1-31-2013

A robust patch-based synthesis framework for combining inconsistent images

Aliakbar Darabi

Aliakbar Darabi
Candidate

Electrical and Computer Engineering
Department

This dissertation is approved, and it is acceptable in quality and form for publication:

Approved by the Dissertation Committee:

Pradeep Sen , Chairperson

Nasir Ghani

Yasamin Mostofi

Joe Micheal Kniss

A robust patch-based synthesis framework for combining inconsistent images

by

Aliakbar Darabi

B.S., Sharif University of Technology, 2005
M.S., Sharif University of Technology, 2007

DISSERTATION

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Doctor of Philosophy
Engineering

The University of New Mexico

Albuquerque, New Mexico

December 2012

Dedication

To my dear parents for their love

Acknowledgments

First of all, I would like to thank my advisor, Dr. Pradeep Sen, for supporting me and being a good role model. It has been an honor for me to be your first Ph.D. student. From you, I have learned a great deal about research, career, and life.

Special thanks go to Eli Shechtman and Dan Goldman for providing research mentorship during my two internships at Adobe Systems. Thanks to the other AGL graphics lab members, who provided useful research support, including Lei Xiao, Maziar Yaesoubi, Vahid Noormofidi, Nima Khademi Kalantari, and Hao He.

I thank the following funding sources for sponsoring this work: National Science Foundation under grant IIS-0845396, Adobe Systems Inc. Finally, I thank everyone who supported me during my graduate school. Particularly, my parents and my brother have provided me support and encouragement.

A robust patch-based synthesis framework for combining inconsistent images

by

Aliakbar Darabi

B.S., Sharif University of Technology, 2005
M.S., Sharif University of Technology, 2007

Ph.D., University of New Mexico, 2012

Abstract

Current methods for combining different images produce visible artifacts when the sources have very different textures and structures, come from far view points, or capture dynamic scenes with motions. In this thesis, we propose a patch-based synthesis algorithm to plausibly combine different images that have color, texture, structural, and geometric inconsistencies. For some applications such as cloning and stitching where a gradual blend is required, we present a new method for synthesizing a transition region between two source images, such that inconsistent properties change gradually from one source to the other. We call this process *image melding*. For gradual blending, we generalized patch-based optimization foundation with three key generalizations: First, we enrich the patch search space with additional geometric and photometric transformations. Second, we integrate image gradients into the patch representation and replace the usual color averaging with a screened Poisson equation solver. Third, we propose a new energy based on mixed L_2/L_0 norms for colors and gradients that produces a gradual transition between sources without sacrificing texture sharpness. Together, all three generalizations enable patch-based solutions to a broad class of image melding problems involving inconsistent sources: object

cloning, stitching challenging panoramas, hole filling from multiple photos, and image harmonization.

We also demonstrate another application which requires us to address inconsistencies across the images: high dynamic range (HDR) reconstruction using sequential exposures. In this application, the results will suffer from objectionable artifacts for dynamic scenes if the inconsistencies caused by significant scene motions are not handled properly. In this thesis, we propose a new approach to HDR reconstruction that uses information in all exposures while being more robust to motion than previous techniques. Our algorithm is based on a novel patch-based energy-minimization formulation that integrates alignment and reconstruction in a joint optimization through an equation we call the HDR image synthesis equation. This allows us to produce an HDR result that is aligned to one of the exposures yet contains information from all of them.

These two applications (image melding and high dynamic range reconstruction) show that patch based methods like the one proposed in this dissertation can address inconsistent images and could open the door to many new image editing applications in the future.

Contents

List of Figures

List of Figures

Chapter 1

Introduction

1.1 Motivation

Recently due to popularity of digital photography, much research has been dedicated to image editing. Consequently, the variety of applications for editing images has grown considerably. We can divide these applications into two main categories: first, the ones operating on a single image as their input to generate a new one based on a task; and second, the ones that take extra images and combine their inputs to generate new ones. See Figure 1.1 for some image editing examples. Image completion is an instance that fits into the first category where the algorithm has to remove part of an input image and synthesize new content to fill the missing region. For the second category, there is image cloning where the goal is to transfer part of content of an image into another one, seamlessly. In the rest of this thesis, many more applications will be shown for both categories.

Although several popular tools exist for many existing image editing tasks that often work well in their target applications, they each have limitations in a general image manipulation framework. These tools cannot be applied to problems other than those for which they have been designed and they are usually fundamentally limited to consistent sources

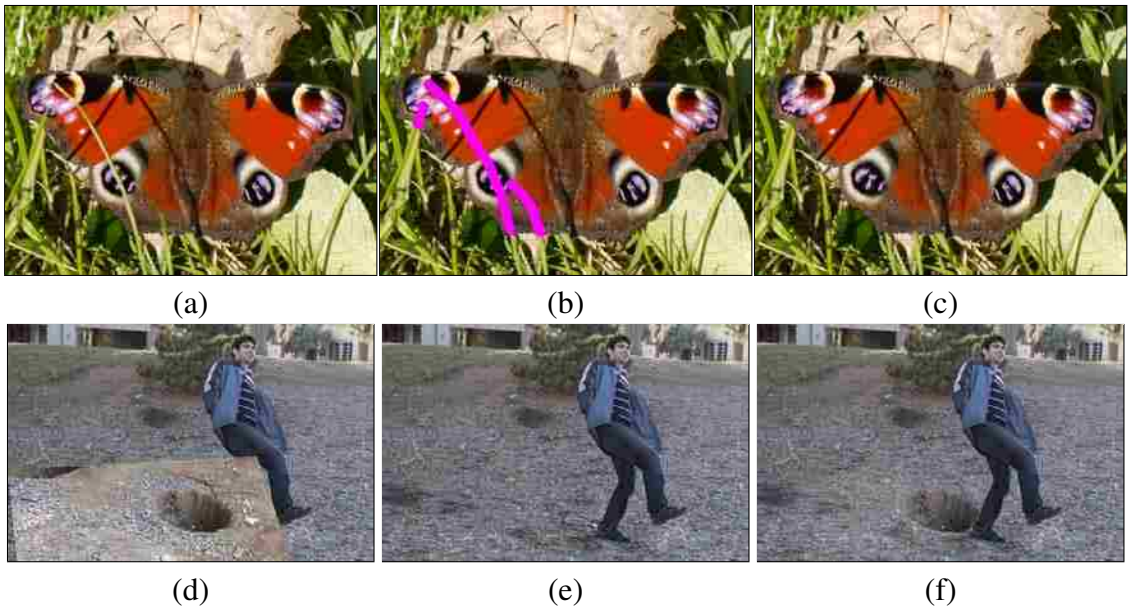


Figure 1.1: *Image editing examples. (a-c) Image completion as an example for single source image editing (a) an original image (the butterfly is covered by the grass)(b) mask image (the magenta shows the "hole" regions (c) hole filled image (the algorithm used the rest of the image to reconstruct the missing region.) (d-f) Object cloning, instance of multi-image editing category (d) source image, (e) target image, (f) the hole from source is seamlessly cloned into the target image and colors and structures are adjusted.*

where the source have geometric and photometric similarities. In this thesis, we are trying to solve the inconsistency problem that may exist between the input sources. By "inconsistent", we mean that the image contents can have different orientations, scales, exposure, color palettes, or textures, making the matching and combination processes difficult. This inconsistency can happen even in a single image where the content inside that cannot be used without any extra steps. By applying the new introduced framework on several different applications, we will demonstrate how we can improve the quality of state-of-the-art methods specifically designed for those problems. In this thesis, we will also demonstrate several types of image combinations. In applications such as stitching the goal is to spatially blending different sources together with a seamless and gradual spatial transition from one source to the other(s). In applications such as morphing, a temporal blend hap-

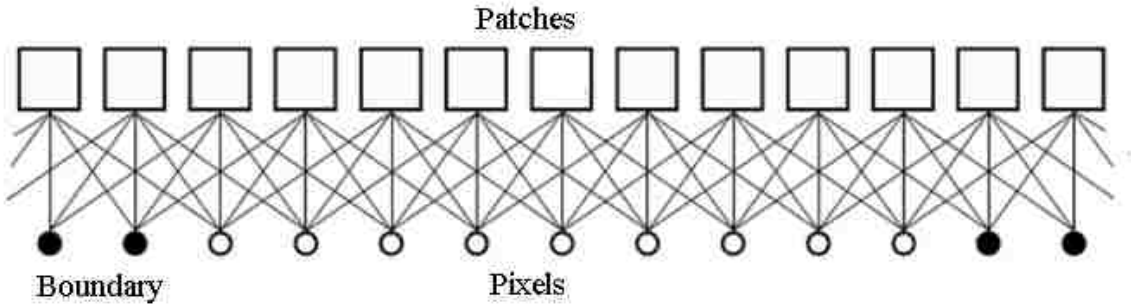


Figure 1.2: *Chain of patches. illustrating how constraints flow through all patches. The circles represent pixels and the shaded ones show the ones that have constraints like the ones located at boundaries. The patches overlapping each other share some pixels and for the good match all the pixels have to well-present the patches consisting that pixel so the patches are now connected with these pixels and through this, data propagates through the chain of pathes (taken from Wexler et al. [1]).*

pens between sources in a way that when the frames for synthesis get closer to each of the sources, they have to look more similar to them and therefore the combination appears in time. Also, for High Dynamic Range (HDR) reconstruction, we will show how we can look at the problem as a mixture problem in the irradiance domain. We will present many of these combinations as an unified energy optimization framework and generate state-of-the-art quality result by optimizing that target function. See Figure 1.3 for some examples of different types of blending.

1.2 What is patch-based image synthesis?

This thesis is based on the patch-based family of algorithms which means that instead of looking at individual pixels we examine $w \times w$ patches where w is the width of the patch. Unlike the blocks of pixels used in many image processing applications (e.g., graph cut textures [2]), these patches can and do overlap because every pixel is considered to have a $w \times w$ patch around it. The patch is a block of local pixels and has been proven to be

a successful tool in solving many existing problems as we will describe later. Here, we always approach the problems by first defining an energy function for that problem and then developing an algorithm to optimize that function in order to generate a plausible result. As we will explain later, in all of the applications we solve the problem by going down-hill to a local minimum by reducing the function iteratively. The convergence is guaranteed in this method because we always enforce the algorithm not to increase the energy function. Our energy minimization technique is built upon an existing work [3] but we altered the strategy to adapt it to our new proposed energy function.

In Section 2.5 , we will briefly represent the history of patch-based methods and discuss how they have evolved until they get to their current state. Figure 1.2 which was taken from the seminal work by Wexler et al. [3] shows how the constraints flow in a chain of patches. Intuitively speaking, the good match/synthesis happens when each patch agrees with its neighbors on every pixel it shares with them. Because the neighbors themselves need to be consistent to their own neighbors, the consistency constraint flows over all the patches so in the end if the algorithm can come up with a good solution, it usually produces a plausible result for the problem.

Most of the patch-based algorithms have two main stages: 1) nearest neighbor search, and 2) synthesis stage. In the first stage, the algorithm looks for the best suited patches for the target regions and in the second stage it uses the found patches and combine them to produce content for the region. In this thesis, we are mostly interested in the second part and we rely on existing fast search algorithm that introduced in [4]. In the Ph.D. thesis by Connelly Barnes [5], he introduced new fast randomized algorithm to search nearest neighbor patch(es) but in that work the concentration was around the first stage (the search) and the synthesis part was kept the same as before. Instead, in this thesis we are mostly looking into the synthesis part as we generalize the current techniques to broaden the applicability of this family of algorithms. We will show how we can reduce many existing problems in computer graphics and computer vision areas into a simple

energy minimization optimization framework and produce similar and in many cases better results comparing to the state-of-the-art existing method that was developed to solve that specific problem.

1.3 Overview

Here, we give a brief overview of the remainder of this thesis.

1.3.1 Previous work

In Chapter 2, we will review the most common existing editing tools for image editing and blending. Image pyramids, gradient based techniques, graph cuts, texture synthesis, and patch-based methods are involved in most of the advanced image editing tasks. In this chapter, we do a short literature review for those techniques and will talk about the strengths and shortcomings of each of them.

1.3.2 Image Melding

In Chapter 3, we will explain our new approach for novel way to combine images using image synthesis concepts. There we will introduce the new concept of "image melding" which is about a new texture interpolation technique that can be applied on natural images without any assumption on homogeneity of underlying texture. Our synthesis is based on generalizing existing methods by changing their core energy function. There are three main differences between the proposed algorithm and its ancestors. First, we allow the algorithm to apply many geometrical or photometrical operations in addition to simple translation to let the algorithm cope with the large appearance and textural differences in our examples. Second, we will show how by adding gradient channels into our features,

we could incorporate advantages of using Poisson blending inside patch-based techniques. Finally, we will show how we can enforce gradual textural transfer by adding a term into our optimization function.

1.3.3 Patch-based High Dynamic Range image reconstruction

In Chapter 4, we will talk about an alternative way for blending images together. We will introduce a novel way of looking at HDR image reconstruction. In the proposed approach, the combination happens in irradiance domain and we will show how we can reduce HDR reconstruction to be a patch-based image summary problem. In this way, we could produce high quality HDR images despite the existence of motion of non-rigid objects. There, we will compare our technique with many existing algorithms in the area and show superior result for many challenging examples.

1.3.4 Conclusion and future work

We conclude by discussing future work that could be done using our core ideas in synthesis, and potential future applications. Also, we will discus some of the problems of our technique and suggest some of the solutions that can improve our algorithm results. We believe our framework can solve many more existing challenging problems in the field and we will name some of the possible applications in this chapter.

1.4 Contributions of this thesis

The contribution of this dissertation is the first demonstration that patch-based optimization algorithms can be used to address the synthesis problem when images have inconsistencies. To do this, we extend traditional patch-based optimization by making it more

flexible to allow for rotation, scale, exposure differences, and other inconsistencies. We demonstrate two general set of applications:

1. Proposing a patch-based synthesis algorithm, to plausibly combine different images that have color, texture, structural, and geometric inconsistencies and gradually transforming one to another.

2. To address the problem of HDR image reconstruction from a set of LDR bracketed exposures, we introduce a novel patch-based energy-minimization formulation that integrates alignment and reconstruction in a joint optimization through an equation we call the HDR image synthesis equation.

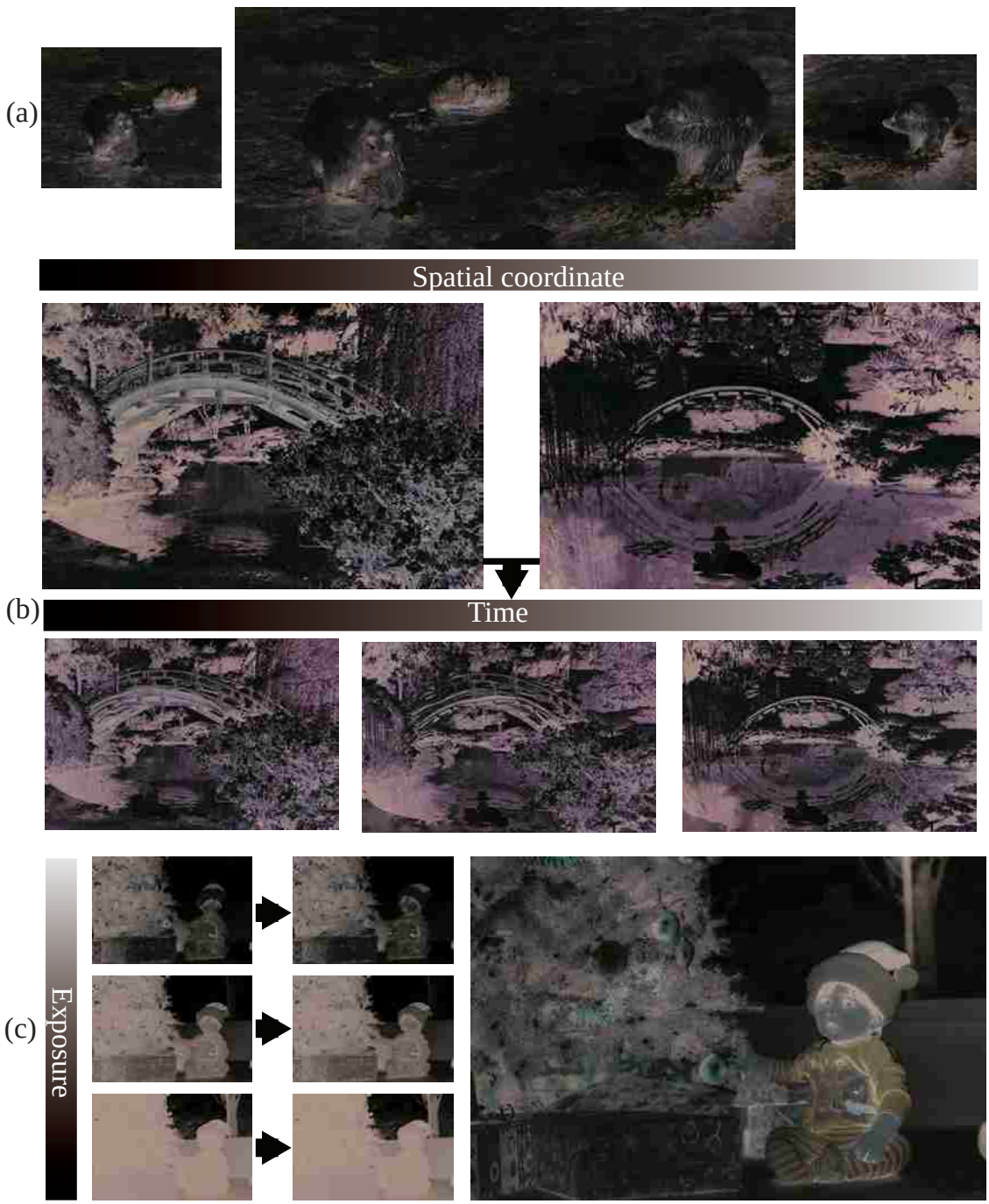


Figure 1.3: *Different examples of mixing images. (a) Stitching as an example of spatial blending; (b) Morphing as an example of temporal blending , and (c) HDR reconstruction as an example of blending images in the irradiance domain.*

Chapter 2

Previous work in image editing

In this chapter we review the previous work for image editing that is related to the subject of this dissertation, such as graph-cut, patch based, gradient, texture synthesis, and image pyramids.

2.1 Image pyramids for blending

The seminal image stitching work by Burt and Adelson [6] introduced the process of combining images by a pyramidal image decomposition, merging its levels and collapsing back to obtain a fused/blended result. One known limitation of the methods in this family is the artifacts around strong edges due to inconsistent treatment of the different levels, but these limitations have recently been addressed [7, 8].

Currently, this family of algorithms have been used mostly for image tonal adjustment as well as detail enhancement/reduction [7]. Also, these methods can be applied as a post processing filter to make the details hidden in irradiance (that has high dynamic range) more stand out when mapping it back to a regular low dynamic range format for display [7]. Recent work by Wu et al. [9], smartly used Laplacian filters both temporally and spatially

to exaggerate temporal differences between video frames and as the result it could reveal many hidden details such as blood flow changes due to heart beat in a regular video clip.

Image Harmonization [8] improved the combination process by smoothing the histogram matching edits across the pyramid levels, and they finally add the noise of one sources to the other one to make the composition more coherent. The technique shows impressive results of transferring the reference coarse structure and blending it nicely with the surrounding colors, as well as rendering similar noise patterns to the target image. However, their ability to render textures is limited to matching statistics of the very fine textural frequencies. Our method shows similar or better results in typical examples but can handle more challenging textures and structures all the way to "pure" texture interpolation.

