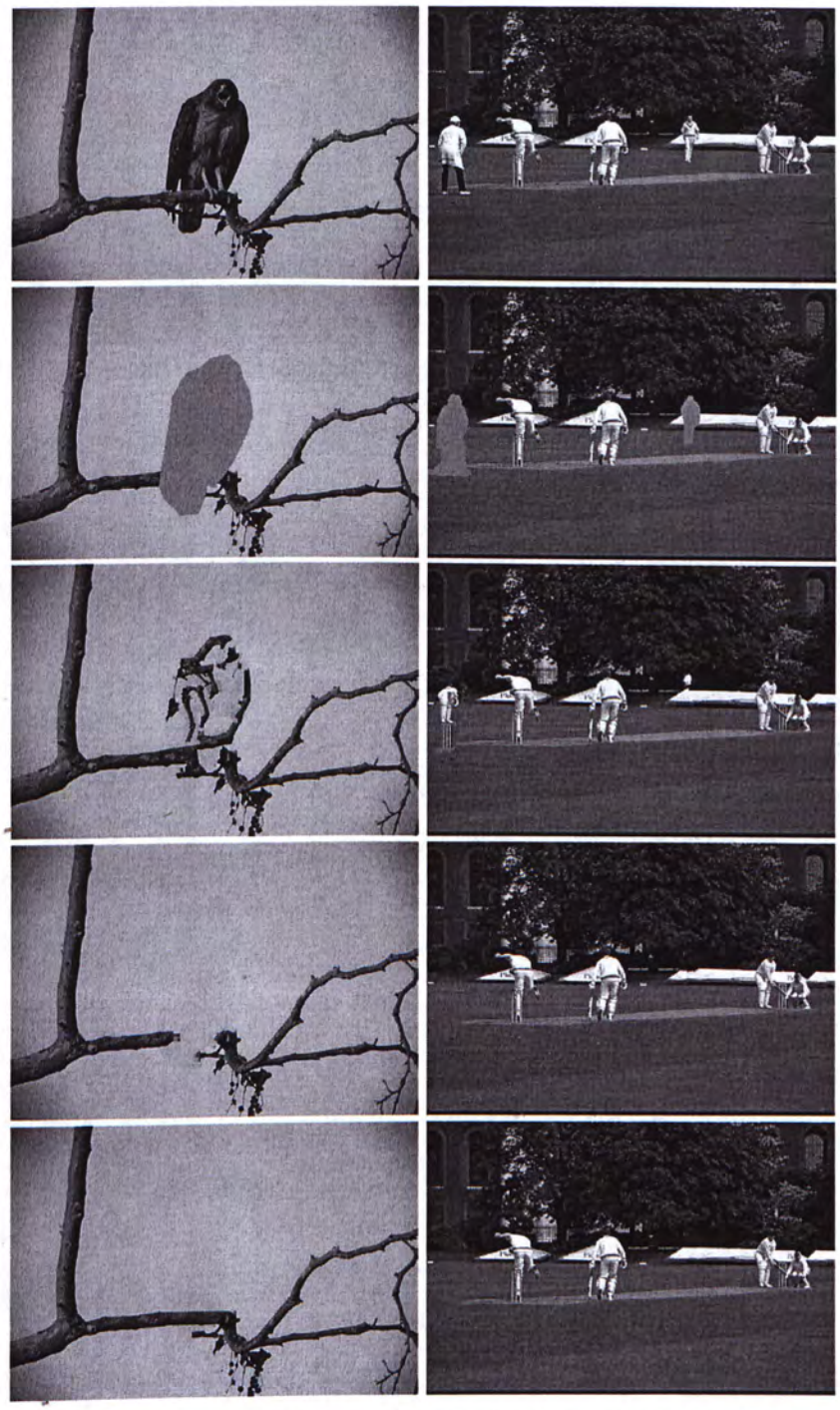

Figure 4.2: More comparative results. Each column contains a set a experiment. From top to bottom, respectively, each row contains the original images, the masked images, the results obtained by the greedy algorithm in [12], the results of BP without confidence and structure terms (i.e., C and E_2^s) as in [30], and our results. (The results can be seen more clearly on the screen with enlarged images.)

