Analysis and Transfer of Photographic Viewpoint and Appearance

by

Soonmin Bae

B.S., Korea Advanced Institute of Science and Technology (2003)
M.S., Massachusetts Institute of Technology (2005)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2009

Author .....
Department of Electrical Engineering and Computer Science
August 14, 2009

Certified by .....
Frédo Durand
Associate Professor
Thesis Supervisor

Accepted by .....
Terry P. Orlando
Chairman, Department Committee on Graduate Theses

Analysis and Transfer of Photographic Viewpoint and Appearance

by

Soonmin Bae

Submitted to the Department of Electrical Engineering and Computer Science
on August 14, 2009, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Electrical Engineering and Computer Science

Abstract

To make a compelling photograph, photographers need to carefully choose the subject and composition of a picture, to select the right lens and viewpoint, and to make great efforts with lighting and post-processing to arrange the tones and contrast. Unfortunately, such painstaking work and advanced skill is out of reach for casual photographers. In addition, for professional photographers, it is important to improve workflow efficiency.

The goal of our work is to allow users to achieve a faithful viewpoint for rephotography and a particular appearance with ease and speed. To this end, we analyze and transfer properties of a model photo to a new photo. In particular, we transfer the viewpoint of a reference photo to enable rephotography. In addition, we transfer photographic appearance from a model photo to a new input photo.

In this thesis,we present two contributions that transfer photographic view and look using model photographs and one contribution that magnifies existing defocus given a single photo. First, we address the challenge of viewpoint matching for rephotography. Our interactive, computer-vision-based technique helps users match the viewpoint of a reference photograph at capture time. Next, we focus on the tonal aspects of photographic look using post-processing. Users just need to provide a pair of photos, an input and a model, and our technique automatically transfers the look from the model to the input. Finally, we magnify defocus given a single image. We analyze the existing defocus in the input image and increase the amount of defocus present in out-of focus regions.

Our computational techniques increase users' performance and efficiency by analyzing and transferring the photographic characteristics of model photographs. We envision that this work will enable cameras and post-processing to embed more computation with a simple and intuitive interaction.

Thesis Supervisor: Frédo Durand
Title: Associate Professor

Acknowledgments

Thanks to my advisor, Prof. Frédo Durand, for guiding my research work, for supporting my family life, and for being a true mentor. He has been a great teacher and leader.

Thanks to Dr. Aseem Agarwala for co-supervising computational rephotography work. Thanks to my other thesis committee members, Prof. Bill Freeman and Prof. Antonio Torralba, for providing me with their feedback and comments.

Thanks to the Computer Graphics Group for being great friends and for supporting my research through proofreading my drafts, being users, and making videos. I enjoyed my graduate student life thanks to their support and friendship. Special thanks to Dr. Sylvain Paris for his encouragement and co-working on style transfer work.

I thank my family, my parents, my sister, and my parents-in-law for their love and support. In particular, I thank my husband for his understanding and support. And thanks to my daughter for being a good baby and giving us pleasure. I thank my friends in First Korean Church in Cambridge and in KGSA-EECS for their prayer and encouragement.

Lastly, I would like to thank the Samsung Lee Kun Hee Scholarship Foundation for their financial support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A good snapshot stops a moment from running away.

- Eudora Welty (1909-2001)

Photography is a medium of discovery and documentation. Every photograph has its own purpose. Landscape photographs are to capture magnificent sceneries. Artistic landscape photographs evoke emotion and carry the viewer away to the breathtaking beautiful sceneries. Snapshots are to record memorable moments. Good snapshots are tangible reminders that bring the past experience and moment into the present. In addition, photographs can be used to document changes. For example, a photograph taken at the same location many years later can emphasize change over time. This is called re-photography. Rephotographs are to visualize historic continuities and changes. When a photograph and its rephotograph match well, it becomes very evident what is preserved and what changed across time.

Digital technologies have made photography less expensive and more accessible. Still, casual photographers are often disappointed with their photos that are different from what they thought to capture. Landscape photos lack contrast and strength. Portrait photos are not correctly focused or have distracting background clutter. The viewpoint of the rephoto could be reproduced better at capture time. The framing and composition could have been done better.