2.2 Gradient-based image editing

Gradient-domain compositing was introduced to the imaging community by Pérez et al. [10] and has since become the standard for seamless compositing for image stitching [11] and object cloning [12]. As we will show later, this family of algorithms is a strong tool for hiding the color differences when compositing images with different color palettes. Relying on gradients for synthesis constrains the algorithm to distribute the errors uniformly over all parts of the image. Because human are more sensitive to abrupt changes comparing to gradual variations, it is harder for us to see the errors in gradient-based technique.

Pérez et al. [10] showed different ways for blending using gradients. For instance, in the cloning application, the composite gradients is set to be the gradients of the source image for the parts coming from the source. Also, at boundaries the colors have to be colors of target image. In this way, target image and cloned part will have the same color at boundaries and the the correction of the colors we transfer form the other image will be smoothly interpolated over the hole area. This algorithm works well when the source and target have no high frequency details like hard edges. Color bleeding is a well-known

artifact for this technique when the boundary happens at edges because in this case the difference is big and this error affects big region of the image. Some researches has tried to address this issue such as in Farbman et al.'s paper [13] where it allows user interaction and where with some strokes he/she can tell where not to propagate the errors. Tao et al. [14] proposed an adaptive method to hide the error at parts with more details where the users are less sensitive on the error.

Due to the wide usage of this family of algorithms, numerous acceleration techniques have been developed [12, 13, 15]. Agarwala et al. [12] proposed quad tree structures to solve the Poisson linear equation. Farbman et al. [13] suggests that instead of solving the least square problem, simple interpolation can give a similar result. They proposed to use Mean-Value coordinates for error interpolation. They also used an adaptive triangulation to accelerate the the process. Also, parallel computation using GPU was explored by McCann and Pollard [16]. They reached real-time performance and therefore they enabled users to draw with gradients and get real-time feedback about the result. Later, Farbman et al. [15] also reached real-time performance with only using CPU. In their approach, instead of solving the least square problem for the whole image, they break the solver to the recursively filtering an image with a filter that has small footprint. As we will explain later, we approached to our problem in similar way as they did to solve our least square equation.

Beyond blending applications, gradients have a wide range of uses for image editing area. Many de-blurring techniques use a regularization for gradients of the output such as [17]. Usually, these regularizers put a norm lower than two-norm on the gradients to get a sharper result.

If the linear equation for editing contains a function and its gradient at the same time, the equation is called the screened Poisson equation. Bhat et al. [18] showed impressive results when applying different functions on color and gradients separately and then combine them together suing screened Poisson. Similar filtering effects have been shown by

Xu et al. [19] when applying L_0 term on gradients.

In addition to being a powerful synthesis tool, gradients are also commonly used for feature extraction due to their invariance to the lighting conditions [20]. Also, in texture synthesis community, gradients have been commonly selected as feature mostly to find a good warp field [21]. To our knowledge, however, gradients have not been used for synthesis in texture synthesis algorithms. Also, gradients are efficient guidelines for segmentation algorithms in applications that separating different regions of an image is of users interest such the work in [2].

2.3 Graph cuts

In computer vision, graph cuts were first applied by Gerig et al. [22]. Although graph cuts were originally designed for binary labeling problems, Boykov et al. [23] showed that it could also be extended to more general cases. In the general case, the solution is not the global minimum answer and it is an approximation, but it has been proven to be a strong tool for solving computer vision problems.

Graph cuts were introduced to graphics by Kwatra et al. [2] to seamlessly combine textures and stitch images. Kwatra et al. include a search for only a few discrete rotations, scales and a reflection. This search helped the algorithm alleviate repetition artifacts. In contrast, our method includes the continuous-domain transformation search as part of our global optimization formulation.

Agarwala et al. [24] combined gradient domain blending with graph cuts to seamlessly combine different sources together at interactive rates for a variety of compositing applications. This framework has been successfully used for stitching unrelated photos with roughly similar overlapping regions [25]. The main limitation of these methods is their inability to *deform* the inputs when combining images with large viewpoint, textural or

structural differences. As mentioned before, misalignments can cause "color bleeding" artifacts in the gradient blending step [14].

ShiftMap [26] is a recent graph cut based image editing method that showed some impressive image completion, retargeting and reshuffling results but it can not be extended to general transformations of the source data. This method uses graph labeling to decide how to rearrange and image to put it in a new context and also uses gradients as an extra feature for labeling. ShiftMap and PatchMatch can produce similar results when carefully tuned, but ShiftMap cannot be extended to general transformations of the source data. Similar algorithm has been proposed by Gal et al. [27] to seamlessly blend different sides of a texture taken from different views.

2.4 Patch-based synthesis

Patch-based synthesis methods have become a popular tool for image and video synthesis and analysis. Applications include texture synthesis, image and video completion, retargeting, image reshuffling, image stitching, new view synthesis, morphing, denoising and more. We will next review some of these applications.

Efros and Leung [28] introduced a simple non-parametric texture synthesis method that samples patches from a texture example and pasting them in the synthesized image. Later research modified the search and sampling approaches for better structure preservation [2, 29, 30, 31]. The greedy fill-in order of these algorithms sometimes introduces inconsistencies when completing large holes with complex structures, but Wexler et al. [32] (and later Kwatra et al. [33]) formulated the completion problems as a global optimization, thus obtaining more globally consistent fills in larger missing regions. All of the synthesis approaches in this thesis belong to this family, but addresses robustness to the presence of slight orientation, scale, illumination or color deviations of the source patterns with respect to their desired appearance inside the hole. By adding an additional objective term

capturing local similarity of the source to the target [34, 35], additional applications are possible, such as image and texture summarization, stitching collages and image morphing [36]. These methods are effective when the sources have similar textures and colors, but otherwise produce a visible feathering effect in the transition between different photos/frames.

Barnes et al. [4] accelerated this family of techniques using PatchMatch, a fast randomized patch search algorithm. This method has been extended to search over rotations and scales for computer vision applications [37], as well as a search of the bias and gain per color channel to find correspondence between different photos of shared content [20]. The recent work by Mansfield et al. [38] attempted to use Generalized PatchMatch for image completion. However expanding the transformation space alone gives too much freedom to the algorithm, thus resulting in convergence to a bad local minimum (we will get back to this observation later on). Their conclusion corroborates our observation by showing poor results for natural images even when initializing the hole with the *original* colors.

Several works extended the patch-based energy function to improve robustness of image completion. Kawai et al. [39] used patch contrast in their energy to compensate for the contrast differences, and Arias et al. [40] include a gradient term in the patch similarity and apply a L_1 norm for gradients to handle regions with high details textures. Our method shares some similar components, but as we show in and Figures 3.1 and 3.5, our method combines several strategies such that each technique complements the rest. In addition, our framework is much more general and allows a range of different applications with completion being one of them. Distances between patch color histograms were used in [41] in addition the L_2 norm on pixel colors to avoid blurriness, as histograms are robust to geometric transformations. This significantly slows down the algorithm and we found it unnecessary in our method.

2.5 Discussion

Small misalignments can be addressed as a postprocess [42] or using a more complex warping [43], but these solutions are not general enough for larger misalignments and texture differences. Other methods combine different images with simple feathering of the boundaries or using a large dataset of web photos [44], or assume the object can be easily segmented [45] but without solving color incompatibilities with the background. Our method resynthesizes the transition region, and effectively warps, stitches and blends colors in the same unified framework. It can automatically eliminate small objects and reduce redundancy for a coherent appearance of the output panorama, and interpolate textures when needed.

Chapter 3

Image melding

The issue of blending or stitching image regions arises in a range of image editing problems. It is well-known as a core issue in constructing panoramas from image sequences [46], and in cutting-and-pasting from a source image to a destination image [6]. But it can also be relevant in many other cases, such as image completion [47], in which the image contents to be replaced must blend seamlessly with their surroundings. Several classes of computational tools have been developed to address this issue, including graph cuts [24], gradient-domain blending [10], and patch-based synthesis approaches [32].

Although these methods often work well in their target applications, they each have limitations in a general image manipulation framework. For example, graph cut/gradient domain blending methods often work well for combining overlapping images, but cannot fill gaps between the stitched images (see Figure 3.10). They can combine regions of different color and intensity, but cannot change textural and structural properties in the source images (see Figure 3.6). In contrast, patch-based based methods can complete holes and gaps, stitch images and compensate for small mismatches in texture and structure, but produce blurry outputs when the inputs have large color and texture discrepancies (see Figure 3.2).

We propose a general framework for image manipulation that augments patch-based syn-

thesis algorithms, improving their flexibility and addressing many problems that previously required the use of multiple independent computational tools. Our algorithm can complete holes with image regions that differ in scale, orientation, color, and brightness from any other content outside the hole (see Figure 3.2), smoothly interpolate between two different texture samples (Figure 3.4), stitch panoramas with large viewpoint and visibility changes (Figure 3.10), and clone an image region into a destination image with substantially different color and texture properties (Figure 3.6). We achieve all this through an energy minimization framework that combines the benefits of patch-based, gradient-based and texture interpolation approaches into a unified method.

Specifically, the proposed method addresses the problem of combining inconsistent image sources when synthesizing a single region. By "inconsistent," we mean that the image contents can have different orientations, scales, color palettes, or textures, making the matching and combination processes difficult. The field of texture interpolation (e.g., [48]), in which two or more inconsistent input texture samples are used to synthesize a gradual transition from one texture to another, is thus an important area of related work. However, existing methods for texture interpolation do not work with general images, because they typically assume inputs that are stochastically homogenous, and some methods require a manually-constructed feature map.

The proposed approach is inspired by previous patch-based methods [4, 32, 34] that produce promising synthesis results when the images are consistent. However, for inconsistent input images, these algorithms fail because their energy function minimizes appearance differences between input and output, typically measured using Euclidean distance of patch pixel colors. Yet a seamless blend between regions requires synthesizing contents that may not be similar to either source under this metric. This leads to one of the key observations of the proposed work: we can modify the similarity metric using a transformation on the patches. We compensate for both geometric and photometric transformations to address structure and texture alignment as well as color and intensity inconsistencies.

An additional observation is that humans are very sensitive to gradient inconsistencies which motivated the gradient-domain methods [49, 10, 18]. These showed impressive image editing and cloning results by locally manipulating gradients instead of pixels, and then integrating the color field. This local adjustment of gradients leads to a globally smooth transition of intensity and color - a property that is lacking in patch-based methods. This leads us to a second contribution of the proposed work, combining the capabilities of patch-based approaches and gradient-domain methods into a single framework that solves more challenging problems than any of these approaches alone.

To illustrate the proposed method, we present results in the following four application areas of image manipulation: image completion, image blending, morphing, and warping. In Chapter 3.1.1 we present our results for image completion (e.g., hole filling). We also show that our method is well suited for the multi-source image completion case, where additional images containing parts of the missing region under different camera viewpoint and illumination are provided. In Chapter 3.1.2, we demonstrate a new patch-based image blending method that allows for gradual transitions from structured detail in one source region to the other. This can be utilized when stitching panoramas with large parallax shifts, object cloning with complex backgrounds, and even "pure" texture interpolation. We also show that patch-based methods and Poisson cloning approaches are both special cases of our proposed method when certain terms are inactive. So rather than simply adjusting the colors through Poisson blending we can also perform synthesis to match not only the colors but also the structures and textures in the intermediate region. A related application is image cloning, where the user composites two images and specifies a region where the algorithm will perform texture synthesis to fit the two images seamlessly together. Our framework enables us to produce cloning results in cases where previous approaches would fail, such as when there is textured detail around the cloned object that had to be synthesized with the target image. In Chapter 3.1.3, we take a step further and perform interpolation both temporally and spatially to accomplish image morphing. Our technique enables more continuous morphs between completely different images than ex-

isting methods.

3.1 Image melding algorithm

The proposed method belongs to the *patch-based optimization* family of image and video synthesis methods [32, 33, 34, 35, 4, 36]. These methods pose the synthesis task as an optimization problem with the objective that every small patch (typically of size 7×7 pixels) centered around every pixel in the output image, must be similar to some other patch in the input, under some task-specific constraint (e.g., the boundaries of the hole in completion [32], the output size in retargeting [34], and other high level constraints [4]). Some of the above synthesis tasks require bidirectional similarity [4, 35, 36] in which a converse term is added to the energy function, requiring every patch in the input to be similar to some patch in the output. By enforcing these local similarities at multiple scales the outputs tend to look globally coherent.

These objective functions are often optimized using an alternating optimization in which each iteration consists of two steps - nearest patch search and color voting - iterated until convergence, and repeated across scales in a coarse-to-fine fashion. Excellent image and video editing results were obtained using this method, and using the PatchMatch algorithm for nearest neighbor search [4] they can often be performed at interactive rates. However they usually work best for a single input with substantial textural redundancy, in which the synthesis can be done by combining shifted local replicas only.

We advance the family of patch-based synthesis methods by generalizing their core objective function in two ways. First, we enrich the space of possible source patches using geometric and photometric transformations. And second – inspired by gradient domain editing methods [11, 10, 18, 13] – we add gradients to the patch color representation, which necessitates replacing the color voting step with a solution to the Screened Poisson equation [50] in the inner optimization loop. These changes not only significantly

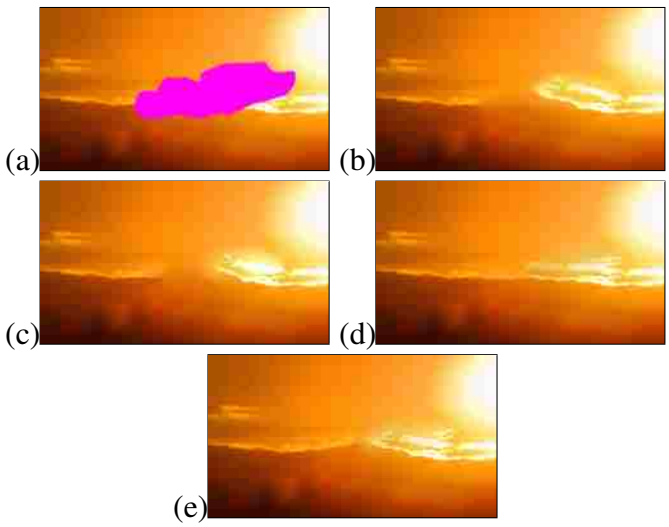


Figure 3.1: *Analysis of our completion method by eliminating components: (a) input hole (magenta); (b) no gain and bias correction per channel; (c) using only color patches (no gradients); (d) no rotation and scale search, and (e) full method.*

improve the capabilities of these methods with existing single source tasks (e.g., image completion), but also enable new single source tasks (e.g., texture aware warps) as well as tasks that require multiple sources with spatially varying weights (e.g., image stitching and cloning).

For simplicity, Section 3.1.1 introduces the new algorithm in the context of the single source image completion task. In Section 3.1.2 we generalize this algorithm to the multi-source variable weight case.

3.1.1 Generalized patch based synthesis

Single source image completion – In the simple image completion case we are given a user-defined mask dividing the image into source region S and target region T (the "hole"), and the objective is to replace the contents of region S using contents from region T. As discussed above, this task is posed as a patch-based optimization problem with the

following energy function:

$$E(T,S) = \sum_{\substack{q \subset T \\ Q = \mathcal{N}(q)}} \min_{\substack{p \subset S \\ P = f(\mathcal{N}(p))}} (D(Q,P) + \lambda D(\nabla Q, \nabla P)), \tag{3.1}$$

where Q is a $w \times w$ patch with target pixel q in its center, and $P = f(\mathcal{N}(p))$ is a $w \times w$ patch that is a result of a geometric and photometric transform f applied on a small neighborhood $\mathcal{N}$ around source pixel p. All patches have five channels: three color (in L*a*b* color space) and two gradient channels of the luminance at each pixel ($L, A, b, \nabla_x L$ and $\nabla_y L$). However, to simplify our notation, we will heretofore use P (or Q) to denote only the three color channels of the patch, and ∇P (or ∇Q) to denote the two luminance gradient channels. The transformations f encompass translation, rotation, non-uniform scale and reflection, as well as gain and bias in each channel. These transformations are limited to predefined ranges that can vary depending on the task or on prior information (e.g., small expected geometrtic variations). D is the sum of squared distances (SSD) over all channels, and the gradient dimensions are weighted by λ w.r.t the color dimensions. This energy function defines that the optimal fill should look everywhere locally similar (under some transformation) to some location within the target. This energy function resembles the one from Wexler et al. [3] up to two main differences:

1. We search over local geometric and appearance transformations of the patches in the source, as opposed to only shifted patches. This is important, as natural and man-made scenes often have self-similar repeating structures that appear at different scales, orientations and colors. These local transformations enrich the space of possible examples, thus obtaining more plausible filled contents. It is also especially important in the case of multiple sources, which contain much larger variations in geometry and color. Later we will show how these transformations can improve the quality of the results.

2. We include patch gradients in our distance metric in addition to colors. This is mo-

tivated by the high sensitivity of the human visual system to gradients, as observed in the perception and gradient-domain editing literature [10, 18]. It is also easier to handle large scale illumination changes with gradients than with colors. Note that by adding gradients to our representation we are not only boosting the high frequencies of local descriptors/patches, as seen in other editing methods [24, 26], but we are also affecting the "voting" step that updates the colors. We will show next that color averaging is no longer sufficient as a step in our optimization, and the optimal voting requires solving the Screened Poisson equation in each iteration.

Wexler et al. [3] proposed an iterative algorithm to optimize their objective function. They showed that it is an Expectation Maximization (EM) algorithm that alternates in every scale between two steps - patch *search* and color *voting*, and each step is guaranteed to decrease the energy function. In the search step, similar (nearest neighbor) input patches are retrieved for all overlapping patches in the output. These patches are then blended together in the voting step by averaging the color "votes" that each such patch casts on every output pixel, resulting in a new output image. The iterations continue until the colors converge, and are repeated across scales in a coarse-to-fine fashion. Our algorithm is similar to [3] and we now detail the changes required to the search and voting steps to guarantee that each reduces our energy function (Eq. 3.5).

Search - To find the closest patch P in Eq. 3.1 we used the generalized PatchMatch algorithm [37] (which extends [4]). Barnes et al. showed that it is possible to find efficiently dense approximate nearest neighbor target patches for all source image patches, with a search space of three degrees of freedom: translations, rotations and scales. We extend the search space further to handle reflections and non-uniform scale, as these transformations occur often in natural images, and were crucial in some examples. In Fig. 3.1, we show how these extensions improve the quality of our image completion algorithm. In order to obtain invariance to small illumination, exposure and color changes, we follow HaCohen et al. [20] and apply gain g and bias b adjustments in each channel of a

source patch to best match the target patch (in the L_2 sense). We limit these adjustments within some reasonable predefined ranges. These are computed as follows: $g(P_i) = \min\{\max\{\sigma(P_i)/\sigma(Q_i), g_{min}\}, g_{max}\}$, $b(P_i) = \min\{\max\{\mu(P_i) - g(P_i)\mu(Q_i), b_{min}\}, b_{max}\}$, where $i \in L, a, b, \nabla_x, \nabla_y$, $\sigma()$ and $\mu()$ are the standard deviation and mean of the input patch at each channel i, and $[g_{min}, g_{max}]$ and $[b_{min}, b_{max}]$ are the gain and bias ranges. These gain and bias are used to adjust the colors of the patch P_i: $P_i \leftarrow g(P_i)(P_i + b(P_i))$.