C and E_2^s) as in [30], and our results. These comparisons further demonstrate the advantages of our algorithm. The results can be seen more clearly on the screen with enlarged images.

We do further comparison with [34], even though it is based upon manually added structure guidance and not belongs to automatic inpainting algorithms. As Fig. 4.3 shows, by adopting manual guidance, the algorithm in [34] (3rd row) performs almost perfectly. Obviously, our fully automatic algorithm can hardly perform better than Sun's, but it also produces comparable results when the source region provides enough structure and texture information (e.g., the "window" image in Fig. 4.3). When the information is lacking (e.g., the "eagle" image in Fig. 4.3), our result is reasonable and satisfying to some extent.

Fig. 4.5 and Fig. 4.6 shows more results by our algorithm. Our global propagation scheme can produce visually natural results without obvious artifacts. Both structure (e.g., branches and the horizon) and texture (e.g., the pasture and snow) are handled very well in the images with excellent visual consistency. These comparison experiments demonstrate the advantages of our global optimization approach. By combining both structure and texture information under the unified framework, the BP process finds the optimal solution which automatically balances the structure and texture.

In our approach, the patch size ranges between 9×9 and 15×15, the cluster constants K_1 and K_2 range from 100 to 200. For a typical 481 $\times$ 321 image, the K-means algorithm (in Matlab) costs about 1 minute. The two-step BP procedure (in C++) takes less than 20 seconds, while a standard BP would take weeks to tackle one image. This makes our approach practical without losing any information or reducing the output quality.