To make a compelling photograph, photography requires creativity, technical

knowledge, and persistence. Photographers need to carefully choose the subject and composition of a picture, to select the right lens and viewpoint, and to make great efforts with lighting and post-processing to arrange the tones and contrast. Unfortunately, such painstaking work and advanced skill is out of reach for casual photographers. In addition, for professional photographers, it is important to improve workflow efficiency.

Advanced computer vision techniques embedded in recent digital cameras, such as face detection and viewfinder alignment [5], help focus on the right subject and change framing but do not address viewpoint and composition. Software such as Apple's Aperture and Adobe's Lightroom focuses on workflow optimization but offers little interactive editing capabilities.

The goal of our work is to allow users to achieve a faithful viewpoint for rephotography and a particular appearance with ease and speed. To this end, we analyze and transfer properties of a model photo to a new photo. In particular, we transfer the viewpoint of a reference photo to enable rephotography. In addition, we transfer photographic appearance from a model photo to a new input photo.

1.1 Overview of Our Approach

In this thesis, we present two contributions that transfer photographic view and look using model photographs and one contribution that magnifies existing defocus given a single photo. In "Computational Re-Photography", we address the challenge of viewpoint matching for rephotography. Our interactive, computer-vision-based technique helps users match the viewpoint of a reference photograph at capture time. Our technique estimates and visualizes the camera motion required to reach the desired viewpoint. In "Style Transfer", we focus on the tonal aspects of photographic look using post-processing. We decouple the tonal aspects of photos from their contents. Our method handles global and local contrast separately. In "Defocus Magnification", we magnify defocus given a single image. We analyze the existing defocus in the input image and increase the amount of defocus

present in out-of focus regions. Let us describe them more in details below.

1.1.1 Computational Re-Photography

Our real-time interactive technique helps users reach a desired viewpoint and location as indicated by a reference image. This work is inspired by rephotography, the act of repeat photography of the same site. Rephotographers aim to recapture an existing photograph from the same viewpoint. However, we found that most rephotography work is imprecise because reproducing the viewpoint of the original photograph is challenging. The rephotographer must disambiguate between the six degrees of freedom of 3D translation and rotation, and the confounding similarity between the effects of camera zoom and dolly.

The main contribution of our work is the development of the first interactive, computer-vision-based technique for viewpoint guidance. We envision our tool running directly on the digital camera. However, since these platforms are currently closed and do not have enough processing power yet, our prototype consists of a digital camera connected to and controlled by a laptop. At capture time, users do not need to examine parallax manually, but only need to follow our real-time visualization displayed on a computer in order to move to a specific viewpoint and location. The user simply captures a video stream of the scene depicted in the reference image, and our technique automatically estimates the viewpoint and lens difference and guides users to the desired viewpoint. Our technique builds on several existing computer vision algorithms to detect and match common features in two photographs [6] and to compute the relative pose between them [7, 8]. We demonstrate the success of our technique by rephotographing historical images and conducting user studies. We envision that this work would enable cameras to become more interactive and embed more computation.

1.1.2 Style Transfer

Our technique transfers the tonal aspect of photographic look from a model photograph onto an input one. We handle global and local contrast separately. Our method is inspired by traditional photography, where the darkroom offers remarkable global and local control over the brightness, contrast, and sharpness of images via a combination of chemical and optical processes [4, 9].

Our method is based on a two-scale non-linear decomposition of an image. We modify different layers according to their histograms. To transfer the spatial variation of local contrast, we introduce a new edge-preserving textureness that measures the amount of local contrast. We recombine the two layers using a constrained Poisson reconstruction. Finally, additional effects such as soft focus, grain and toning complete our look transfer.

1.1.3 Defocus Magnification

We take a single input image that lacks defocus and increase the amount of defocus present in out-of focus regions. A blurry background due to shallow depth of field is often desired for photographs such as portraits, but, unfortunately, small point-and-shoot cameras do not permit enough defocus because of the small diameter of their lenses and their small sensors. We present an image-processing technique that increases the defocus in an image and simulates the shallow depth of field of a lens with a larger aperture.

Our technique estimates the spatially-varying amount of blur over the image, and then uses a simple image-based technique to increase defocus. We first estimate the size of the blur kernel at edges and then propagate this defocus measure over the image. Using our defocus map, we magnify the existing blurriness, which means that we blur blurry regions and keep sharp regions sharp. In contrast to more difficult problems such as depth from defocus, we do not require precise depth estimation and do not need to disambiguate textureless regions.