Voting - Eq. 3.1 is quadratic in all patch terms, where every target pixel participates in $w \times w$ terms — one for each overlapping patch. Therefore, the optimal target image satisfies:

$$T = \arg\min_I \{D(I, \overline{S}) + \lambda D(\nabla I, \overline{\nabla S})\}, \tag{3.2}$$

where the values $\overline{S}$ and $\overline{\nabla S}$ at pixel (i, j) correspond to:

$$\begin{aligned} \overline{S}(i,j) &= \sum_{\substack{k=1\ldots w \\ l=1\ldots w}} \frac{P(i-k, j-l)}{w^2} \\ \overline{\nabla S}(i,j) &= \sum_{\substack{k=1\ldots w \\ l=1\ldots w}} \frac{\nabla P(i-k, j-l)}{w^2} \end{aligned} \tag{3.3}$$

and $P(i, j)$ is the nearest patch in source S to the target patch $Q(i, j)$ (assuming that the top right of the patch is its coordinate). The gradient channel $\overline{\nabla S}$ is assigned in the same manner. For the complete proof please see Section. 3.2. Interestingly, we find that the proposed energy function reduces to the Screened Poisson equation [50, 18] applied to the color and gradient channels computed using the original average-per-pixel "voting" method of Wexler et al.. For an efficient solution of Eq. 3.2, we extend the fast method of Farbman et al. [15] to the Screened Poisson equation. Please see Chapter 3.3 for details.

We continue to alternate the *search* and *voting* steps until convergence — or, in practice, stopping the iterations after 10-30 iterations, more at coarse scales and less at fine scales. The process is repeated at multiple scales in a coarse-to-fine manner, using a Gaussian pyramid and initializing with colors interpolated from the hole boundaries with inverse

Algorithm 1 EMIterationsHoleFilling()

Input: Input image S and "hole" mask of pixels to be filled

Output: Final image T

1: Downsample S and "hole" to coarsest scale d_0
2: Initialize T
3: **for** Scale d from $d_0 \rightarrow 1$ with step size d_s **do**
4: If $d > d_0$, downsample S and upsample last T to scale d
5: **for** EM iteration $k = 1 \rightarrow n$ **do**
6: $T', \nabla T' \leftarrow$ ReconstructImage(S, T)
7: $T \leftarrow ScreenedPoisson(T', \nabla T')$.
8: **end for**
9: **end for**

distance weighting [3]. Note that, as in [3, 34], each step in our algorithm is guaranteed to reduce the objective (Eq. 3.1) [1]. Although this coordinate descent method finds only local minima to the overall objective, the minima we obtain are often visually plausible. We include pseudo-code of our algorithm in Alg. 1.

Multi-source image completion – The utility of our new framework is even more evident when it is necessary to combine pieces from different sources, such as when trying to fill a hole in a target image using other images as sources: for example, unstructured photos from the web, or from a personal album containing shared content. These photos often exhibit large viewpoint, illumination, color and exposure changes relative to the target image. Our method can handle these variations by extending Eq. 3.1 to multiple sources $\{S_1 \dots S_N\}$ in the following way:

[1] Assuming an exact nearest neighbor method is used during *search*. In practice the error of the randomized algorithm we use [37] is very small.

Algorithm 2 ReconstructImage()

Input: source image S and target image T we want to reconstruct

Output: reconstructed image I

1: Initialize $I = 0$ (of the size of T with 5 channels).
2: Generate full-resolution scale space pyramid S_p for image S.
3: **for all** pixels $q \subset T$ with coordinate i, j **do**
4: Create target patch Q with coordinates $i' = i, \ldots, i + w$, $j' = j, \ldots, j + w$.
5: Use PatchMatch to find the best matching source patch P in S_p under the search range for translation, scale, rotation, reflection, non-uniform scale, gain, and bias.
6: Calculate vertical and horizontal gradients (∇_x, ∇_y) of P.
7: **for all** the coordinates i' and j' **do**
8: **for** channel $c = \{r, g, b, \nabla_x, \nabla_y\}$ **do**
9: $I(i', j', c) \leftarrow \frac{I(i',j',c)+P(i'-i,j'-j,c)}{w^2}$
10: **end for**
11: **end for**
12: **end for**

$$E(T, \{S_1 \ldots S_N\}) = \tag{3.4}$$
$$= \sum_{Q \subset T} \min_{P \subset \{f(S_1),\ldots,f(S_N)\}} (\|Q - P\| + \lambda \|\nabla Q - \nabla P\|)$$

This implies that patches P can now come from *either* of the sources to match target patches Q, within the space of admissible transformations. Fig. 3.2 shows a comparison of our method to the naive approach of warping a single region from one of the sources using a homography, followed by gradient blending (similar to Whyte et al. [51]). These examples show that in the presence of a complex 3D scene with viewpoint and illumination differences, a simple copy-paste approach cannot suffice. Our method can combine pieces from multiple sources in a more flexible way, leading to more coherent results.

3.1.2 Multi-source spatial blending

Although the approach described thus far is sufficient for hole filling, we find that it is still insufficient for applications like image stitching and object cloning, in which two sources differing in color, texture and structure must be combined in nearly arbitrary ways. In these applications we want to *gradually* transform from one source to the other within a transition zone separating the two sources. Although previous gradient domain stitching techniques [10, 11] focused on blending color, none of them blend both texture and structure differences also. Adding gradients to our patch based framework helps with the color transition aspect but not with texture. Choosing the best patch from either of the sources — as we described for hole filling — leads to an abrupt change in texture between the two sources (see Fig. 3.6). Therefore we want our method to give us direct control over the influence of each source at each point in the transition area. Our solution is inspired by the Regenerative Morphing method [36] that showed how to *temporally* morph two different images using a patch based approach.

The simplest way to obtain a smooth transition between two regions is by using alpha blending: $T = \alpha_1 S_1 + \alpha_2 S_2$, where $\alpha_1 = \alpha$, $\alpha_2 = 1 - \alpha$ and α changes linearly from 0 to 1. However this approach can easily produce "ghosting" and feathering artifacts due to lack of alignment of high-frequency edges and structure between the sources. Thus, Ruiters et al. [52] applied a non-linear warp to the patches before alpha blending them (though using a manual external feature map) for texture interpolation.

Our method combines the benefits of gradient domain blending and texture interpolation in one unified patch-based optimization framework, building upon the objective presented in Chapter 3.1.1. In order to obtain a spatially gradual blending between sources S_1 and S_2, the optimal result T in the transition area should minimize the following objective function:

$$E_{blend}(T, \{S_1, S_2\}) = \alpha_1 E(T, S_1) + \alpha_2 E(T, S_2), \tag{3.5}$$

This objective requires the patches in T to be similar to *both* S_1 and S_2, where the relative contribution of each source transitions gradually from one source to another as in alpha blending. Again, we use an iterative method to minimize this objective, where each iteration consists of the following four basic steps: *search* for nearest neighbor patches in both sources, *vote* for colors and gradients in each of the two sources independently, *blend* the colors and gradients from the two sources using the given α, and finally *integrate* the blended colors and gradients by solving the Screened Poisson equation, using the two source colors at the boundaries as boundary conditions. See Algorithm 3 for more details.

This algorithm combines the benefits of alpha blending and gradient domain methods in three ways. First, edges and structures are aligned before blending by a *search* across geometric variations, and warping the patches accordingly during *voting*. Second, wide photometric and appearance variations can be matched by the use of a gain and bias per channel as well as matching of gradients. Third, *integration* of colors and gradients using the Screened Poisson equation allows local patch-based edits to propagate globally, leading to smooth and gradual transition of color from one source to another, similarly to traditional gradient domain methods.

One caveat in the above algorithm is that a simple average of gradients tends to wash out small details when they are not perfectly matching after alignment. Perez et al. [10] made a similar observation about averaging gradients for combining different sources and used a maximum-norm per pixel operator instead. Others [53] observed that since gradients are sparse in natural images, one should use robust norms (L_p with $p = 1$ or lower) for optimization terms involving image gradients. We handle this problem in a similar way by replacing the weighted L_2 norm with an L_0 norm, leading to a weighted maximum instead of weighted averaging in the *blending* step. See more details in Section 3.2.1. The effects of this operator are demonstrated in Fig. 3.5.

Section 3.4 shows results of applying this method on challenging object cloning and image stitching examples displaying differences in color, texture and structure. We found

Algorithm 3 EMIterationsBlending()

for scale d from $d_0 \rightarrow 1$ with step size d_s **do**
  for for EM Iteration $k = 0 \rightarrow n$ **do**
    $T_1 \leftarrow ReconstructImage(T, S_1)$
    $T_2 \leftarrow ReconstructImage(T, S_2)$
    $T = \alpha_1 T_1 + \alpha_2 T_2$
    if $\alpha_1 |\nabla T_1| > \alpha_2 |\nabla T_2|$ **then**
      $\nabla T \leftarrow \nabla T_1$
    else
      $\nabla T \leftarrow \nabla T_2$
    end if
    $T \leftarrow ScreenedPoisson(T, \nabla T)$
  end for
end for

the most challenging task was texture interpolation, in which the challenge is to gradually transform one texture into another, interpolating both color and structural differences. Our method handles this case as well, showing comparable results to previous methods ([52, 48]) that were tailored solely for this application. This application also demonstrates the necessity of each component in our method. Fig. 3.5 compares our complete method against regular patch-based stitching synthesis ([34, 4, 37]), our method without gradients, our method without geometric deformations (rotation, scale, reflection), our method without photometric deformations (gain+bias per channel correction), our method without alpha-weighted blending (instead using multi-source image completion) and finally a result by the state-of-the-art patch-based texture interpolation method of Reuters et al. [52].

This comparison shows clearly the utility of each of the components of our method: gradients help with preserving edges and structure, and aid in smooth long-range color interpolation; geometric and photometric deformations help find matches for sources with

Algorithm 4 EMIterationsMorphing()

for scale d from $d_0 \rightarrow 1$ with step size d_s **do**
  for for frame $k = 1 \rightarrow F$ **do**
    for for EM Iteration $k = 0 \rightarrow n$ **do**
      $T_1 \leftarrow ReconstructImageBDS(T(k), S_1)$
      $T_2 \leftarrow ReconstructImageBDS(T(k), S_2)$
      $T_3 \leftarrow ReconstructImageBDS(T(k), T(k-1))$
      $T_4 \leftarrow ReconstructImageBDS(T(k), T(k+1))$
      $T \leftarrow \alpha_1 T_1 + \alpha_2 T_2 + \alpha_t T_3 + \alpha_t T_4$
      $i_{max} \leftarrow \arg\max_{i=1...4}\{\alpha_i \nabla T_i\}$
      $\nabla T \leftarrow \nabla T_{i_{max}}$
      $T(f) \leftarrow ScreenedPoisson(T, \nabla T)$
    end for
  end for
end for

different content and allows more flexible and less repetitive synthesis; and spatially gradual blending using the weighted L_0 norm forces a continuous transition both in color and texture.

3.1.3 Multi-source temporal blending

In the previous section we showed how we could spatially interpolate the transition regions between two different image sources. Shechtman et al. [36] used a similar patch-based optimization method with a related source blending scheme, to *temporally* interpolate two different images, showing impressive automatic morphing results on unrelated images. Following their objective, we pose the morphing task as an optimization for all frames

$T_{1...K}$ given the two sources S_1 and S_2:

$$E_{morph}(T_{1...K}, \{S_1, S_2\}) = \sum_{k=1}^{K} \{\alpha_1 E_{bds}(T_k, S_1) + \tag{3.6}$$
$$+ \alpha_2 E_{bds}(T_k, S_1) + \alpha_t E_{bds}(T_k, T_{k-1}) + \alpha_t E_{bds}(T_k, T_{k+1})\}$$

This objective is similar to the source blending objective from Eq. 3.5, with the following differences: First, in addition to an alpha weighted similarity to the two sources, it requires similarity of each frame to its neighboring frames T_{k-1} and T_{k+1}; Second, it uses Bidirectional Similarity (BDS) [34] as the basic patch-based similarity measure between images. BDS combines the patch-based term from Eq. 3.1 with another term that sums distances for all patches in the source S to their nearest neighbor in the target T: $E_{bds}(S, T) = E(S, T) + E(T, S)$. The latter term helps ensure that the content from the source will appear in the target and avoids converging towards excessively smooth and repetitive solutions. This objective is optimized using a similar iterative algorithm to the ones described earlier. See Alg. 4 for more details.

Our proposed energy function differs from that of Shechtman et al.in one main point: they claimed that simple alpha blending of patches leads to blurry results, and therefore introduced a fifth terms called α-*Disjoint Coherency* (not required in our method), varying the portion of patches sampled from each source. This heuristic helps maintain sharpness in some cases but is not as general as our use of the blending coefficient. In Chapter 3.4 we show that our method can handle images with substantially larger geometric and color differences while preserving sharpness.

3.2 Proofs

3.2.1 Texture Interpolation

During synthesis, we have two voted images $\overline{T}_1$ and $\overline{T}_2$ containing color and gradients from the corresponding sources. We need to combine these together to get a final color and gradient, prior to Poisson integration (see Alg. 3). The texture interpolation energy is defined as:

$$\begin{aligned} E =& E_{\text{color}} + E_{\text{gradient}} \\ =& \sum_{i=1}^{2} \alpha_i \|T - \overline{T}_i\|^2 + \alpha_i \|\overline{\nabla T}_i\| \|\nabla T - \overline{\nabla T}_i\|_0. \end{aligned} \tag{3.7}$$

Here T is the unknown target pixel color, $\overline{T}_i$ is the voted pixel color, α_i is the interpolation parameter, and gradients are indicated using ∇. This energy has an L_0 term and makes the optimization problem *NP*-complete [54]. In the Compressive Sensing community it has been shown that in some specific conditions the L_0 problem can be reduced to L_1, however common L_1 solvers are too slow for large problems like ours. Moreover, many recent greedy solvers have been shown to be able to efficiently *approximate* the solution. Our solution for solving Eq. 3.7 is a greedy approximation and resembles [55]. The solver iteratively makes a greedy choice between the source to be used for each pixel and then according to this choice, the method uses L_2 least square solver (screened Poisson solver) to evaluate the final values. Our fast greedy solution converges to an acceptable local minimum and in more than 90% of the iterations it decreases the energy in Eq. 3.7 compared to its value in previous iteration. Exploring more sophisticated solvers is left for future research.

We specifically take a greedy downhill step in Eq. 3.7 by minimizing separately the colors and gradient energies. Minimizing separately the target color gives a simple linear interpolation for color: $T = \sum_{i=1}^{2} \alpha_i \overline{T}_i$. The optimal gradient ∇T can be found by noting

that when E_{gradient} is at a minimum, at least one of the zero norms must be zero. So ∇T is simply one of the gradients $\overline{\nabla T}_i$, specifically the gradient $\overline{\nabla T}_i$ for which $\alpha_i \|\overline{\nabla T}_i\|$ is maximal. That is, we choose the source gradient which has maximum magnitude after weighting by α_i. This gives rise to the conditional in lines 6-10 of Alg. 3.

3.2.2 Voting and the screened Poisson Equation

We demonstrate that minimizing the patch energy of Eq. 3.1 is equivalent to solving the discrete screened Poisson equation [50] using the mean gradient and color of the overlapping patches. Recall that Eq. 3.1 is optimized by an alternating optimization, where we first find nearest neighbor patches that decrease the energy, and then "vote" using the proposed overlapping patches to further decrease the energy. Thus, we want to find image T minimizing:

$$E(T,S) = \sum_{\substack{q \subset T \\ Q = \mathcal{N}(q)}} D(Q, \mathrm{NN}(Q)) + \lambda D(\nabla Q, \nabla \mathrm{NN}(Q)), \tag{3.8}$$

where Q are overlapping patches in the output target image T, $\mathrm{NN}(Q)$ is the nearest neighbor source patch to Q, and D is sum-squared difference as before. Now we use an identity of quadratic forms:

$$\frac{1}{n}\sum_{i=1}^{n}(a-b_i)^2 = \left(a - \frac{1}{n}\sum_{i=1}^{n} b_i\right)^2 + C(b_1, \ldots, b_n). \tag{3.9}$$

Here C is a constant function of b_i variables. This states that a sum of quadratic forms in the unknown target color a is equivalent to a single quadratic form. The identity can be shown directly by expanding the quadratics, and also applies if any linear operator ∇ is applied to a and b_i. Applying Eq. 3.9 to Eq. 3.8 allows us to replace the sum of quadratics for overlapping patches with a single quadratic per target pixel color and gradient, that is, up to constant factors, Eq. 3.8 is equivalent to:

$$\tilde{E} = \sum (T - \overline{T})^2 + \lambda \|\nabla T - \overline{\nabla T}\|^2. \tag{3.10}$$

Here $\overline{T}$ and $\overline{\nabla T}$ are the averaged overlapping colors and gradients (Eq. 3.1.1). This energy is the discrete screened Poisson equation [50].

3.3 Implementation Details

Search and vote: We use a high order (Lanczos3) sampling filter and a densely sampled scale-space (10 filtered scales with the same resolution of the original image, with no subsampling), for higher quality filtering than previous patch-based method that searched over rotations and scales for analysis applications [37, 20]. We pay with a higher memory load but this allows us to use a simple nearest-neighbor sampling of the patches all the way to the finest scale for faster performance and better quality. We use a few bilinear interpolations at the last EM iterations of the finest scale for best quality. Also, we pre-calculate the Gaussian weighted mean and standard deviation centered at each pixel for the input images and adjust the gain and bias of the patches in the search and vote based on those values [20]. We also use the gain and bias for early rejection of source patches whose gain or bias deviates more than $\times 1.1$ than those of the target patch, in addition to the early rejection based on the distance [4]. Also, usually during the EM iterations the changes happen at boundaries of coherent regions, so we limit the search to happen just at those boundaries at finer resolutions.

Screened Poisson solver: In our method we solve the Screened Poisson equation 3.2 in each EM iteration our method. Bhat et al. [50] suggested a Fourier based solver to the same problem. However it wasn't fast enough when applied many times on large images. Farbman et al. [15] introduced a new efficient way to solve linear translation-invariant (LTI) problems with a pyramidal convolution approach. These include a family of problems like the Poisson equation and Shepard's interpolation, commonly used for

gradient domain stitching and cloning. They reduced the $O(n^2)$ complexity involved with a straight forward convolution with a large kernel associated to the Green's function of the problem, with iterative convolutions with small kernels at multiple scales, resulting in an extremely fast $O(n)$ approximation algorithm. Although not derived in their work, this equation's Green function is "in between" the Poisson function and a delta function associated with the color term, and thus belongs to the family covered by their method. We learned the specific $5 \times 5 \backslash 3 \times 3$ kernels of our problem and use their fast pyramidal convolutions as our solver, taking only a small portion of the overall runtime.

Parameters: In patch based methods the patch size is a crucial parameter. Large patches capture more structure and lead to better synthesis of structures, if good matches are found. However if such a matches are not found the result can easily converges to a blurry solution. Therefore previous methods [3, 34, 4] used smaller patches (e.g., 5×5 or 7×7) that generally lead to sharper and more flexible synthesis (linear structures can slightly bend to better connect) and the expense structural changes. The larger geometric and appearance search space in our method allows us to use larger 10×10 patches while well preserving structures, having flexibility when needed and obtaining sharp results. Unless mentioned otherwise, set the search range to be $[\frac{-\pi}{2}, \frac{\pi}{2}]$ for rotation, $[0.9, 1.3]$ for uniform scale, and $[0.9, 1.1]$ for relative scale (horizontal/vertical). The range of the bias for all the three channels is $[-10, 10]$ and for gain is $[0.9, 1.3]$. The algorithm is fairly robust to variation of these ranges. Additionally, because these parameters are semantically meaningful, e.g. rotation, scale, brightness, and contrast adjustment (gain and bias), it is easy to adjust them in a meaningful way for a particular task. When we have no blending (hole filling, warping) we chose the gradient weight λ to be 0.2 and otherwise we set it to be 0.5. The reason is because effective blending between different textures is easier by blending their gradients than colors. For blending applications, such as cloning and morphing we limited the search range for the offset of the patches to be 0.1 to 0.2 of the image size to avoid irrelevant patches from distant regions. We usually have 30 EM iterations at lower resolution and gradually reducing the iterations until we get to 2 iterations at the finer resolution.