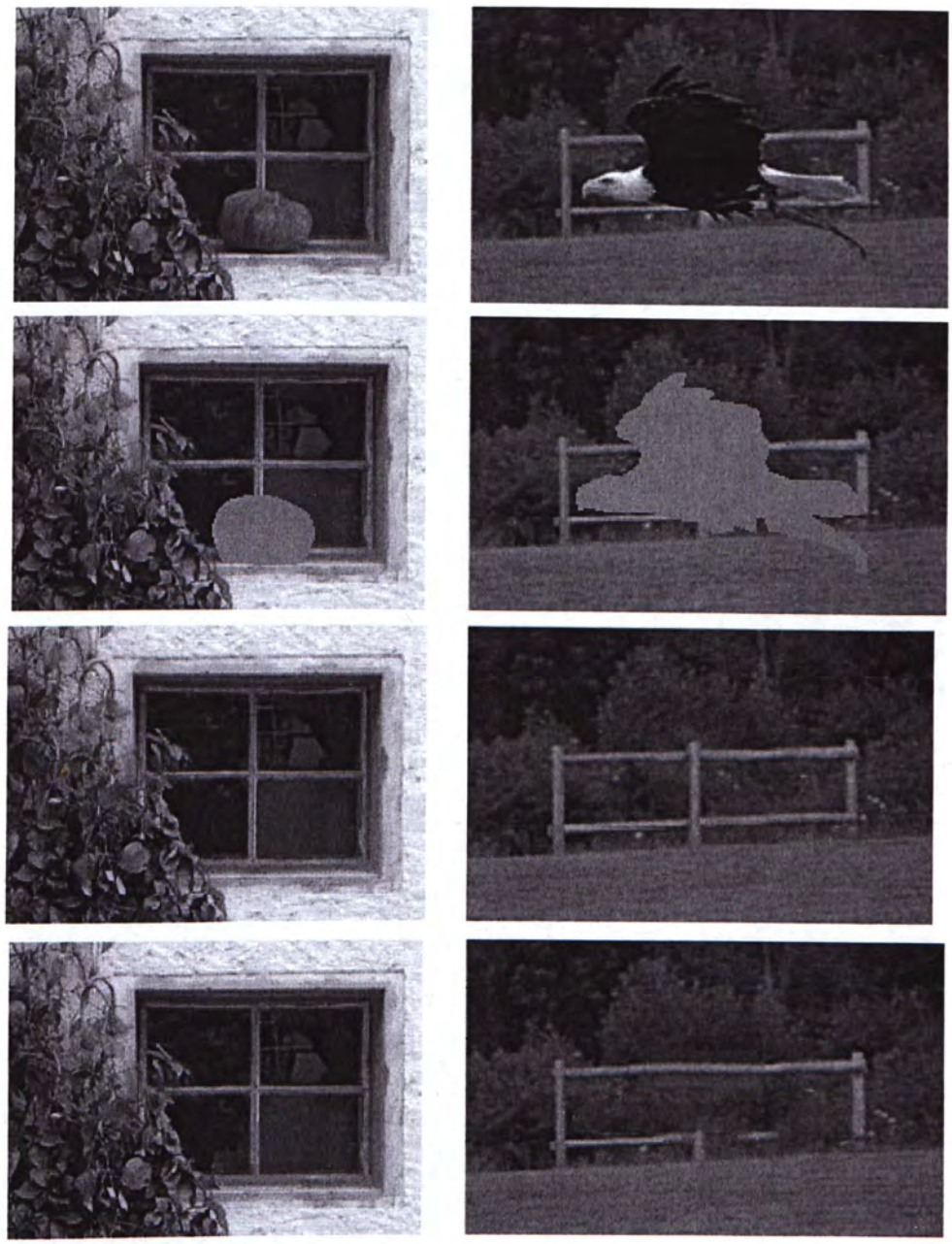

Figure 4.3: The comparison with [34]. Each column contains a set a experiment. From top to bottom, respectively, each row contains the original images, the masked images, the results obtained by [34] with manual guidance, and our results.

4.2 Failure Case

Definitely, there're failure cases for our algorithm, on which we cannot produce satisfying outputs. Fig. 4.4 shows an example. (b) is our result which obviously has some blurring effect. There're two reasons for this: (1) most of the label candidates in the source region are blurred; (2) the SSD energy function in our algorithm "prefer" blurred textures than detailed textures. But we notice that the structure of the leaf is well propagated. We also test this failure case on the other algorithms. The greedy approach shows many artifacts. The result of BP without C and E_2^s cannot fill-in structures, and produce blurring effects too.