1.2 Thesis Overview

This thesis addresses analysis and transfer of photographic viewpoint and appearance. Given a model photo, we transfer its photographic viewpoint and appearance to an input photo. Chapter 2 introduces some background information that inspired this thesis. We provide a brief overview on traditional photography. Chapter 3 discusses previous work that addresses computer vision and computer graphics challenges related to our goal. Chapter 4 presents our viewpoint guidance technique. We describe our method and demonstrate the success of our technique by presenting rephotography results and user study results. Chapter 5 presents our two-scale tone transfer method. We perform a faithful reproduction of the tonal aspects of model photographs. Chapter 6 shows our defocus magnification technique. Our image-processing technique simulates the shallow depth of field effects. Chapter 7 summarizes our work and discusses our future work.

Chapter 2

Background

Our techniques are inspired by and related to traditional photography. This chapter provides a brief overview on camera models including the pinhole camera, lens camera, and view camera, photographic printing, and rephotography.

2.1 Camera Models and Geometric Image Formation

In this section, we review some of the basic camera models related to our technique. First, we describe the geometric image formation by a pinhole camera. Second, we examine a lens camera and the amount of defocus. Finally, we discuss a view camera and its rising front adjustment.

2.1.1 Pinhole Camera and Perspective Projection

Most computer vision algorithms assume perspective projection formed by a pinhole camera. The pinhole camera is the simplest camera model. It maps 3D onto 2D using perspective projection. Rays of light pass through a pinhole and form an inverted image of the object on the image plane, as shown in Figure 2-1.

Using homogeneous coordinates allows projection to be a matrix multiplication, as shown in Equation 2.1. K is a 3×3 intrinsic matrix that maps the 3D camera coordinate to the 2D pixel coordinate in homogeneous coordinates. Equa-

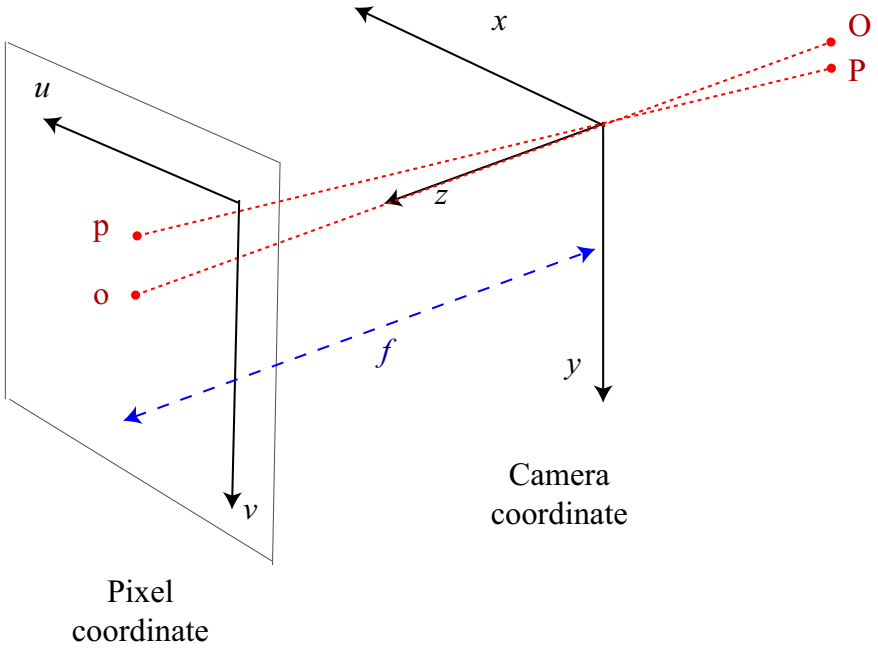


Figure 2-1: Illustration of the image formation using perspective projection of a pinhole camera. A 3D world point P is projected onto a 2D image point p.

tion 2.2 shows that K has five degrees of freedom including skew s, focal length f_y, principal point (u_0, v_0), and aspect ratio f_x/f_y. In this thesis, we assume that there is no skew and the aspect ratio is equal to 1. This leaves us three parameters to estimate: two for the principal point and one for the focal length.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{z} \begin{bmatrix} K & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.1}$$

$$K = \begin{bmatrix} f_x & s & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

2.1.2 Lens Camera and Defocus

In the pinhole camera, a smaller pinhole generally results in sharper images. However, due to the wave properties of light, an extremely small hole can produce diffraction effects and a less clear image. In addition, with a small pinhole, a long exposure is required to generate a bright image. A lens replaces the pinhole to focus the bundle of rays from each scene point onto the corresponding point in the image plane. This substitution makes the image brighter and sharper. Although most camera lenses use more intricate designs with multiple lens elements, here we review the simplified thin lens model, which suffices in our context.