In morphing we start from 6 temporal sweeps over all frames coarser level and reduce it to be 1 at the highest resolution.

3.4 Results

Our implementation is written using Matlab/C++, and the code was designed for versatility and quality rather than performance. The experiments were done on an Intel dual quad-core Xeon X5570 3.06GHz machine. Our method takes about 58 seconds to complete a hole of 0.25 megapixels in a 1340×2048 image. If we use only color patches and do not use any transformations (implementation of Wexler et al. [3] using PatchMatch [4]), the run time is $26sec$, vs. $4sec$ using Photoshop's Content-Aware Fill [56] that is based on the same algorithm. This suggests that a more optimized implementation could be significantly faster. The bottleneck of our method is the search, which is linear in the number of pixels to be synthesized [4]. As with previous patch-based optimization methods that used PatchMatch, intermediate results at coarse scales are obtained at interactive rates, allowing the user to quickly assess the final quality, change parameters and add constraints if needed. Our most computationally demanding application is image morphing, for which we have to synthesize a sequence of frames. This process required a few tens of minutes for a sequence of size $635 \times 456 \times 20$ frames, similar to the runtimes reported by Shechtman et al. [36].

We will now demonstrate results of our method applied to a wide variety of image editing application, and illustrate that it performs comparably to, and often better than the state-of-the-art method for each.

Image completion– State-of-the-art automatic image completion methods, effectively synthesize the content of a hole using shifted exemplars from outside the hole. Often when the hole is large, or the available area outside the hole is small, it could be very hard to produce a good fill by shifting the available examples, regardless of the method used.

Fig. 3.7 shows that our method can still succeed in many of these cases using a richer search space [2]. It can exploit rotational and reflection symmetry, complete edges and textures using examples from different orientations, scales and colors. It can also gracefully handle small lighting and perspective differences that are not captured well by the other methods.

Our method also allows additional relevant photos to be used as source content for completion. This can often happen in a personal photo collection, or with web photos of a popular site or event, where other photos of the same scene contain relevant content of places, objects and people. Most previous methods, could not use effectively this additional data because the shared content appears often at different view points, scale, illumination, exposure, white balance and other camera parameters. Whyte et al. [51] handled the special case of rigid scenes, where a homography transform can bring the corresponding content into good alignment. But in general, aligning photos under these variation is a challenging problem in itself [20]. Fig. 3.2 shows a few examples of our results vs. result of Wexler et al.'s [3] method as well using a manual homograhy computed around the hole to align the sources, followed by gradient domain blending. Our method uses the relevant content in the other images in a plausible way despite the color and viewpoint changes. Even if the correspondence around the hole can be found, a simple copy-paste of the region can often fail as shown in Fig. 3.3.

We have also extended our hole filling framework to the task of texture-preserving warping, similar to [57]. In this task the user defines a geometric warp of an object within the image, and we use our method with the object (or the entire image) as the region to be synthesized such that the original small-scale textural properties of the object are preserved to avoid the appearance of stretching. Instead of using the warping field to render the pixel colors directly, we use it to define a constraint map defining for each pixel in the target image (to be synthesized) the corresponding location and orientation in the source image

[2] In our comparisons, unless stated otherwise, we used Photoshop's Content-Aware Fill [56] as an optimized implementation of [3] using PatchMatch [4].

(the unwarped input). To maintain the constraints, we define a small search window in translation and orientation for each patch (where the window size increases linearly with the distance to the object boundaries), and do not search over scale dimensions in order to avoid stretching of the texture. Fig. 3.8 shows a comparison to [57] on one of the examples in their paper as well a more extreme case. Our method performs well in both cases.

Texture interpolation – We found texture interpolation to be the most demanding application of source stitching. In this case, both color and texture should gradually change from one source to another. A few methods have been introduced to solve this specific problem. Ruiters et al. [52] proposed a patch-based synthesis algorithm that does not use an external dataset, but it requires a *manually* created feature map that marks the "cracks" between the basic texture units. This requirement also limits the types of textures applicable to this method. Our method is fully automatic. Fig. 3.4 shows results of our method applied on a few examples from [52] and direct comparisons can be found in the supplementary material. Both methods give plausible interpolation results, but as we will see next, our method can be applied on any images, while [52] can work on certain texture inputs only and takes a few hours to compute vs. tens of seconds for our method.

Image cloning– Object cloning methods [10, 24, 8] allow the user to define a rough selection around the object, containing some of its nearby background, and paste it seamlessly in a new background in another image. However when the backgrounds contain large contrast textures or structures that do not align, existing methods produce color bleeding artifacts and obvious compositions (e.g. as can be seen in the right side of the hole touching the tree texture in the squirrel example, Fig. 3.6(c) third row). In Fig. 3.6, we compare our method with the Photomontage method by Agrawala et al. [24] on a few cases where the textures are inconsistent. Our system performs more seamless composites. We get a nicer blend between the snow and sand textures in the first row. In the third row, we get a better color compatibility between the squirrel and the tree trunk as we correct the colors throughout the optimization process and at patch level, in contrast to [10] that is based

on gradients at the finest scale and therefore tries to match colors based on pixel wide boundaries.

Panorama stitching– Seamless stitching of panoramas is a hard problem that attracted a few solutions in the past years [58, 43]. Problems arise in the presence of parallax, occlusions and moving objects. Our method resynthesizes the overlap region with some large margins, and can cope with very large changes as demonstrated in Fig. 3.10. Note that our method removes objects or even columns of the building windows to better fit the content into the image, where the space budget is tight.

Image harmonization– Image harmonization [8] cleverly combines image pyramid levels from the sources using smooth histogram and noise matching in order to transfers some textural properties in addition to color and intensity. This method cannot handle structured textures, and tends to transfer the high frequency texture to the object itself rather than only to its surrounding background.

To compare against [8] we adjusted our algorithm, because in this application we want to extract structure from one image and detail from another. The extension was simply to hold the structure image in our cloning process constant, and giving it a large constant importance ($\alpha(i,j) = 0.9$ in our blending formula. In this manner, the structure will come from that image except where it is missing high frequency details. Thanks to our L_0 optimization, those small-scale details will be replicated from the other image. In Fig. 3.9 we show two comparisons against this method: In the first row, a result of applying our method on one of their failure cases, preserving coarse scale orientation properties of the sand texture, and avoiding contamination the hydrant with the sand texture. In the second row, both results are comparable, but we preserve the fine-scale details of the original while replacing the gross structure.

Morphing– In Figure 3.11 we demonstrate blending between two images across time, via three intermediate frames of our automatic morphing output. Our method produces better transitions than Regenerative Morphing [36] on some challenging examples, primarily

because of the large space of deformations and the use of gradient domain blending. Note that in the second row we *automatical* corresponding features are found through our morph between two images of the same scene, but from totally different viewpoint and illumination.

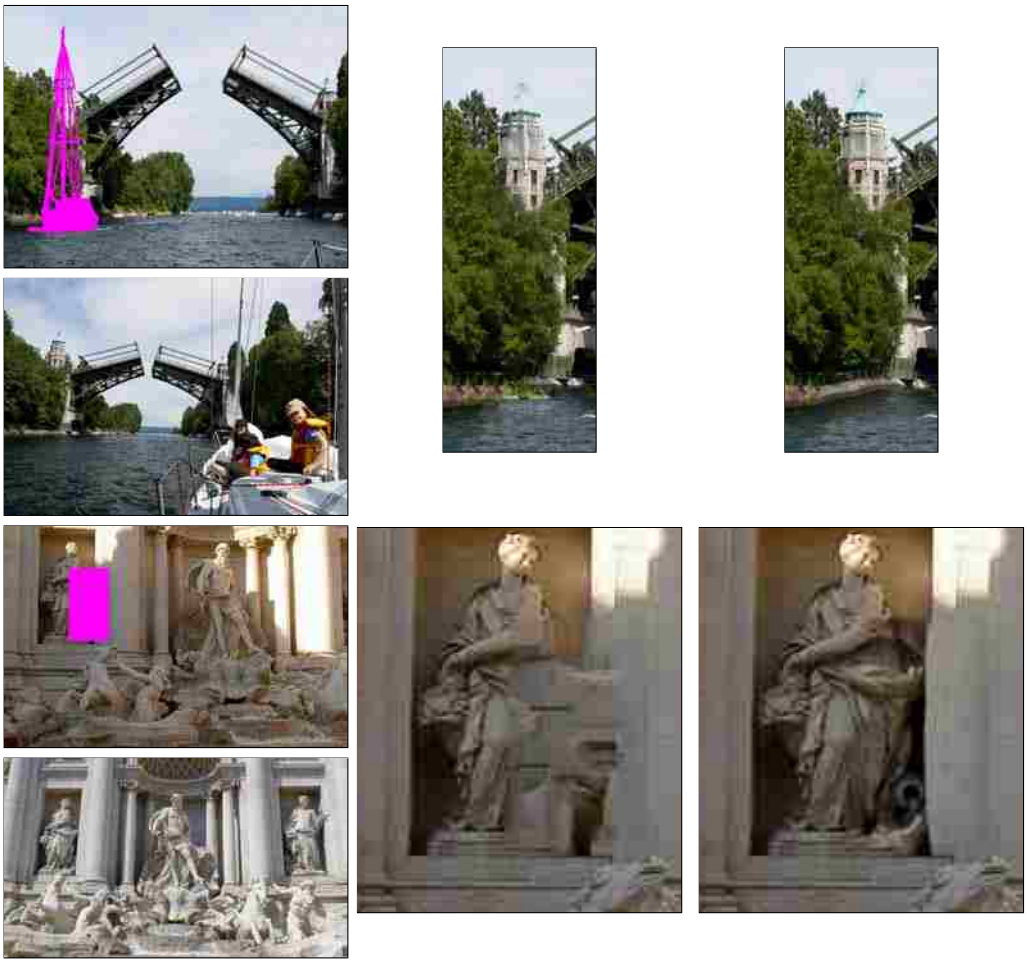
3.5 Sensitivity to parameters

Our algorithm relies on Generalized PatchMatch algorithm [37] to find the best match in search stage and in this thesis we mostly contribute in synthesis part. As Barnes et al. [37] stated, Generalized PatchMatch is fairly robust to the variation of search range. We saw a similar effect in our algorithm where tweaking parameters does not affect the quality of results in most cases (see Fig. 3.12). However, by making the search space wider, the algorithm needs more search iterations to converge to the right answer so as a future work, one could extract patch statistics in a similar way that [59] does and limits the search space based on those statistics.

One can argue that by extending the search space, the risk of convergence to a bad local minimum grows. This could happen as we will show in the limitation section (see Sec. 3.6), though in multi-source examples such as cloning or stitching the risk of bad convergence is lower. The reason is although the algorithm gets more freedom in synthesis stage, at the same time however, it has more constraints as the result of now instead of one source, it has to match to many sources. Therefore, through our generalization we get a right balance of constraints and freedom. Overall, we get a convergence to a good local minimum most of the times.

3.6 Limitations

Our method is not without limitations: in some examples too many degrees of freedom might lead to unwanted distortions (such as line bending). These are visible in Fig. 3.13 (distortions in buildings). Barnes et al. [4] demonstrated that line (and other model based) constraints can provide an intuitive tool for the user to protect important content, and our method can benefit from such constraints in the same way (see Fig. 3.13). A limitation of our cloning solution can be seen in Fig. 3.6 - a large background margin around the object may be needed for a pleasing texture interpolation between very different textures. Of course some textures are simply too disparate to be stitched in a seamless way (e.g., a clear sky would not blend with any coarse texture). Finally, the additional quality obtained by our modifications have sacrificed much of the interactive performance shown in Barnes et al. [4]. However, because the bulk of the additional computation results from filtering and interpolation, we believe our method could be well-suited to GPU implementation.



(a) (b) (c)

Figure 3.2: *Multi-image completion results. (a) a hole is marked (magenta) in a source image, and additional source with different viewpoint, scale, appearance; (b) filling the hole with Photoshop's Content Aware Fill with both sources given; and (c) our method.*

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

Figure 3.3: *Multi-image completion comparisons. (a) a hole is marked (magenta); (b) additional source; (c) filling the hole using Photoshop's Content Aware Fill ; (d) filling by a manual Homography alignment of the region around the hole and Poisson blending (note the discontinuity of the fountain edge); and (e) our method.*

Figure 3.4: *Texture interpolation results. Our method applied on a few examples from Reuters et al. [51]. No manual feature map is used. Both methods obtain comparable results where our method puts more focus on gradually changing the relative density of each texture, whereas theirs changes more the shape thanks to the usage of feature maps. See comparisons in the supplementary material.*

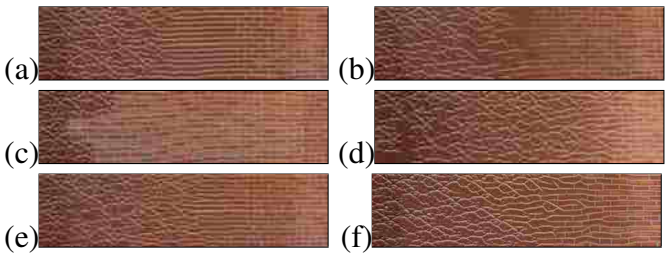
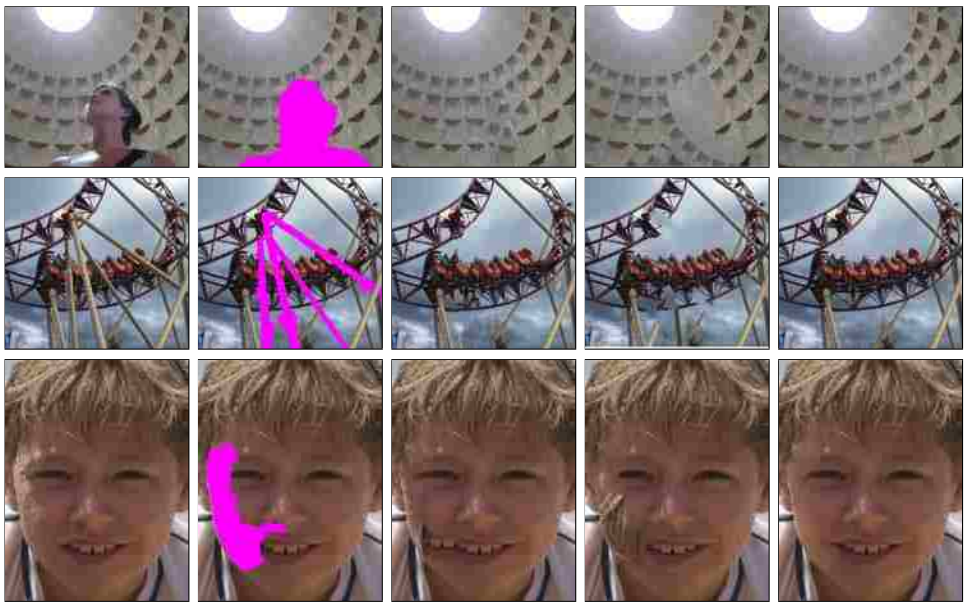


Figure 3.5: *Analysis of our blending method by eliminating components. (a) using only color patches (no gradients); (b) using L_2 norm for gradients instead L_0 when combining sources (Eq. (9)); (c) no blending - use the best patch from either of the sources (Eq. (4)); (d) no gain and bias correction per channel; (e) no rotation and scale search, and (f) full method.*

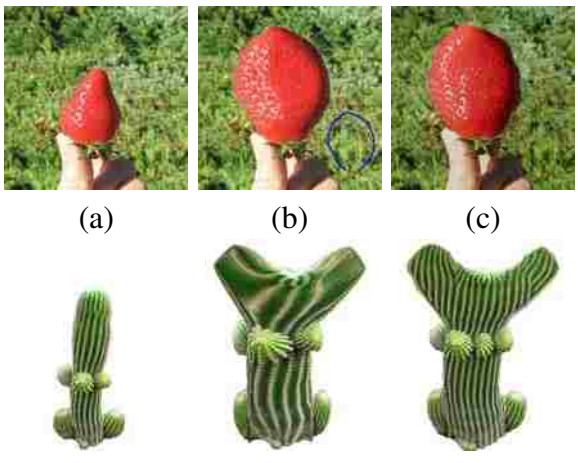
(a) (b) (c) (d) (e)

Figure 3.6: *Seamless image cloning. (a) source image; (b) target image;(c) blending region marked in magenta, (d) Photomontage result ([Agarwala et al. 2004]), and (e) our result. Texture is blended better by our method and as well as we have less color "bleeding" artifacts (such as in (d) for the squirrel).*



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

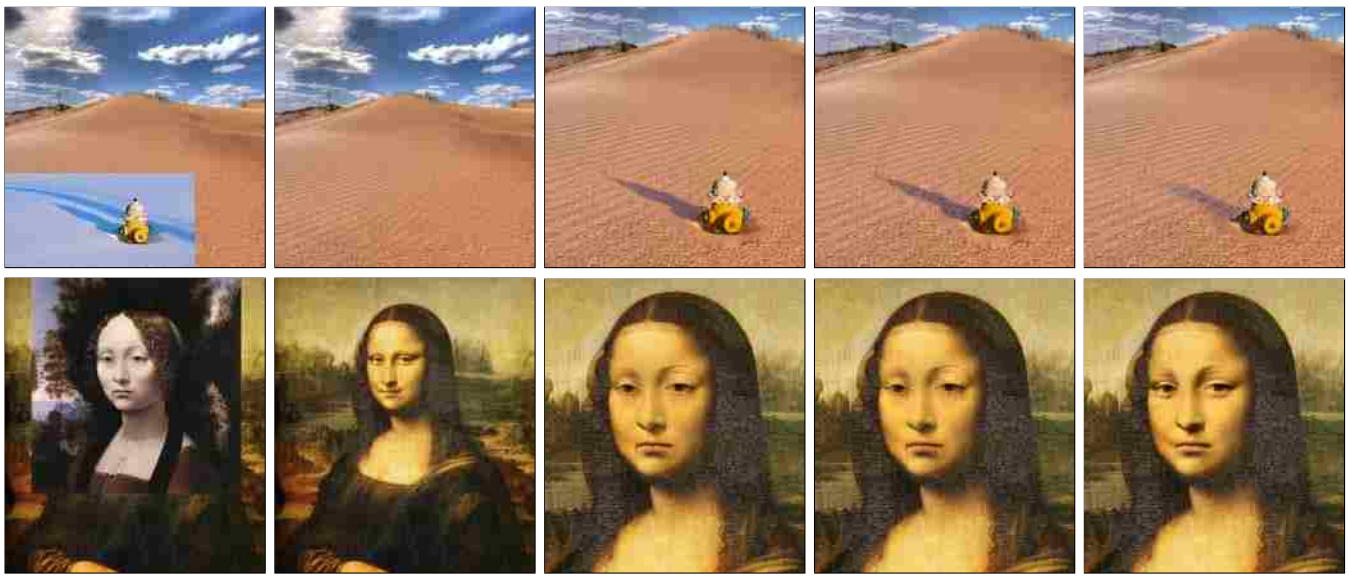
Figure 3.7: *Image completion comparison. Left to right (a) original image; (b) a hole is marked (magenta); (c) hole filled image using Photoshop content aware;(d) output of the Shift-Map, and (e) ours.*



(a) (b) (c)

(d) (e) (f)

Figure 3.8: *Texture preserving warping comparison. Top (left to right): source from [Fang and Hart 2007] along with their result and ours on the right. Bottom: another source, simple warp and our result.*



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

Figure 3.9: *Comparison between our method and Image Harmonization. (a,b) Two examples with two sources from [Kimo et al.];(c) Poisson blending, (d) Harmonization result taken from [Kimo et al.], and (e) our result. In the hydrant example our result preserves better the orientation of the sand texture, and does not contaminate the hydrant. In the Mona Lisa example, our result adopts more of the shadows from the Mona Lisa source (can be controlled) and renders more authentic structured noise patterns.*