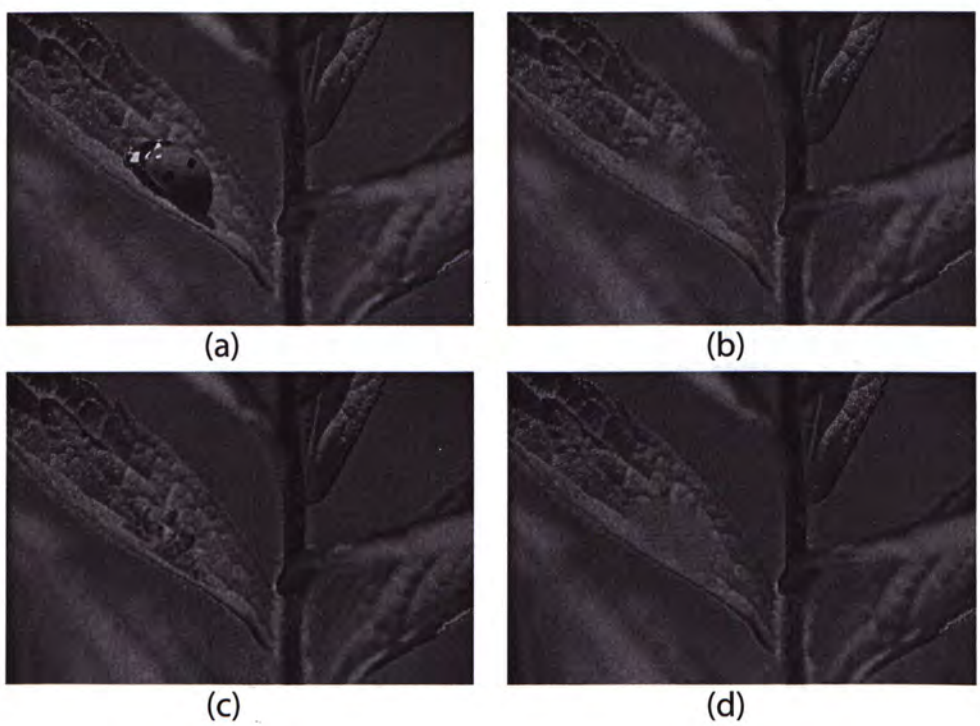

Figure 4.4: A failure case of removing the bug. (a) original image; (b) our result; (c) the result obtained by the greedy algorithm in [12]; (d) the result of BP without confidence and structure terms (i.e., C and E_2^s) as in [30].