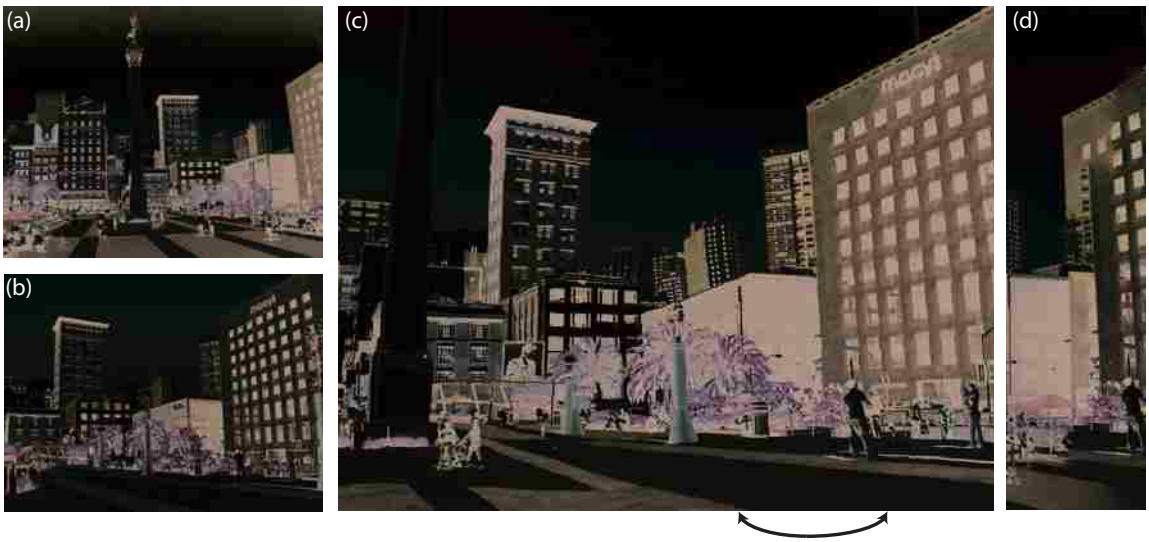
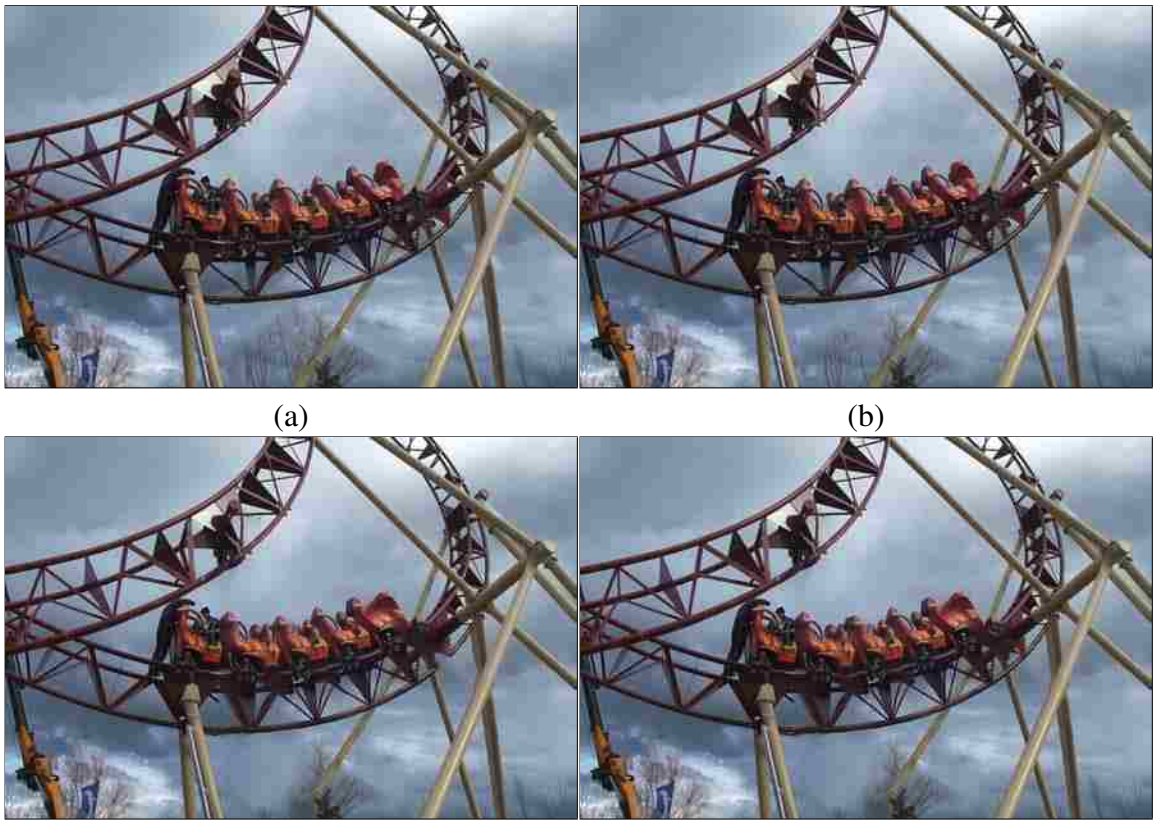


Figure 3.10: *Panorama stitching. Our method synthesizes in (c) a transition area between the two sources (a) and (b) after roughly aligning them with a homography. (d) shows a comparison to Photoshop's Photomerge tool, based on a homography alignment, graph-cut and gradient domain blending. Typical stitching artifacts are visible in (d) due to the large view point change, whereas removes some redundancy (a column of windows in two buildings, and small objects) to put in most of the important content in both source. As in other patch-based methods, adding manual constraints could further protect important content.*

Figure 3.11: *Morphing results. Results of applying our method to morphing different images (another result appears in Fig. 1). Our method handles sources with larger geometric and appearance differences than Regenerative Morphing [Shechtman et al. 2010]. See comparisons in supplementary material.*



(a) (b)

(c) (d)

Figure 3.12: *Examples of output of algorithm with different parameters. (a) default parameters; (b) result with high gradient constraint, the algorithm avoids to put branches in bottom area because gradient is against of using textures to fill the holes ;(c) result with low gradient importance (λ) and low range of rotation and scale, the result gets worse than two previous ones because it does not have enough rotation and scale search range, and also it cannot connect rail roads because it does not have enough constraints to avoid disconnected edges;(d) result with smaller range of rotation and scale but higher gradient importance (λ), in this case result gets slightly better than (c) because it could connect the rail line but worse than default because it cannot use the right rotation and scale.*

(a) (b) (c)

Figure 3.13: *Example of the result with distortion. (a) synthesized image; (b) added line constraint;(c) result with constraint*

Chapter 4

HDR reconstruction

High-dynamic range (HDR) imaging has the potential to transform the world of photography. Unlike traditional low-dynamic range (LDR) images that measure only a small range of the total illumination of a scene, HDR images (HDRI) capture a much wider range and therefore more closely resemble what photographers see with their own eyes. However, despite their tremendous potential, existing approaches for high-quality HDR imaging have serious limitations. For example, specialized camera hardware has been proposed to capture HDR content directly (e.g., [60, 61]), but these devices are typically expensive and are currently unavailable to the general public.

To make high-quality HDR imaging widespread, we must focus on approaches that use standard digital cameras. The most common approach is to take sequential LDR images at different exposure levels (known as bracketed exposures) and then merge them into an HDR image (HDRI) [62, 63]. Although this technique can produce spectacular results (see, e.g., [64]), the original approaches work only for static scenes because they typically assume a constant radiance at each pixel over all exposures. When the scene has moving content (or the camera is hand held), this method produces ghost-like artifacts from even small misalignments between exposures. This is a serious limitation, since real-world scenes often have moving objects and real-world cameras are not often mounted on tripods.

The problem of removing motion artifacts for sequential HDR imaging has been the subject of extensive research and has led to two major kinds of approaches. The first kind assume that the images are mostly static and that only small parts of the scene have motion. These "deghosting" algorithms use all the frames to determine whether a given pixel is static or has motion and then apply different merging algorithms in each case. For static pixels, the traditional HDR merge can be used. For pixels with motion, many algorithms use only a subset of exposures (in many cases only one) to produce a deghosted HDR. The fundamental problem with these techniques is that they cannot handle scenes with large motion if the changing portions of the scene contain HDR content.

The second set of approaches try to handle moving HDR content by first aligning the sources to a reference exposure as a preprocess before merging them into an HDR image. The most successful algorithms use optical flow to register the images together, but these are still brittle and the "aligned" images often do not match the reference very well in cases of large, complex motion. For this reason, alignment algorithms for HDR often introduce special merging functions that reject aligned exposures in locations where they do not match the reference. Therefore, as with deghosting algorithms, they do not reconstruct HDR content in these regions.

We observe that trying to align the images to each other is a difficult problem that can be made easier if we can use information from the HDR result. After all, the exposures overlap in the radiance domain and information from one aligned image can be propagated to another. This led us to the development of a new patch-based optimization that jointly solves for both the HDR image and the aligned images *simultaneously*, which we present in this paper. Our algorithm can handle large, complex motion and during alignment can even fill in information that was occluded in an exposure, something not possible if we were doing simple alignment as a preprocess.

Our algorithm is inspired by recent work in patch-based algorithms in the graphics and vision communities. Researchers have been studying these algorithms because of their

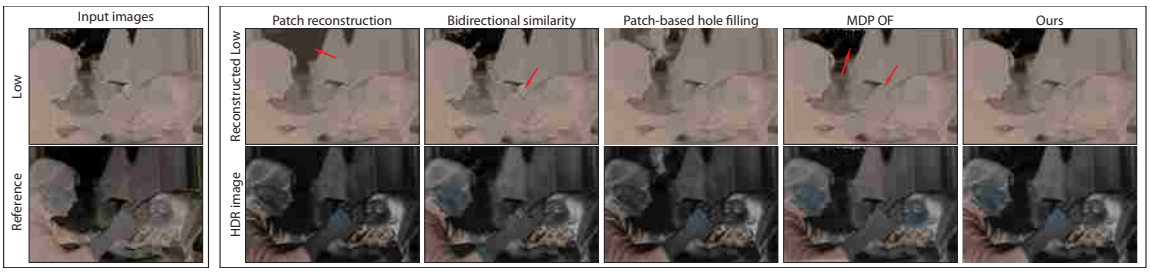


Figure 4.1: *Results from direct application of standard patch-based algorithms and optical flow alignment techniques. First, we might do a single iteration of PatchMatch [Barnes et al. 2009] (as shown in Fig. 3 of that paper) to match the low image to an exposure-adjusted version of the reference. The reference exposure is missing information in the over-exposed regions, so the direct use of PatchMatch simply matches these saturated regions and produces a gray background, defeating the purpose. Second, we might try to use Simakov et al.'s bidirectional similarity metric [2008] to compute a new version of the low image using the lowered reference as a target. However, this does not work either because the image diverges from the desired result. The lady's hand is moved in the low source with respect to the reference which this method cannot register, as indicated by the arrow. We might also label the saturated regions in the lowered reference as an alpha-blended hole and use Wexler et al.'s patch-based holefilling algorithm [2007] to complete it using the low image. Here the boundary condition cannot compensate for the motion and so the algorithm diverges to draw coherently from another region, in this case the face in the low input. Finally, using the motion detail preserving optical flow (MDP OF) algorithm of Xu et al. [2010] to register the low image to the middle has artifacts, indicated by the arrows. Our approach, on the other hand, correctly aligns the exposures and produces a good HDR result.*

power to exploit self-similarities in images to reconstruct information for image hole-filling [65], image summarization/editing [66], and image morphing/view interpolation [67]. However, the direct application of standard patch-based methods to this problem does not work, as shown in Fig. 4.1. For this reason, previous patch-based algorithms have not addressed the problem of HDR image reconstruction.

Our patch-based algorithm, on the other hand, is based on a new *HDR image synthesis equation* that codifies what we want to do: create an HDR image containing HDR information from all the exposures but aligned to one of them, as if it was taken by an HDR camera at the same moment in time. Specifically, this paper makes the following con-

tributions: 1) we pose HDR reconstruction as a new energy minimization problem that jointly solves for the HDR image and the aligned exposures; 2) we introduce a multi-source bidirectional similarity metric for this purpose; and 3) we demonstrate high-quality HDR results from images with complex motion and occlusion that are superior to previous work.

4.1 Previous Work

We begin by reviewing the previous work to remove the HDR ghosting artifacts of dynamic scenes captured with a set of bracketed exposures. A thorough review of HDR imaging is beyond the scope of this paper, so interested readers are directed to texts on the subject [68, 69]. We categorize the two general kinds of proposed algorithms to address the ghosting problem in the subsections that follow.

4.2 Algorithms that reject ghosting artifacts

These algorithms assume the images can be globally registered so that each pixel can be classified as either static or "ghosted" (containing movement across the different exposures). These techniques try to identify ghosted pixels and only use information from a subset of exposures in these locations.

The key differences between these methods is how they detect the ghosting regions. Liu and El Gamal [70] proposed a new sensor model that rejects information from ghosted regions. Grosch [71] mapped pixels from one exposure to the other and used the difference between these values to compute an error map that accounts for motion. Jacobs et al. [72] proposed approaches based on variance and entropy. Jinno and Okuda [73] used Markov Random Fields to detect occluded and saturated regions and exclude them from the HDR

result. Sidibe et al. [74] used the fact that pixel values in static regions usually increase as the exposure increases to identify ghosting. Gallo et al. [75] detected motion between two exposures by measuring the deviation of their pixel values from the expected exposure ratio. Min et al. [76] proposed to compute multilevel threshold maps from the images and compare them to detect motion. Wu et al. [77] used criteria such as consistency in the radiance and color across exposures. Pece et al. [78] computed the median threshold bitmap for each exposure and labeled pixels that did not have the same value as movement. Raman and Chaudhuri [79] used a segmentation algorithm based on superpixel grouping to detect which regions have motion. Finally, Zhang and Cham [80, 81] detected motion by looking for changes in the gradient between exposures.

Some algorithms do not require the explicit identification of ghosted pixels at all. Khan et al. [82] modified the weights of the HDR merging function based on the probability that a pixel is static. Eden et al. [83] used the distance of an exposure's radiance to that of a reference to select a single exposure for each pixel. Heo et al. [84] computed the joint probability density function between exposures to map values from one exposure to another, and then used the Gaussian-weighted distance to a reference value to weight each exposure during merging.

However, none of these deghosting algorithms can produce accurate results when there is moving HDR content since they all assume that a pixel's radiance can be computed from the same pixel (or block around it) in all exposures. Instead, a moving HDR object would have properly-exposed pieces in different parts of the image in each frame. For this reason, these papers all show results using only largely static scenes with small moving objects – none are like that of Fig. 1.3 with a large moving subject. However, these techniques tend to produce fewer artifacts than the optical-flow based alignment methods we will discuss next, and so commercial HDR software typically uses deghosting approaches like these (e.g., [85]).

4.3 Algorithms that align the different exposures

These approaches try to avoid the problem of moving HDR content by aligning the different exposures first and then merging them into the final HDRI. Although the alignment of images has long been studied in the image processing and vision communities (see, e.g., [86, 87]), its application to HDR imaging has special considerations. Here, the input images are not of equal exposure so the color constancy assumption of many algorithms is violated. Even if we map images to the same radiance space using the camera response curve [63, 88], they will have regions that are too dark/light and therefore invalid during alignment. This makes standard image registration techniques unsuitable for this application.

The simpler approaches to align the LDR sources solve for a transformation that accounts for camera motion between exposures. Ward [89] solved for a translation factor while Tomaszewska and Mantiuk [90] used SIFT feature points to compute a homography to align the images. Akyüz [91] used a simple correlation kernel assuming only translation. Yao [92] used phase cross-correlation to perform global motion estimation. These approaches all assume that the scene is rigid and on a plane, which is not the case for scenes such as the one in Fig. 1.3.

More sophisticated alignment methods are based on optical flow (OF) algorithms [93, 94]. Bogoni [95] used local unconstrained motion estimation using optical flow to warp the images into alignment. Kang et al. [96] significantly improved optical-flow approaches by introducing two key steps: a hierarchical homography to constrain the flow in regions where the reference was too light/dark to make it converge better, and an HDR merging process that rejects the aligned image wherever it is too far from the reference, similar to those used in deghosting approaches. Mangiat and Gibson [97] proposed a block-based bidirectional optical flow method using color information to find a better correspondences.

The current state-of-the-art method in LDR alignment for HDR applciations is the work

of Zimmer et al. [1]. They used an optical flow based method to minimize their proposed energy function consisting of a gradient term and a smoothness term to ensure smooth reconstruction of the regions where matching fails due to occlusion or saturation. Based on the displacement map obtained from previous stage and using another energy function, they reconstruct the HDR image, which has also been super-resolved.

In summary, however, the quality of the HDR images produced by all of these techniques is fundamentally limited by the accuracy of the alignment. Even the state-of-the-art optical flow are brittle in cases with complex motion and occlusions. For these reasons, many OF approaches use special HDR merging steps that reject misaligned images (as in deghosting) and cannot use standard merging techniques. Furthermore, optical flow cannot typically synthesize new content and thus cannot handle disocclusion when trying to align certain images (see, e.g., Fig. 4.5).

To address this problem, we were inspired by the recent success of patch-based optimization methods in related tasks like image editing [66] and view interpolation [67]. Our main observation is that instead of registering the LDR images as a preprocess (as is done by OF-based methods and which is a hard problem) and then merging them into an HDR image, we can do better if we solve for the HDR image and the aligned images simultaneously. This way, information from the HDR merging process will propagate across the images and help with the alignment. This will enable us to reconstruct a visually-plausible HDR image that looks locally like one of the sources but contains information from all of them. Therefore, our approach not only reconstructs the HDR image directly, but also computes "aligned" exposures as a by-product that can be merged with any standard technique.

4.4 Optimization for HDR reconstruction

Given a set of N LDR sources taken with different exposures and at different times $(L_1, \ldots, L_N)$, our goal is to reconstruct an HDR image H that is aligned to one of them

(the reference, called L_{ref}), but contains HDR information from all N exposures. To pose the problem as an energy minimization, we begin by asking the question: what are the desired properties of H?

If we "expose" our ideal H with function $l^{\text{ref}}(H)$ that maps the radiance values of H to the exposure range of the reference source (Eq. 4.8), we should get something that is very close to L_{ref}. This ensures that H looks like it was taken by a real camera and does not have unrealistic artifacts. Similarly, if we expose H with the parameters of the n^{th} input exposure to produce an LDR image $l^n(H)$, it should be "similar" to input source L_n. It may not be identical to L_n, since the movement between the sources means that H cannot match both L_{ref} and L_n exactly.

An appropriate measure of similarity might be bidirectional similarity (BDS) [66]: for every patch of pixels in $l^n(H)$ we should be able to find a comparable patch in L_n (coherence), and for every patch in L_n we should find a comparable patch in $l^n(H)$ (completeness). However, in some cases there might be content that should be visible at this exposure when aligned to reference L_{ref} but is occluded or missing in L_n, so applying BDS with a single source N times might not always work. Instead, we should use information from all the other exposures as well, because the missing content might be visible in one of these other images. This leads to a new multisource bidirectional similarity (or rather dissimilarity) measure for our application:

$$\begin{aligned}\text{MBDS}(T \mid \{S\}_1^N) = &\frac{1}{N}\sum_{n=1}^{N}\sum_{P\in S_n} w_n(P)\min_{Q\in T} d(P,Q) + \\ &\frac{1}{|T|}\sum_{Q\in T}\min_{P\in\{S\}_1^N} d(Q,P),\end{aligned} \tag{4.1}$$

where $|T|$ is the number of patches in T (the target image), P and Q are patches in S and T, respectively, and $d()$ is an L_2 distance metric. Here, the first term is the completeness, the second is the coherence. There are two main differences between Eq. 4.1 and standard BDS. The obvious difference is that MBDS takes multiple sources as input, so in the

completeness term we loop over all N sources and in the coherency term we find the best patch out of all N. A more subtle but important difference is the addition of the $w_k(P)$ term to weight the source patches when calculating completeness. The key idea is that not all source patches might be properly exposed (we use the term "valid" to say how well-exposed a patch is), so we should ignore saturated or under-exposed patches when computing completeness and give priority to well-exposed source patches when multiple sources map to the same target location. These weights are normalized to sum to 1.

We now need to apply the similarity measure to all N source images in our input stack. Therefore, we define an energy function such that each exposure n of the HDRI H is as similar as possible to all input sources adjusted to that exposure:

$$E_{\text{MBDS}}(H) = \sum_{n=1}^{N} \text{MBDS}\big(l^n(H) \mid \{g^n(L_k)\}_{k=1}^{N}\big), \tag{4.2}$$

where $g^n(L_k)$ is a function that maps the k^{th} LDR source to the n^{th} LDR exposure. It is computed as $g^n(L_k) = l^n(h(L_k))$ where $h(L_k)$ is a function that maps LDR source L_k to the appropriate range in the HDR linear radiance domain (Eq. 4.9). Note that the input to our MBDS function is the set of *all* N exposure-adjusted input sources [1]. Although Eq. 4.2 will produce images that are similar to the inputs, the resulting image might not be aligned to the reference. Therefore, we add a term that constrains H to match the HDR projection of the reference image wherever its pixels are well exposed:

$$E(H) = \sum_{p \in \text{pixels}} \big[(1 - \alpha_{\text{ref}_{(p)}}) \cdot E_{\text{MBDS}}(H) + \alpha_{\text{ref}_{(p)}} \cdot (h(L_{\text{ref}})_{(p)} - H_{(p)})^2\big]. \tag{4.3}$$

Here, $\alpha_{\text{ref}_{(p)}}$ is a simple trapezoid function that tells us which reference image pixels are valid. We could also have written the constraint as $(L_{\text{ref}} - l^n(H))^2$, but as we shall see later this form makes it more amenable for optimization.

[1] Experimentally, we found that using MBDS with only one source (as in standard BDS) worked in most of the cases we tested.

Eq. 4.3 is the key to this paper and we call it the *high-dynamic range image synthesis equation*. In plain English, the E_{MBDS} term of the HDRI synthesis equation states that for every patch in the final HDR image H at a given exposure there should be a similar patch in one of the LDR inputs after adjusting for exposure, which makes the final result H looks like a consistent image resembling the inputs. Likewise, every valid exposure-adjusted patch in all input images should be contained in H at this exposure, so that valid information from the inputs is preserved. The second term ensures that this addition of information happens only in the parts where the reference image L_{ref} is over/under-exposed, otherwise the result H should stick to L_{ref} as closely as possible.

Optimizing Eq. 4.3 is difficult because it requires us to solve for the HDRI H directly at all exposures. To minimize this equation, we approximate it by introducing an auxiliary variable I_n for $l^n(H)$. Intuitively, I_n is the LDR image that would be captured from the HDR image H if we "exposed" it with the settings of the n^{th} exposure. This substitution allows us to decouple one hard optimization into two easier optimizations, making the equation for E_{MBDS}:

$$\begin{aligned} E_{\text{MBDS}}(H, \{I\}_1^N) = &\sum_{n=1}^{N} \text{MBDS}\big(I_n \mid \{g^n(L_k)\}_{k=1}^N\big) \\ &+ \sum_{n=1}^{N} \sum_{p \in \text{pixels}} \Lambda(I_{n_{(p)}})(h(I_n)_{(p)} - H_{(p)})^2, \end{aligned} \tag{4.4}$$

where the second term has been added to keep I_n as close as possible to $l^n(H)$ in an L_2 sense (again, we have written it using $h(I_n)$ instead to clarify our optimization). We weight the comparison with a merging function $\Lambda()$ that tells us how the I_n's are weighted when combined to form H, because we want to give more/less importance to values of $I_{n_{(p)}}$ that contribute more/less to H as specified by the merging function. We see that if $I_n = l^n(H)$, then $h(l^n(H)) = H$ in the support of $\Lambda()$[2], and so the entire second term would be zero

[2]Because of the clipping process defined in Eq. 4.8, $l^n(H)$ is not invertible in general, but because the merging function $\Lambda()$ has the same clip bounds this statement is true.

everywhere. This means that when $I_n = l^n(H)$, then Eq. 4.4 will have the same energy as Eq. 4.2, validating our approximation. Plugging this in to our HDRI synthesis equation, our energy at every pixel p becomes:

$$
\begin{aligned}
E(H, \{I\}_1^N) = \\
&\sum_{p \in \text{pixels}} \Big[(1 - \alpha_{\text{ref}_{(p)}}) \sum_{n=1}^{N} \text{MBDS}\big(I_n \mid \{g^n(L_k)\}_{k=1}^N\big) \\
&+ (1 - \alpha_{\text{ref}_{(p)}}) \sum_{n=1}^{N} \Lambda(I_{n_{(p)}})(h(I_n)_{(p)} - H_{(p)})^2, \\
&+ \alpha_{\text{ref}_{(p)}} \cdot (h(L_{\text{ref}})_{(p)} - H_{(p)})^2 \Big]
\end{aligned} \tag{4.5}
$$

Eq. 4.5 suggests an iterative solution to solve for H and $\{I\}_1^N$ simultaneously, which forms the core of our algorithm for HDR image reconstruction (see Fig. 4.2). We first minimize for $\{I\}_1^N$ in the first two terms (which encourages the I_n's to look like, and contain information from, all the inputs) with a patch matching and voting process similar to Simakov et al. [66]. We then minimize for H in the bottom two terms (which constrain I_n to be part of H and that L_{ref} match the appropriate range of H), through a merging process that combines the reconstructed images I_n into an intermediate HDRI $\tilde{H}$ and blends in the radiance-adjusted reference $g(L_{\text{ref}})$.

We now discuss these two key stages of our algorithm. The first stage of the core algorithm (Sec. 4.6.2) uses a matching and voting process to reconstruct intermediate LDR images $I_1 \ldots I_n$ that are similar to exposure-adjusted versions of the sources. We also blend in $l^n(H)$ using the H from the previous iteration in order to encourage the solution to be close to the exposed value from H. This stage jointly minimizes the MBDS and the second term of Eq. 4.5 correspondingly. The second stage will optimize for the H variable of Eq. 4.5 by merging all the I_n images together to form the intermediate HDR result H, as well as ensure that H is close to $h(L_{\text{ref}})$ in an $\mathcal{L}_2$ sense over the valid range of L_{ref}. To handle this last part, we always inject the input reference directly into the merging process with the

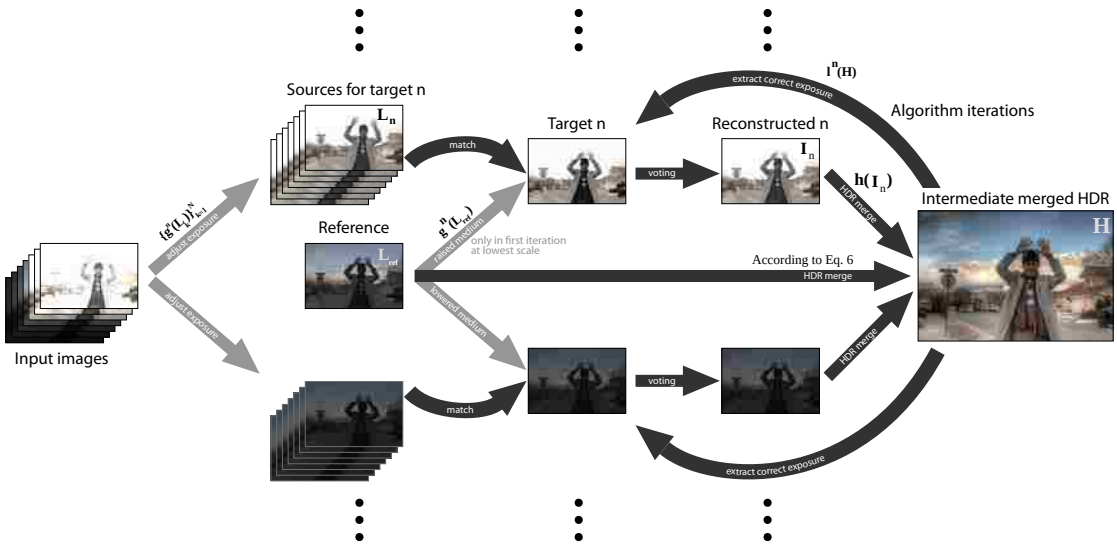


Figure 4.2: *This figure shows the inner core of the algorithm that runs at a single scale to find a solution to the HDRI synthesis equation. We show three exposure levels here, although our algorithm runs on all N exposures. This process is repeated at multiple scales.*

appropriate alpha blending weights from Eq. 4.5. Therefore, a pixel in our intermediate H can be computed as follows:

$$H_{(p)} \leftarrow (1 - \alpha_{\text{ref}_{(p)}}) \cdot \tilde{H}_{(p)} + \alpha_{\text{ref}_{(p)}} \cdot h(L_{\text{ref}})_{(p)}, \tag{4.6}$$

where $\tilde{H}$ is an HDR image computed with the standard merging of all N images $\{I\}_1^N$:

$$\tilde{H}_{(p)} \leftarrow \frac{\sum_{n=1}^{N} \Lambda(I_{n_{(p)}}) h(I_n)_{(p)}}{\sum_{n=1}^{N} \Lambda(I_{n_{(p)}})}. \tag{4.7}$$

In our implementation, we use the triangle weighting function defined by Debevec and Malik [63] (Eq. 4 of that paper) for $\Lambda()$. Note that the first and second terms of Eq. 4.6 minimize the second and third terms of Eq. 4.5, respectively. We perform these two stages at every iteration, and as is common for patch-based algorithms like this (e.g., Simakov et al. [66]), this core algorithm is then performed at multiple scales, starting with the coarsest

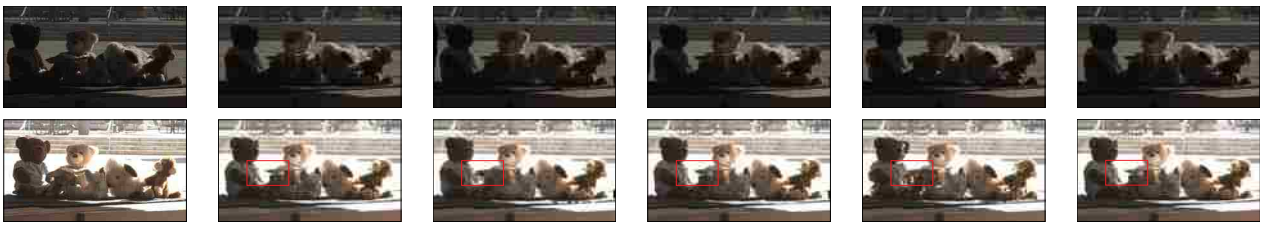


Figure 4.3: *To test the accuracy of our reconstructed images, we compare our aligned reconstructions of the low/high images in Fig. 4.4 to the actual ground truth images taken. On the left we have the input low/high images (one per row), followed by the corresponding ground truth image taken at the middle position. The next three results show the output of optical flow algorithms when matching to the lowered/raised medium image, and then we show the output of our approach. We see that our result matches the ground truth images more accurately.*

resolution and working to the finest (Sec. 4.6.4). After the algorithm has converged, we have solved for both the desired HDRI H and as well as the "aligned" images at each exposure $\{I\}_1^N$.

4.5 Results

To test the quality of our reconstructed images, we compare against several state-of-the-art approaches for HDR image alignment and deghosting. We compare our results to four optical flow (OF) algorithms: (1) the motion detail preserving optical flow (MDP OF) algorithm of Xu et al. [98], (2) the large displacement optical flow (LD OF) of Brox and Malik [99], (3) the optical flow implementation of Liu (Liu OF) [100] based on the work of Brox et al. [101] and Bruhn et al. [102] to enable them to handle large motion, and (4) the algorithm of Zimmer et al. [1], which is perhaps the state-of-the-art in preprocess alignment methods.

For the first three OF methods, we used the hierarchical homography proposed by Kang et al. [96] to constrain the flow in the regions where the reference image was unreliable,

(a) Low input **(b)** Middle input **(c)** High input

(d) Ground truth HDRi **(e)** Our HDRi **(f)** HDRi without deghosting

(g) MDP OF HDRi **(h)** Heo deghosting HDRi **(i)** Zhang deghosting HDRi

Figure 4.4: *In this test, we captured* **(a)** *low,* **(b)** *medium, and* **(c)** *high exposures of a test scene while moving the toys between frames to simulate motion. We also took pictures of the medium pose at low/high exposure to produce the* **(d)** *ground truth result.* **(e)** *Our tonemapped HDR matches the ground truth fairly closely.* **(f)** *HDR image produced when merging original images without deghosting in Photomatix, which shows the amount of motion in the scene.* **(g-h)** *HDR images produced by some competing approaches.*

but it only improved the results of a few scenes. Often these methods did equally well (or sometimes better) without it (we show the best results obtained either way). We also used Kang et al.'s merging approach, which improved the quality of the OF results considerably by filtering out misalignments. Therefore, we can consider the results presented here with these OF methods to be at least comparable to that of Kang et al., although they used a variant of the Lucas and Kanade [93] OF with a Laplacian pyramid. Note that our results are shown as the result of a standard HDR merge without the need to handle misalignment artifacts.

We also compare our algorithm with current deghosting methods: Gallo et al.'s block based deghosting [75], Pece and Kautz's bitmap movement detection [78], Heo et al.'s weighting method based on joing probability density functions [84], and Zhang and Cham's gradient-directed exposure composition [80, 81]. Finally, we also compare our results against the commercial software packages Photomatix and Photoshop's Merge to HDR Pro tool.

We begin with results for experimental scenes to validate our approach. The first scene is a static scene (taken on a tripod) where the objects were moved between frames to simulate motion. With the objects in the middle position, we captured low/high exposure frames to have a ground-truth comparison. We compare the quality of the aligned reconstructions in Fig. 4.3 and that of the HDR images produced by the different methods in Fig. 4.4. We see that our algorithm produces results closer to the ground truth image. In terms of MSE, our aligned reconstructions were one to two orders of magnitude better than the OF approaches.

The next test scene, Fig. 4.5, demonstrates the ability of our algorithm to fill in a visibility hole with complex information, which is difficult for OF algorithms (note the large artifacts, even after Kang et al.'s plausibility map rejects misalignments). Deghosting methods also fail for this scene, since motion is in an HDR region and the algorithm has to choose which image to draw the radiance values from. In this case, it draws from the reference image (the high exposure), but the pixels are saturated which causes the radiance to be clamped in this region, producing a dark halo when tonemapped. Our algorithm, on the other hand, is able to reconstruct the detail in the occluded region using the information from neighboring patches that are visible, since our HDRi synthesis equation produces a final image that has content that exists somewhere in all input images.

Finally, we show the results of our algorithm on natural scenes in Figs. 4.6 – 4.8. Our algorithm worked robustly in every scene we tested and outperformed previous approaches. In particular, we point out the comparisons with Zimmer et al. [1], which is the state-of-the-art approach for HDR image alignment in Figs. 4.13 and 4.14. For the first figure,

Figure 4.5: *Our patch-based optimization can hole-fill information when visibility inconsistencies occur, which is not possible by any of the previous approaches. In this example, we have two input images (high and low, separated by 4 stops), and we are registering to the high exposure. However, the desired detail in the background of the low image is occluded by the subject, so the algorithm must reconstruct this missing information when aligning the images. Clearly optical flow methods and deghosting methods cannot handle this situation. Our algorithm, on the other hand, uses the information surrounding the hole to fill it in in a plausible manner.*

we provided Zimmer with our images to run them with their algorithm with their optimal parameters. We can clearly see that they are unable to align the source to the reference when undergoing such complex motion, while ours produces a very good alignment and therefore subsequent HDR result. In Fig. 4.14 we use the failure case in Zimmer. We refer readers to additional images in the supplementary material uploaded with our submission.

4.6 Implementation

We implemented our HDR image alignment algorithm in MATLAB which was sufficient for our purposes. Although we plan to release our implementation and data sets when the paper is published, this section will provide some of the necessary implementation details to reproduce our results.

4.6.1 Image pre-processing

If the sources are in JPEG or some other non-linear format, we first convert them into a linear space (range 0 to 1) using the appropriate camera response curve which is assumed

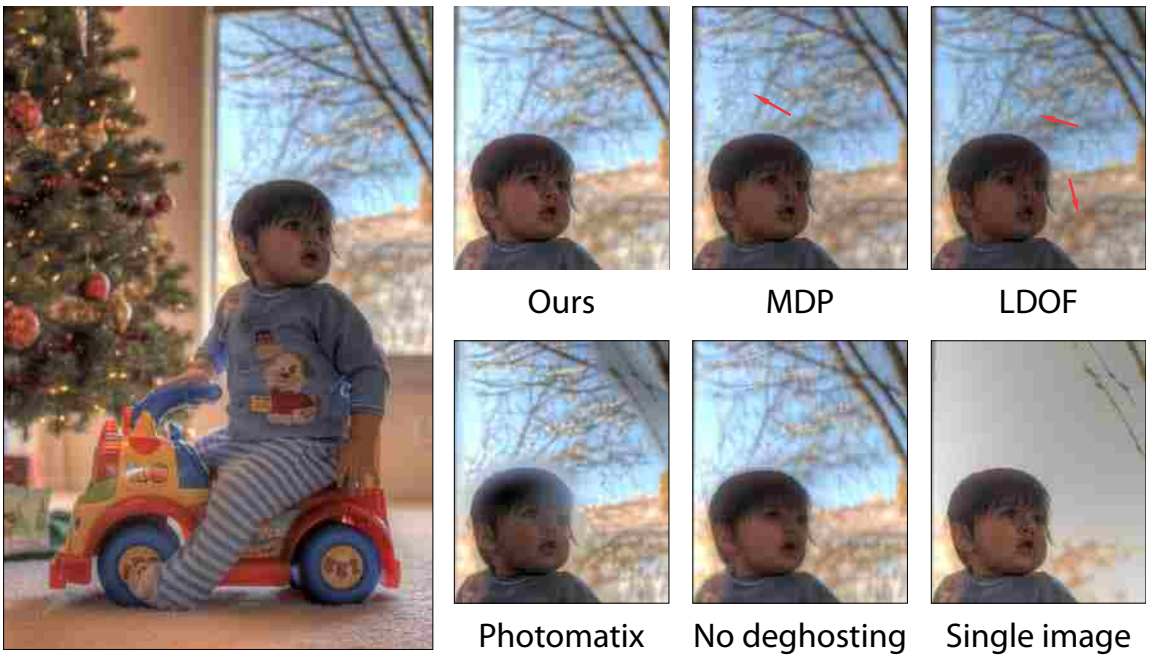


Figure 4.6: *Optical flow methods have problems maintaining the continuity of the content outside the window in this scene, while Photomatix's ghost removal algorithm appears to use only one exposure in the regions with motion, which results in a saturated halo around the subject's head and on the tree branches outside. Our method produces good results.*

to be known [63]. We then apply a gamma curve with $\gamma = 2.2$ to the linear raw data to get the input sources $L_{1...N}$ for our algorithm. We do this because we compute differences between patches during the matching process, and doing this in a linear space does not adequately reflect the way people see differences perceptually. We found that by performing the MBDS process in the gamma domain, the final reconstructions look better in the dark parts of the image. All operations are in floating point and we define the range of the reference exposure to be of unit radiance.