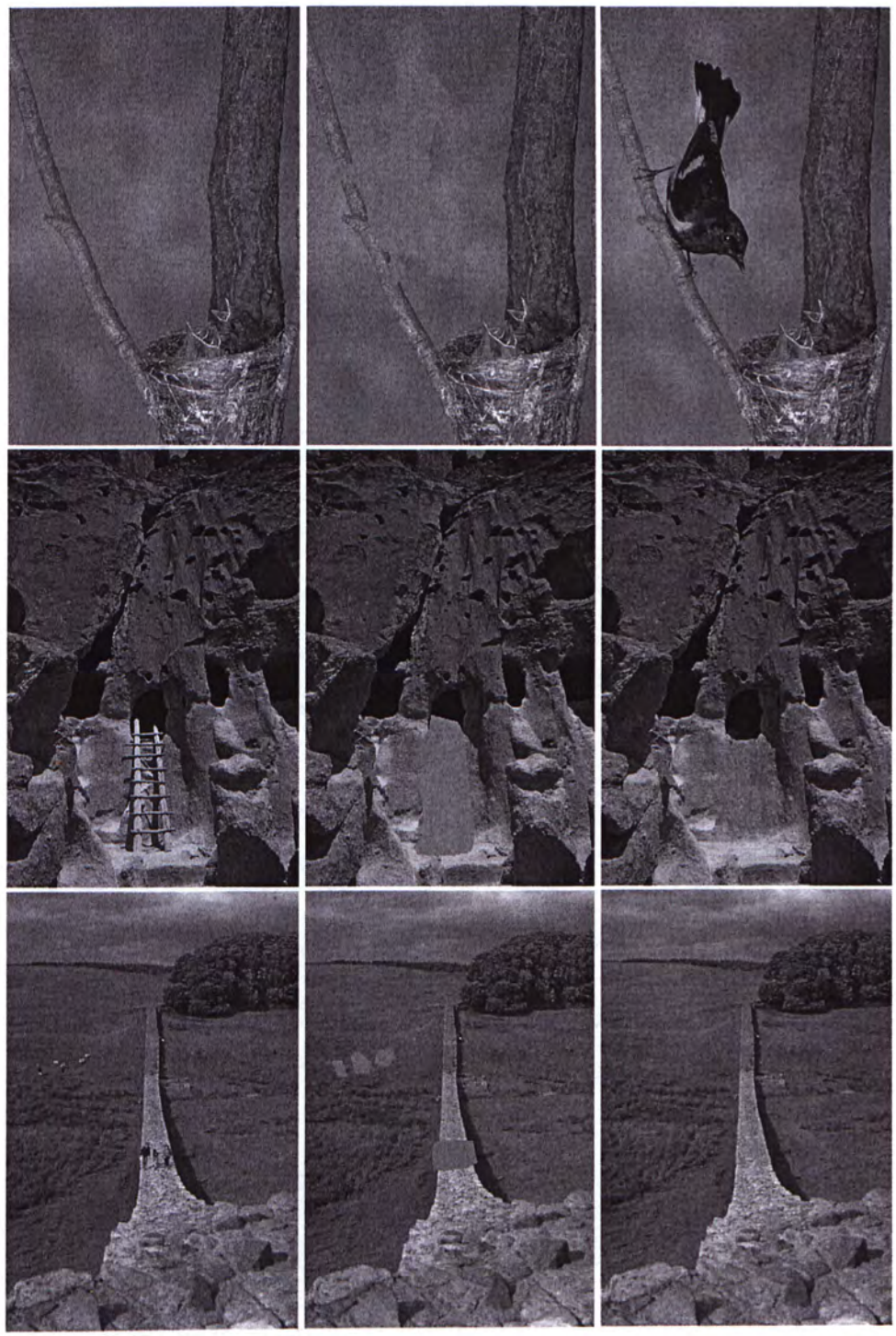

Figure 4.5: More results by our algorithm. Each row contains a set of experiments. From left to right: original images, masked images, and the results.

Figure 4.6: More results by our algorithm. Each row contains a set of experiments. From left to right: original images, masked images, and the results.

Chapter 5

Conclusion

A novel image inpainting approach incorporating texture and structure information is proposed in our work. We formulate the inpainting as a global discrete optimization problem by an MRF model. Our unified framework automatically balances structure and texture information, which are the two critical factors in image inpainting.

Notice that, in our energy function definitions, we use two simple forms to represent structure and texture information. The experiments show that these simple definitions are capable enough to produce satisfactory results. We can surely use some other more complex structure and texture features to enhance the output quality.

The max-product BP algorithm is used to solve the optimization problem. To improve the efficiency, a coarse-to-fine scheme is taken in our algorithm, where the BP runs twice with two much smaller numbers of label candidates. As we maintain all the label candidates from the source region, we preserve the label diversity and keep important structure and texture information as much as possible, which promise good performance. By using a two-step BP, the most critical benefit is that it makes our algorithm practical without losing any information or reducing the output quality. Actually, it reduces the computational cost from $O(TNK^2)$ to $O(TNK)$ by setting K_1 and K_2 appropriately, and solves the recalculating problem by fitting $E_2(x_i, x_j)$ into memory.

Such a coarse-to-fine BP procedure is not only limited to our specific problem. It can also be used to speed up other MRF based applications in computer vision and computer graphics. The experimental results demonstrate that our algorithm is efficient and outperforms two most recent algorithms.

Though we have made encouraging progress in image inpainting, it is far from the end of the road. A lot of work can be done to make further improvement. Video completion is one of our further works.

Bibliography

[1] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro, and J. Verdera. Filling-in by joint interpolation of vector fields and gray levels. *Image Processing, IEEE Transactions on*, 10(8):1200–1211, 2001.

[2] W. Barrett and A. Cheney. Object-based image editing. In *SIGGRAPH*, 2002.

[3] M. Bertalmio, A. Bertozzi, and G. Sapiro. Navier-Stokes, Fluid Dynamics, and Image and Video Inpainting. *Proc. of IEEE-CVPR*, pages 355–362, 2001.

[4] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. *Image inpainting.* ACM Press/Addison-Wesley Publishing Co. New York, NY, USA, 2000.

[5] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher. Simultaneous structure and texture image inpainting. *Image Processing, IEEE Transactions on*, 12(8):882–889, 2003.

[6] J. Besag. On the Statistical Analysis of Dirty Pictures. *Journal of the Royal Statistical Society. Series B (Methodological)*, 48(3):259–302, 1986.

[7] R. Bornard, E. Lecan, L. Laborelli, and J. Chenot. Missing data correction in still images and image sequences. In *ACM MM*, 2002.

[8] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.

[9] Y. Boykov, O. Veksler, and R. Zabih. Markov random fields with efficient approximations. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 648–655, 1998.

[10] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1222–1239, 2001.

[11] T. Chan and J. Shen. Nontexture Inpainting by Curvature-Driven Diffusions. *Journal of Visual Communication and Image Representation*, 12(4):436–449, 2001.