4.6.2 Reconstructing the intermediate images

In the reconstruction stage and through out our algorithm, we use the following functions to map the radiance domain of H into one of the exposures and vice-versa:

$$l^n(H) = \text{clip}\big((H/\text{exposure}(n))^{\frac{1}{\gamma}}\big), \tag{4.8}$$

$$h(L_k) = L_k^{\gamma} \times \text{exposure}(k), \tag{4.9}$$

where exposure(n) tells us the exposure ratio between the n^{th} exposure and the reference since we assume that the reference exposure has unit radiance in the HDR domain.

To begin our matching process, we need an initial guess for the I_n's in the first iteration. To do this, we simply exposure-correct the reference image to come up with the target for the next higher exposure: $I_{\text{ref}+1} \leftarrow g^{\text{ref}+1}(L_{\text{ref}})$. We continue to do this sequentially for the higher exposures using $I_{\text{n}+1} \leftarrow g^{n+1}(I_n)$ after each I_n has gone through one iteration, and do something similar for the lower exposures. Note, however, that the initial target of the optimization does not affect the final result much, since this only impacts the first iteration at the coarsest scale. Both stages of our algorithm ensure that after the first iteration, information from all sources is propagated to all other exposure levels.

To implement the MBDS metric in a simple way, we used the publicly-available implementation of Barnes et al. [103] for the search/voting portion of the first stage (accelerated by the PatchMatch algorithm), with modifications to handle multiple sources for MBDS. For each target exposure level n, we ran a dense search step a repeated number of times on all adjusted source exposures $g^n(L_k)$ using the current image at that level I_n as the MBDS target input. The bidirectional search produced two nearest neighbor fields (NNF) for each k: one for coherence and one for completeness. Note that the completeness search is masked, which means that we only search in the valid parts of each source $g^n(L_k)$. This effectively implements the $w_k(P)$ term in Eq. 4.1 with a hard mask. For every pixel in the final coherence NNF, we choose the one in the stack of NNFs that results in the smallest $\mathcal{L}_2$ distance. This handles the $\min$ term over all the sources in Eq. 4.1. This results in N NNF's for the complentess term and one NNF (with an additional component to identify the source) for the coherence term for every exposure level n.

For voting, we sum the patches for the coherence NNF in the standard way [66] using the patches from the appropriate exposure at each pixel. For the completeness NNF's, on the other hand, we use each NNF to sum the respective patches from each adjusted exposure and then averaged together. The final result can then be generated by summing these two terms together and then dividing by the appropriate weight. This gives us our new I_n. This process is repeated for all N sources.

4.6.3 Merging

In order to accelerate the convergence of our algorithm during the merging process, we should avoid blending in pixels from the other sources with the reference exposure in Eq. 4.6 if they have been clearly misaligned. To implement a simple consistency check, we split up the calculation of $\tilde{H}$ in Eq. 4.7 into two parts: one that merges the images that are lower than the reference $\tilde{H}^-$ (by computing Eq. 4.7 from $n = 1$ to ref $- 1$) and the other that will merge the images that are higher than the reference $\tilde{H}^+$ (by computing Eq. 4.7 from $n =$ ref $+ 1$ to N). We then approximate Eq. 4.6 as:

$$H_{(p)} \leftarrow (1 - \alpha_{\text{ref}_{(p)}})(\alpha^+_{(p)}\tilde{H}^+_{(p)} + \alpha^-_{(p)}\tilde{H}^-_{(p)}) + \alpha_{\text{ref}_{(p)}} \cdot g(L_{\text{ref}})_{(p)}, \tag{4.10}$$

In our implementation we used values of 0.1 and 0.9 for the minimum and maximum valid values v_{min} and v_{max}. We can understand this equation better if we realize that at the finest scale α^+ and α^- cannot both be 1 at the same time. The α^+ term focuses on the lower values of the reference (where the higher exposures will provide detail), while α^- focuses on the higher values (where the lower exposures will do this). Because of the triangle functions Λ used to weight the exposures, the exposures lower than the reference would not contribute much to the region covered by the α^+ and vice-versa. So $(1 - \alpha_{\text{ref}_{(p)}})(\alpha^+_{(p)}\tilde{H}^+_{(p)} + \alpha^-_{(p)}\tilde{H}^-_{(p)}) \approx (1 - \alpha_{\text{ref}_{(p)}})\tilde{H}$.

This separation of $\tilde{H}$ into two terms now allows us to do a simple consistency check. In parts of the image where the reference is under-exposed ($L_{\text{ref}(p)} < v_{\text{min}}$), we only blend

values of $\tilde{H}^+$ with Eq. 4.10 if $l_{\text{ref}}(\tilde{H}^+) < v_{\text{min}}$. Likewise, wherever the reference is over-saturated ($L_{\text{ref}(p)} > v_{\text{max}}$), we only blend values of $\tilde{H}^-$ if $l_{\text{ref}}(\tilde{H}^-) > v_{\text{max}}$.

Unlike many optical flow-based algorithms, after our algorithm has converged, the aligned images $\{I\}_1^N$ do not require any consistency check and we can use any standard merge. Furthermore, unlike deghosting algorithms where consistency checks are used in one pass to cull information, ours is used as part of our optimization to help the convergence. Removing this check produces comparable images with similar HDR content.

We conclude the second stage by merging the images to form intermediate HDR image H. We then apply $l^n(H)$ and extract the correct exposures to create targets for the first stage in our next iteration. These are then used by the matching/voting step of the algorithm, along with the NNF's from the previous iteration as described in Sec. 4.6.2.

4.6.4 Extending our algorithm for multiple scales

Our optimization is a multiscale algorithm that performs the iterations shown in Fig. 4.2 over multiple scales (see, e.g., [66]). In other words, first we match the global structure in the coarse scales and then match local detail in the high scales. As a preprocess, we generate an image pyramid for each input source by downsampling them using a Lanczos filter in order to accelerate the algorithm. After we complete the set of EM iterations for Fig. 4.2, we move to the next scale. In our implementation, the lowest-resolution scale has 35 pixels in the smaller dimension and we perform a total of 10 scales, so we must upsample the images by a ratio of $\sqrt[9]{x/35}$ in each dimension (x is the minimum dimension of the final image) when moving up a scale. We also adjust the number of EM iterations at each scale, starting with 50 at the lowest scale and linearly decreasing this to 5 iterations at the finest scale.

When a scale is completely converged, we do not perform the regular merging step. Rather, the final reconstructed low/high images is upsampled up to the next scale using a Lanczos

filter. These upsampled images are then merged with the reference image from the input image pyramid using the same merging algorithm described above. This process allows to inject the extra detail that is now available in the new, higher-resolution reference image into our EM iteration process. We also upscale all of the NNF's computed in the previous iteration, and proceed with the next scale's iterations.

4.6.5 Acceleration and other details

To accelerate our algorithm, we implemented several optimizations. First, we only perform our coherency search on the target where the corresponding patches of the reference image have pixels that have $\alpha_{\text{ref}_{(p)}} \neq 1$, because these regions will be directly using values from the L_{ref} source. We also experimented with sub-pixel search in the PatchMatch algorithm but disabled it because it did not significantly impact the results and it was expensive. We also perform the completeness search only in the first half set of scales in our multiscale approach. At this point, our algorithm has added the missing information from other images so from then on we only do coherency.

We also experimented with varying the number of sources $g^n(L_k)$ available to the MBDS algorithm instead of using all N. We found that in 90% of the cases we tested, we were able to get good results with only using one source (the one that matched that particular exposure, $g^n(L_n)$). Therefore, we did this for all the results in the paper for acceleration. However, we did find some cases where this made a difference (see Fig. 4.11).

Finally, in some of the data sets the camera was changing both the aperture and the shutter speed to take the different exposures, which was causing visibly different defocus blur for background objects in the different sources. If we simply used the HDRI synthesis equation of Eq. 4.3 with all N sources at the same time, we noticed that the algorithm would use information with different defocus blur to fill in the HDR information in different parts of the image in a seamless way. However, this change of focus can be noticeable, as shown in Fig. 4.9. We found that by restricting Eq. 4.3 to operate only the immediate

images around the reference exposure, and then pairwise working outward, we would get smoother results overall. This is due to the fact that when matching only a few sources near the reference, the algorithm is able to match the blur pretty reasonably. In the subsequent iterations, we then would match that blur with the next exposure and so on.

4.7 Discussion

Typically, photographers taking a bracketed set of exposures for HDR imaging only change the shutter time between exposures to maintain the same depth-of-field in each image and facilitate alignment/merging. Because our algorithm automatically aligns the reconstructed images while solving for the HDR result, we can produce good results when the aperture changes between exposures as well. Fig. 4.10 shows how our algorithm "sharpens" an input image to match the depth of field of the reference.

This capability gives photographers one more dimension to adjust their apertures for bracketed photography. For example, to take 10 stops of additional dynamic range with bracketing of the shutter time alone, the longest exposure should be $1024\times$ longer than the shortest. This becomes impractical in most situations, especially if the camera is handheld. Our approach gives photographers the flexibility to modify the aperture as well as the shutter time when taking bracketed exposures, thereby allowing them to capture HDR images of scenes that could otherwise not be captured.

However, our approach to HDR imaging has limitations. Unlike specialized HDR cameras that capture all exposures simultaneously, our algorithm cannot reconstruct the HDR content if it moves too much and becomes occluded when we are capturing the correct exposure. This causes these regions to contain only LDR when reconstructed, since our algorithm tries to match the reference image but does not have information from the other exposures to draw from. An example of this is shown in Fig. 4.12, where some of the people only appear in a single frame.

One advantage of our technique over HDR camera hardware is that we can adjust the exposure separation between images based on scene content. Different scenes have different dynamic ranges, and this flexibility ensures that we are "sampling" the dynamic range efficiently. HDR camera hardware cannot typically do this because the separation between exposures is fixed by the hardware.

We hope that this algorithm takes a step towards making high-quality HDR imaging more available to the general public. In the future, it is possible that camera manufacturers provide firmware to automatically take a series of bracketed set exposures for every scene to produce images like those we show in this paper.

(a) Ours **(b)** w/o deghosting **(c)** MDP OF

(d) LD OF **(e)** Ce Liu OF **(f)** Single image tonemapped

Figure 4.7: *This scene has a lot of movement which makes it difficult for OF algorithms. Of all competing approaches, our algorithm matches the color quality of the ghosted HDRi image the best, but without motion artifacts.*

(a) Ours **(b)** MDP OF **(c)** LDOF

(d) Liu OF **(e)** w/o deghosting **(f)** Photoshop

Figure 4.8: *Our algorithm is able to faithfully reconstruct this complex scene. The optical flow methods, however, have artifacts, e.g., in the reflection of the hands on the piano.*

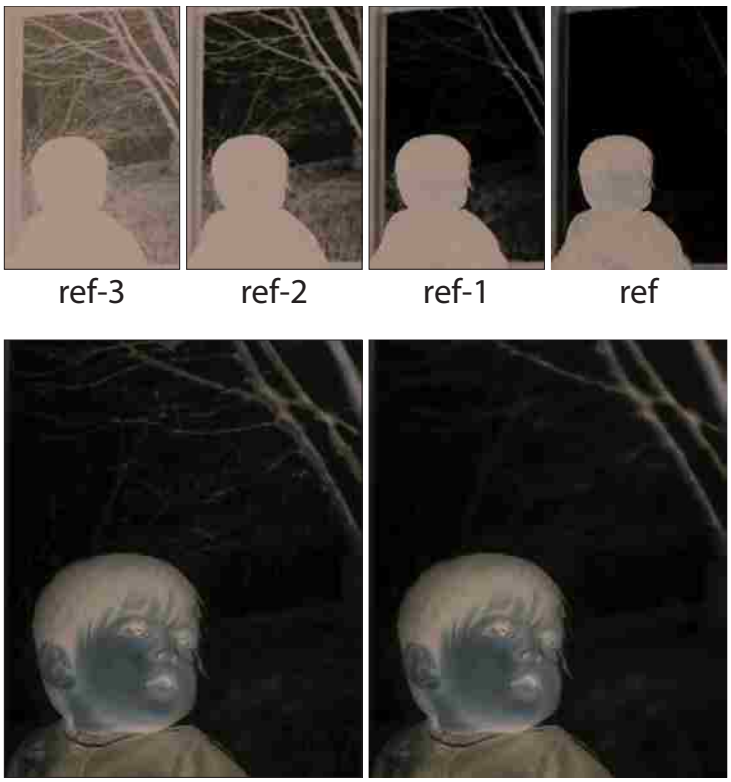


Figure 4.9: *Here, we compare between using all sources simultaneously (left) and just matching to the nearest exposures as explained in Section 4.6.5 (right). The input images lower than the reference are shown in the top row. In each input the defocus blur of the branches in the background is clearly different. By using all the sources at the same time, the algorithm puts together information with different defocus blur to fill in the HDR information in a seamless way. Although the resulting image is plausible, the approach where we use only the nearest exposures iteratively produces a more pleasing result in this case. We note that this only impacts images where the aperture changes considerably between exposures.*

Figure 4.10: *This figure shows how our algorithm can sharpen an image to match the the depth of field of the reference. For this scene (our HDR result shown on the left), we captured 10 stops of bracketed exposure by changing both the aperture as well as shutter time. This was the only way to take this picture since the camera was hand-held. On the right we show one of the original input frames, as well as our reconstruction. We see that the out-of-focus region on the bench has been made sharper to match the reference.*

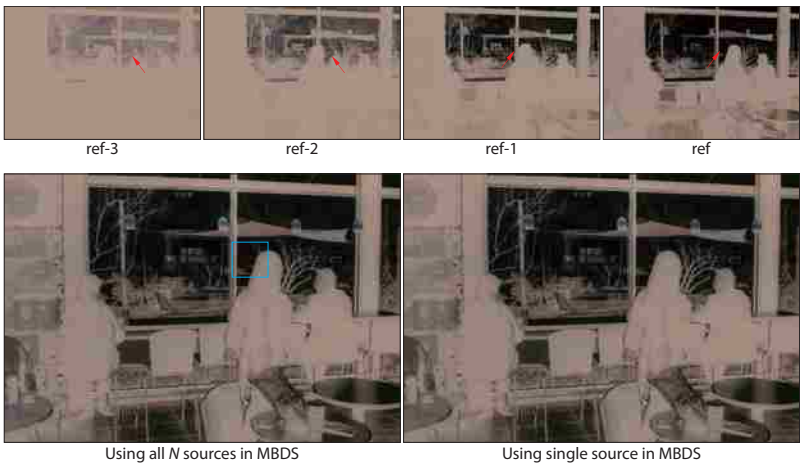


Figure 4.11: *For this complex scene, we compare the results using all the N sources $g^n(L_k)$ in the MBDS function (left) and using only the source at that exposure (right). The top row shows the input images L_1 to L_{ref}. The arrow on the reference indicates a region that is saturated but is also occluded in the L_{ref-1} image. Therefore, if we only one source in the MBDS function, we do not have access to the correct, well-exposed information and therefore we get an incorrect result as can be seen in the image in the lower right. By using all N sources simultaneously, we have access to the L_{ref-2} and L_{ref-3} which provide the missing information to get a high quality HDR result.*

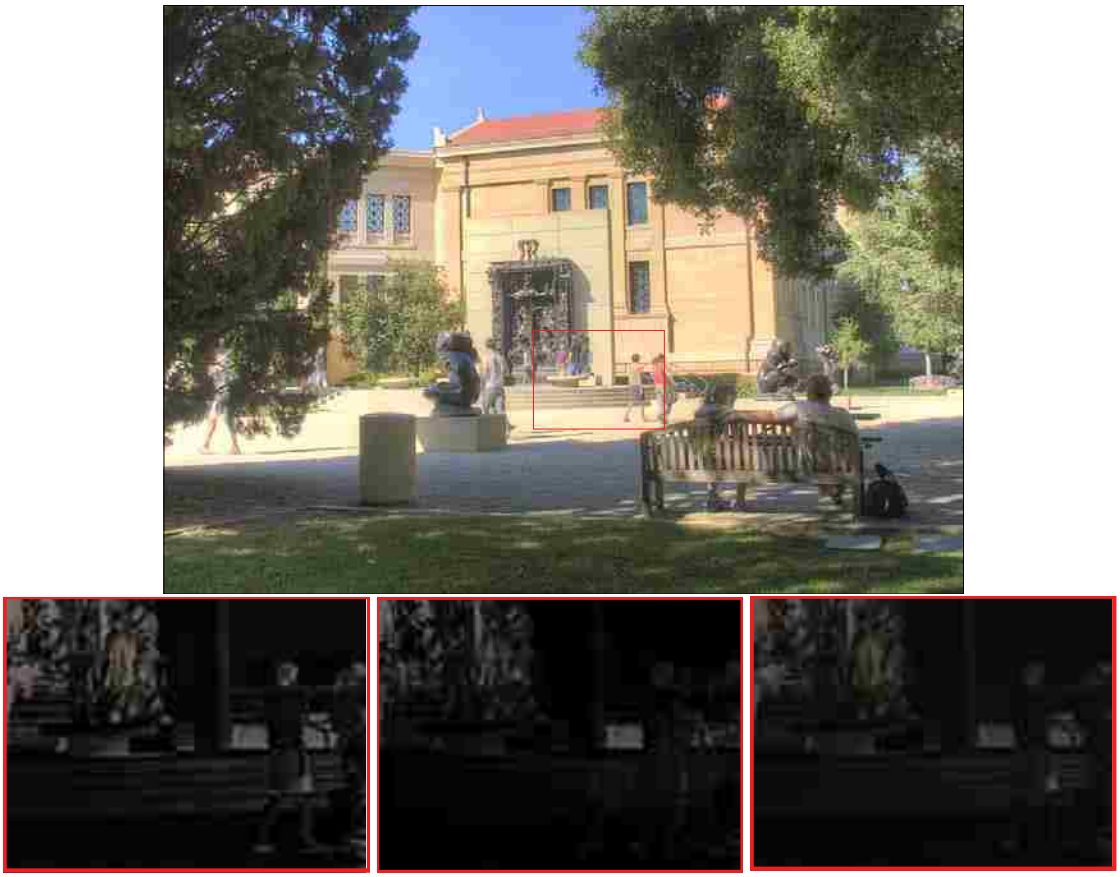


Figure 4.12: *This scene (from Gallo et al. [2009]) has moving people that are different in every frame. We show the results of the deghosting methods of Gallo et al. (left) and Pece and Kautz [2010] (middle) using images provided by the authors. The former has visible block artifacts because of the way they detect motion in a per-block basis, and the latter leaves much of the ghosting. Our method (top and right) can remarkably reconstruct most of the moving people, but it has artifacts as well. These appear as "washed out" regions where our algorithm only had information from one LDR image because the people in the reference disappeared.*

Figure 4.13: *Here we compare the reconstruction and HDRI results of our method with Zimmer et al. [1] method. We gave the images to the authors and they ran their code on them. Zimmer et al. method is not able to reconstruct the moving objects (e.g. the man and reflection of him on the piano) which appears as ghosting in the final HDR image. Our method, however, can produce high quality results.*

Figure 4.14: *This image shows the comparison of our results with Zimmer et al. method on their failure case. Our method can reconstruct the people and cars well, but Zimmer et al. method cannot handle these regions because of the large motion. Furthermore, our method is able to bring more HDR information which can be seen by comparing the details on the clouds.*

Chapter 5

Conclusions

We showed a general patch-based synthesis framework that handles inconsistencies within and across image sources. It combines principles from patch-based synthesis with gradient domain blending and texture interpolation into a unified powerful synthesis engine. We also show that the different components work in harmony and complement each other. For example, when using only translations, the use of the L_2 norm on gradients might lead to blurry results due to lack of accurate matches. However by allowing geometric and appearance deformations this problem goes away - L_2 on gradients works well and results in a much simpler and faster optimization. We originally designed the method to handle multiple sources with substantial inconsistencies for challenging stitching, cloning and morphing problems, however it was found extremely useful also for single source task such as image completion and warping.

We have also presented a novel framework for HDR reconstruction based on a new energy-minization equation called HDR image synthesis equation that crystalizes the objective of many HDR imaging approaches: to produce an HDR image that coherently uses all the content in the input exposures but is properly matched to one of them. We have shown that this approach is more robust than previous work in cases where the motion is complex, such as when a moving object is reflected of a surface, and can handle a wide range of

natural images succesfully.

In summary, the contributions of this thesis include:

- Introducing a general patch-based synthesis framework that can handle inconsistent sources in color, texture, local orientations and scale.
- Combining patch-based and gradient domain techniques in a unified optimization framework.
- A new patch-based blending method which can be used to spatially and/or temporally interpolate textures and general images.
- Introducing a novel patch-based energy-minimization formulation that integrates alignment and reconstruction in a joint optimization through an equation we call the HDR image synthesis equation.
- Extending the operating range of many existing image editing techniques through our general framework: same-source hole filling, multi-source hole filling, texture interpolation, stitching, image cloning, image warping, and automatic morphing, HDR reconstruction.

References

[1] H. Zimmer, A. Bruhn, and J. Weickert, "Freehand HDR imaging of moving scenes with simultaneous resolution enhancement," *Computer Graphics Forum*, vol. 30, no. 2, pp. 405–414, 2011.

[2] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick, "Graphcut textures: image and video synthesis using graph cuts," in *SIGGRAPH*, SIGGRAPH, (New York, NY, USA), pp. 277–286, ACM, 2003.

[3] Y. Wexler, E. Shechtman, and M. Irani, "Space-time completion of video," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, pp. 463 –476, march 2007.

[4] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "PatchMatch: A randomized correspondence algorithm for structural image editing," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 28, Aug. 2009.

[5] C. Barnes, *PatchMatch: A Fast Randomized Matching Algorithm with Application to Image and Video*. PhD thesis, Princeton University, USA, 2011.

[6] P. J. Burt and E. H. Adelson, "A multiresolution spline with application to image mosaics," *ACM Trans. Graph.*, vol. 2, pp. 217–236, October 1983.

[7] S. Paris, S. W. Hasinoff, and J. Kautz, "Local laplacian filters: edge-aware image processing with a laplacian pyramid," in *SIGGRAPH*, SIGGRAPH, (New York, NY, USA), pp. 68:1–68:12, ACM, 2011.

[8] K. Sunkavalli, M. K. Johnson, W. Matusik, and H. Pfister, "Multi-scale image harmonization," in *SIGGRAPH*, SIGGRAPH, (New York, NY, USA), pp. 125:1–125:10, ACM, 2010.

[9] H.-Y. Wu, M. Rubinstein, E. Shih, J. Guttag, F. Durand, and W. T. Freeman, "Eulerian video magnification for revealing subtle changes in the world," *ACM Trans. Graph. (Proceedings SIGGRAPH 2012)*, vol. 31, no. 4, 2012.

[10] P. Pérez, M. Gangnet, and A. Blake, "Poisson image editing," in *SIGGRAPH*, SIGGRAPH, (New York, NY, USA), pp. 313–318, ACM, 2003.

[11] A. Levin, A. Zomet, S. Peleg, and Y. Weiss, "Seamless image stitching in the gradient domain," in *ICCV*, 2006.

[12] A. Agarwala, "Efficient gradient-domain compositing using quadtrees," in *SIGGRAPH*, SIGGRAPH, (New York, NY, USA), ACM, 2007.

[13] Z. Farbman, G. Hoffer, Y. Lipman, D. Cohen-Or, and D. Lischinski, "Coordinates for instant image cloning," in *SIGGRAPH*, SIGGRAPH, (New York, NY, USA), pp. 67:1–67:9, ACM, 2009.

[14] M. W. Tao, M. K. Johnson, and S. Paris, "Error-tolerant image compositing," in *ECCV*, 2010.

[15] Z. Farbman, R. Fattal, and D. Lischinski, "Convolution pyramids," in *SIGGRAPHASIA*, SA '11, (New York, NY, USA), pp. 175:1–175:8, ACM, 2011.

[16] J. McCann and N. S. Pollard, "Real-time gradient-domain painting," in *ACM SIGGRAPH 2008 papers*, SIGGRAPH '08, (New York, NY, USA), pp. 93:1–93:7, ACM, 2008.

[17] A. Levin, "Blind motion deblurring using image statistics," in *In Advances in Neural Information Processing Systems (NIPS)*, 2006.

[18] P. Bhat, C. L. Zitnick, M. Cohen, and B. Curless, "Gradientshop: A gradient-domain optimization framework for image and video filtering," *ACMTOG*, vol. 29, pp. 10:1–10:14, April 2010.

[19] L. Xu, C. Lu, Y. Xu, and J. Jia, "Image smoothing via l0 gradient minimization," *ACM Transactions on Graphics (SIGGRAPH Asia)*, 2011.

[20] Y. HaCohen, E. Shechtman, D. B. Goldman, and D. Lischinski, "Non-rigid dense correspondence with applications for image enhancement," in *SIGGRAPH*, SIGGRAPH, (New York, NY, USA), pp. 70:1–70:10, ACM, 2011.

[21] S. Lefebvre and H. Hoppe, "Appearance-space texture synthesis," in *ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, (New York, NY, USA), pp. 541–548, ACM, 2006.

[22] D. Greig, B. Porteous, and A. Seheult, "Exact maximum a posteriori estimation for binary images," *Royal Journal on Statistical Society*, vol. 51, 1989.

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

[24] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin, and M. Cohen, "Interactive digital photomontage," *ACM SIGGRAPH*, vol. 23, no. 3, pp. 294–302, 2004.

[25] B. Kaneva, J. Sivic, A. Torralba, S. Avidan, and W. T. Freeman, "Infinite images: Creating and exploring a large photorealistic virtual space," in *Proceedings of the IEEE*, 2010.

[26] Y. Pritch, E. Kav-Venaki, and S. Peleg, "Shift-map image editing," in *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 151 –158, 29 2009-oct. 2 2009.

[27] R. Gal, Y. Wexler, E. Ofek, H. Hoppe, and D. Cohen-Or, "Seamless montage for texturing models," *Comput. Graph. Forum*, pp. 479–486, 2010.

[28] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," *ICCV*, vol. 2, p. 1033, 1999.

[29] L. Y. Wei and M. Levoy, "Fast texture synthesis using tree-structured vector quantization," in *ACM SIGGRAPH*, pp. 479–488, 2000.

[30] A. Criminisi, P. Pérez, and K. Toyama, "Object removal by exemplar-based inpainting," *CVPR*, vol. 2, p. 721, 2003.

[31] I. Drori, D. Cohen-or, and H. Yeshurun, "Fragment-based image completion," *ACM SIGGRAPH*, vol. 22, pp. 303–312, 2003.

[32] Y. Wexler, E. Shechtman, and M. Irani, "Space-time video completion," *CVPR*, vol. 1, pp. 120–127, 2004.

[33] V. Kwatra, I. Essa, A. Bobick, and N. Kwatra, "Texture optimization for example-based synthesis," *ACM SIGGRAPH*, vol. 24, 2005.

[34] D. Simakov, Y. Caspi, E. Shechtman, and M. Irani, "Summarizing visual data using bidirectional similarity," in *CVPR*, (Anchorage, AK, USA), 2008.

[35] L.-Y. Wei, J. Han, K. Zhou, H. Bao, B. Guo, and H.-Y. Shum, "Inverse texture synthesis," in *ACM SIGGRAPH 2008 papers*, SIGGRAPH '08, (New York, NY, USA), pp. 52:1–52:9, ACM, 2008.

[36] E. Shechtman, A. Rav-Acha, M. Irani, and S. Seitz, "Regenerative morphing," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 615 –622, june 2010.

[37] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein, "The generalized PatchMatch correspondence algorithm," in *European Conference on Computer Vision*, Sept. 2010.

[38] A. Mansfield, M. Prasad, C. Rother, T. Sharp, P. Kohli, and L. Van Gool, "Transforming image completion," in *Proc. BMVC.*, 2011.

[39] N. Kawai, T. Sato, and N. Yokoya, "Image inpainting considering brightness change and spatial locality of textures and its evaluation," in *Proceedings of the 3rd Pacific Rim Symposium on Advances in Image and Video Technology*, PSIVT '09, (Berlin, Heidelberg), pp. 271–282, Springer-Verlag, 2008.

[40] P. Arias, G. Facciolo, V. Caselles, and G. Sapiro, "A variational framework for exemplar-based image inpainting," *IJCV*, vol. 93, pp. 319–347, July 2011.

[41] A. Bugeau, M. Bertalmi ando, V. Caselles, and G. Sapiro, "A comprehensive framework for image inpainting," *Image Processing, IEEE Transactions on*, vol. 19, pp. 2634 –2645, oct. 2010.

[42] J. Jia and C.-K. Tang, "Eliminating structure and intensity misalignment in image stitching," in *ICCV*, vol. 2, pp. 1651 –1658 Vol. 2, oct. 2005.

[43] W.-Y. Lin, S. Liu, Y. Matsushita, T.-T. Ng, and L.-F. Cheong, "Smoothly varying affine stitching," in *CVPR*, pp. 345 –352, june 2011.

[44] J. Hays and A. A. Efros, "Scene completion using millions of photographs," in *SIGGRAPH*, SIGGRAPH, (New York, NY, USA), ACM, 2007.

[45] J.-F. Lalonde, D. Hoiem, A. A. Efros, C. Rother, J. Winn, and A. Criminisi, "Photo clip art," in *ACM SIGGRAPH 2007 papers*, SIGGRAPH, (New York, NY, USA), ACM, 2007.

[46] R. Szeliski and H.-Y. Shum, "Creating full view panoramic image mosaics and environment maps," in *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '97, (New York, NY, USA), pp. 251–258, ACM Press/Addison-Wesley Publishing Co., 1997.

[47] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, SIGGRAPH, (New York, NY, USA), pp. 417–424, ACM Press/Addison-Wesley Publishing Co., 2000.

[48] W. Matusik, M. Zwicker, and F. Durand, "Texture design using a simplicial complex of morphable textures," in *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, (New York, NY, USA), pp. 787–794, ACM, 2005.

[49] R. Fattal, D. Lischinski, and M. Werman, "Gradient domain high dynamic range compression," in *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, SIGGRAPH, (New York, NY, USA), pp. 249–256, ACM, 2002.

[50] P. Bhat, B. Curless, M. Cohen, and L. Zitnick, "Fourier analysis of the 2d screened poisson equation for gradient domain problems," in *ECCV*, 2008.

[51] O. Whyte, J. Sivic, and A. Zisserman, "Get out of my picture! internet-based inpainting," in *Proceedings of the 20th British Machine Vision Conference, London*, 2009.

[52] R. Ruiters, R. Schnabel, and R. Klein, "Patch-based texture interpolation," *Computer Graphics Forum*, vol. 29, pp. 1421–1429, June 2010.

[53] M. Tappen, W. Freeman, and E. Adelson, "Recovering intrinsic images from a single image," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, pp. 1459 –1472, sept. 2005.

[54] E. Candes, M. Rudelson, T. Tao, and R. Vershynin, "Error correction via linear programming," in *IEEE Symposium on Foundations of Computer Science*, pp. 668 –681, oct. 2005.

[55] J. Tropp and A. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Information Theory*, vol. 53, pp. 4655 –4666, dec. 2007.

[56] Adobe, "Photoshop cs5 content-aware fill," 2010.

[57] H. Fang and J. C. Hart, "Detail preserving shape deformation in image editing," *ACM Trans. Graphics*, vol. 26, no. 3, p. 12, 2007.

[58] R. Szeliski, M. Uyttendaele, and D. Steedly, "Fast poisson blending using multisplines," in *Computational Photography (ICCP), 2011 IEEE International Conference on*, pp. 1 –8, april 2011.

[59] K. He and J. Sun, "Statistics of patch offsets for image completion," in *ECCV*, 2012.

[60] S. Nayar and T. Mitsunaga, "High dynamic range imaging: spatially varying pixel exposures," in *Proceedings of CVPR 2000*, pp. 472 – 479, 2000.

[61] M. D. Tocci, C. Kiser, N. Tocci, and P. Sen, "A Versatile HDR Video Production System," *ACM Transactions on Graphics (TOG) (Proceedings of SIGGRAPH 2011)*, vol. 30, no. 4, pp. 41:1–41:10, 2011.

[62] S. Mann and R. W. Picard, "On being 'undigital' with digital cameras: Extending dynamic range by combining differently exposed pictures," in *Proceedings of Society for Imaging Science and Technology*, pp. 442–448, 1995.

[63] P. E. Debevec and J. Malik, "Recovering high dynamic range radiance maps from photographs," in *Proceedings of ACM SIGGRAPH 1997*, pp. 369–378, 1997.

[64] T. Ratcliff, "Stuck in Customs HDR Photography," 2012. http://www.stuckincustoms.com.

[65] Y. Wexler, E. Shechtman, and M. Irani, "Space-time completion of video," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, pp. 463–476, march 2007.

[66] D. Simakov, Y. Caspi, E. Shechtman, and M. Irani, "Summarizing visual data using bidirectional similarity," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2008*, pp. 1–8, june 2008.

[67] E. Shechtman, A. Rav-Acha, M. Irani, and S. Seitz, "Regenerative morphing," in *IEEE Conference on Computer VIsion and Pattern Recognition (CVPR)*, (San-Francisco, CA), June 2010.

[68] E. Reinhard, W. Heidrich, P. Debevec, S. Pattanaik, G. Ward, and K. Myszkowski, *High Dynamic Range Imaging: Acquisition, Display, and Image-Based Lighting*. Morgan Kaufmann, second ed., 2010.

[69] F. Banterle, A. Artusi, K. Debattista, and A. Chalmers, *Advanced High Dynamic Range Imaging: Theory and Practice*. Natick, MA, USA: AK Peters (CRC Press), 2011.

[70] X. Liu and A. El Gamal, "Synthesis of high dynamic range motion blur free image from multiple captures," *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on*, vol. 50, pp. 530–539, april 2003.

[71] T. Grosch, "Fast and robust high dynamic range image generation with camera and object movement," in *Vision, Modeling and Visualization*, pp. 277–284, 2006.

[72] K. Jacobs, C. Loscos, and G. Ward, "Automatic high-dynamic range image generation for dynamic scenes," *Computer Graphics and Applications, IEEE*, vol. 28, pp. 84–93, march-april 2008.

[73] T. Jinno and M. Okuda, "Motion blur free hdr image acquisition using multiple exposures," in *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pp. 1304–1307, oct. 2008.

[74] D. Sidibe, W. Puech, and O. Strauss, "Ghost detection and removal in high dynamic range images," in *Proceedings EUSIPCO)*, pp. 2240–2244, Aug. 2009.

[75] O. Gallo, N. Gelfand, W. Chen, M. Tico, and K. Pulli, "Artifact-free high dynamic range imaging," *Proceedings of IEEE ICCP*, April 2009.

[76] T.-H. Min, R.-H. Park, and S. Chang, "Histogram based ghost removal in high dynamic range images," in *Proceedings of IEEE ICME*, ICME'09, (Piscataway, NJ, USA), pp. 530–533, IEEE Press, 2009.

[77] S. Wu, S. Xie, S. Rahardja, and Z. Li, "A robust and fast anti-ghosting algorithm for high dynamic range imaging," in *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pp. 397–400, sept. 2010.

[78] F. Pece and J. Kautz, "Bitmap movement detection: HDR for dynamic scenes," in *Visual Media Production (CVMP), 2010 Conference on*, pp. 1–8, nov. 2010.

[79] S. Raman and S. Chaudhuri, "Reconstruction of high contrast images for dynamic scenes," *The Visual Computer*, vol. 27, no. 12, pp. 1099–1114, 2011.

[80] W. Zhang and W.-K. Cham, "Gradient-directed composition of multi-exposure images," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 530–536, june 2010.

[81] W. Zhang and W. Cham, "Gradient-directed multi-exposure composition," *Image Processing, IEEE Transactions on*, vol. PP, no. 99, p. 1, 2011.

[82] E. Khan, A. Akyuz, and E. Reinhard, "Ghost removal in high dynamic range images," in *Proceedings of IEEE ICIP*, pp. 2005–2008, oct. 2006.

[83] A. Eden, M. Uyttendaele, and R. Szeliski, "Seamless image stitching of scenes with large motions and exposure differences," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, pp. 2498–2505, 2006.

[84] Y. S. Heo, K. M. Lee, S. U. Lee, Y. Moon, and J. Cha, "Ghost-free high dynamic range imaging," in *Proceedings of the 10th Asian conference on Computer vision - Volume Part IV*, ACCV'10, (Berlin, Heidelberg), pp. 486–500, Springer-Verlag, 2010.

[85] Photomatix, "HDR processing software," 2012. http://www.hdrsoft.com/.

[86] L. G. Brown, "A survey of image registration techniques," *ACM Comput. Surv.*, vol. 24, pp. 325–376, December 1992.

[87] B. Zitová and J. Flusser, "Image registration methods: a survey," *Image and Vision Computing*, vol. 21, pp. 977–1000, 2003.

[88] T. Mitsunaga and S. Nayar, "Radiometric self calibration," in *Proceedings of CVPR 1999*, vol. 1, pp. 374–380, Jun 1999.

[89] G. Ward, "Fast, robust image registration for compositing high dynamic range photographs from hand-held exposures," *journal of graphics, gpu, and game tools*, vol. 8, no. 2, pp. 17–30, 2003.

[90] A. Tomaszewska and R. Mantiuk, "Image registration for multi-exposure high dynamic range image acquisition," in *Proceedings of the 15th International Confer-*

ence in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG) 2007, 2007.

[91] A. O. Akyüz, "Photographically guided alignment for hdr images," in *Eurographics 2011 - Areas Papers*, (Llandudno, UK), pp. 73–74, Eurographics Association, 2011.

[92] S. Yao, "Robust image registration for multiple exposure high dynamic range image synthesis," in *Proceedings of SPIE 7870*, SPIE, January 2011.

[93] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Seventh International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 674–679, 1981.

[94] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *Int. J. Comput. Vision*, vol. 92, pp. 1–31, March 2011.

[95] L. Bogoni, "Extending dynamic range of monochrome and color images through fusion," in *ICPR*, pp. 3007–3016, 2000.

[96] S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High dynamic range video," *ACM Trans. Graph.*, vol. 22, pp. 319–325, July 2003.

[97] S. Mangiat and J. Gibson, "High dynamic range video with ghost removal," in *Proc. SPIE 7798*, no. 779812, 2010.

[98] L. Xu, J. Jia, and Y. Matsushita, "Motion detail preserving optical flow estimation," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 1293–1300, june 2010.

[99] T. Brox and J. Malik, "Large displacement optical flow: Descriptor matching in variational motion estimation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, pp. 500–513, march 2011.

[100] C. Liu, *Beyond Pixels: Exploring New Representations and Applications for Motion Analysis*. Doctoral thesis, Massachusetts Institute of Technology, May 2009.

[101] T. Brox, A. Bruhn, N. Papenberg, and J.Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Computer Vision–ECCV 2008*, 2004.

[102] A. Bruhn, J. Weickert, and C. Schnörr, "Lucas/kanade meets horn/schunck: combining local and global optic flow methods," *Int. J. Comput. Vision*, vol. 61, pp. 211–231, February 2005.

[103] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patchmatch: a randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, pp. 24:1–24:11, July 2009.