[12] A. Criminisi, P. Perez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *Image Processing, IEEE Transactions on*, 13(9):1200–1212, 2004.

[13] I. Drori, D. Cohen-Or, and H. Yeshurun. Fragment-based image completion. *ACM Transactions on Graphics (TOG)*, 22(3):303–312, 2003.

[14] A. Efros and W. Freeman. Image quilting for texture synthesis and transfer. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 341–346, 2001.

[15] A. Efros and T. Leung. Texture synthesis by non-parametric sampling. *International Conference on Computer Vision*, 2(9):1033–1038, 1999.

[16] G. Emile-Mâle. *The Restorer's Handbook of Easel Painting.* Van Nostrand Reinhold, 1976.

[17] W. Freeman, E. Pasztor, and O. Carmichael. Learning Low-Level Vision. *International Journal of Computer Vision*, 40(1):25–47, 2000.

[18] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms*, pages 564–584, 1987.

[19] G. Guy and G. Medioni. Inferring global pereeptual contours from local features. *International Journal of Computer Vision*, 20(1):113–133, 1996.

[20] P. Harrison. A non-hierarchical procedure for re-synthesis of complex textures. In *WCSG*, 2001.

[21] D. Heeger and J. Bergen. Pyramid-based texture analysis/synthesis. *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 229–238, 1995.

[22] A. Hertzmann, C. Jacobs, N. Oliver, B. Curless, and D. Salesin. Image analogies. *Proceedings of SIGGRAPH 2001*, pages 327–340, 2001.

[23] T. Huang, S. Chen, J. Liu, and X. Tang. Image inpainting by global structure and texture propagation. *Proceedings of the 15th international conference on Multimedia*, pages 517–520, 2007.

[24] J. Jia and C. Tang. Image repairing: robust image synthesis by adaptive NDtensor voting. *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, 1, 2003.

[25] L. Joyeux, O. Buisson, B. Besserer, and S. Boukir. Detection and removal of line scratches in motion picture films. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1:548–553, 1999.

[26] D. King. *The commissar vanishes.* Metropolitan Books, 1997.

[27] A. Kokaram, R. Morris, W. Fitzgerald, and P. Rayner. Detection of missing data in image sequences. *Image Processing, IEEE Transactions on*, 4(11):1496–1508, 1995.

[28] A. Kokaram, R. Morris, W. Fitzgerald, and P. Rayner. Interpolation of missing data in image sequences. *Image Processing, IEEE Transactions on*, 4(11):1509–1519, 1995.

[29] V. Kolmogorov and R. Zabin. What energy functions can be minimized via graph cuts? *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(2):147–159, 2004.

[30] N. Komodakis and G. Tziritas. Image completion using global optimization. *CVPR06, pp. I*, pages 442–452, 2006.

[31] S. Masnou and J. Morel. Level lines based disocclusion. *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on*, pages 259–263, 1998.

[32] K. Murphy, Y. Weiss, and M. Jordan. Loopy belief propagation for approximate inference: An empirical study. *Proceedings of Uncertainty in AI*, pages 467–475, 1999.

[33] M. Nitzberg, D. Mumford, and T. Shiota. *Filtering, Segmentation, and Depth.* Springer, 1993.

[34] J. Sun, L. Yuan, J. Jia, and H. Shum. Image completion with structure propagation. *International Conference on Computer Graphics and Interactive Techniques*, pages 861–868, 2005.

[35] S. Walden. *The Ravished Image; Or, how to Ruin Masterpieces by Restoration.* St. Martin's Press, 1985.

[36] L. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 479–488, 2000.

[37] L. Williams. Stochastic Completion Fields: A Neural Model of Illusory Contour Shape and Salience, 1997.

[38] J. Yedidia, W. Freeman, and Y. Weiss. Generalized belief propagation. *Advances in Neural Information Processing Systems*, 13:689–695, 2001.

[39] J. Yedidia, W. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. *Exploring Artificial Intelligence in the New Millennium*, pages 239–236, 2003.