The main role of a lens is to make all the rays coming from a point at the focus distance converge to a point in the image plane. In contrast, the rays originating at a scene point distant from the focus distance converge in front of or behind the image plane, and that point appears as a blurred spot in the image. The blurred spot is called the circle of confusion. It is not strictly speaking a circle and depends on the aperture shape and diffraction, but it is often modeled as a circle or a Gaussian.

We express the circle of confusion diameter c of a point at distance S (Figure 2-2). A detailed derivation can be found in optics textbooks such as Hecht's [10]. Given the focal length f of the lens, the thin-lens formula gives us the lens-sensor distance f_D to focus at distance D: $\frac{1}{f} = \frac{1}{f_D} + \frac{1}{D}$.

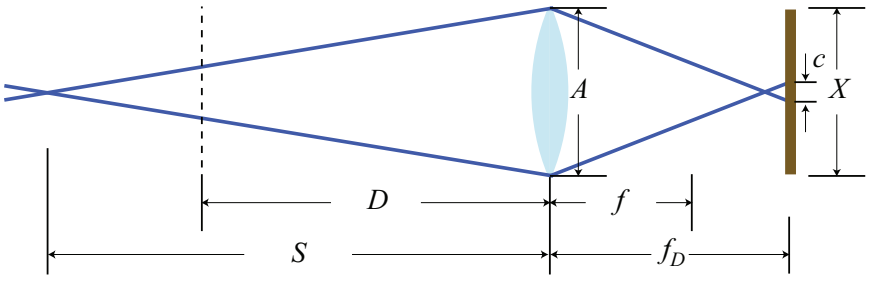


Figure 2-2: A thin-lens system. The lens' diameter is A and its focal length is f. The image plane is at distance f_D from the lens and the focus distance is D. Rays from a point at distance S generates a circle of confusion diameter c. And the rays generates a virtual blur circle diameter C at the focus distance D.

The f-number N gives the aperture diameter A as a fraction of the focal length ($A = Nf$). Note that N has no unit. N is the number, such as 2.8, that photogra-

phers set to control the diaphragm. The aperture is then denoted by, e.g., f/2.8 to express that the diameter is the focal length divided by the f-number. The diameter of the circle of confusion is then

$$c = \frac{|S - D|}{S} \cdot \frac{f^2}{N(D - f)} \tag{2.3}$$

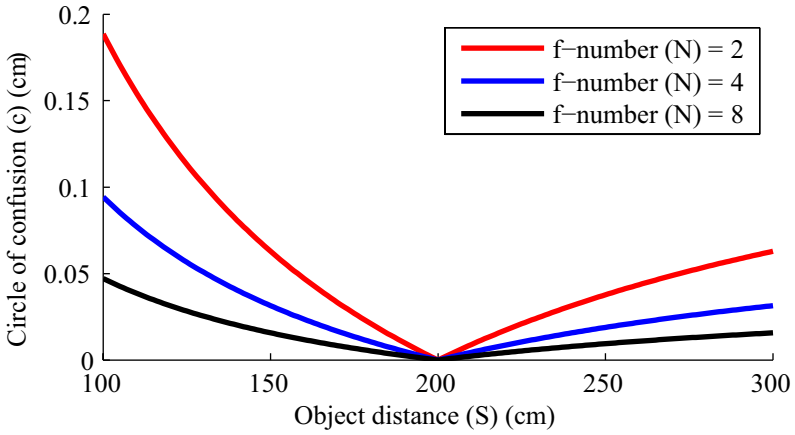


Figure 2-3: This plot shows how the circle of confusion diameter, c, changes according to the change of object distance S and f-number N. c increases as a point is away from the focus distance D. The focus distance D is $200cm$, and the focal length f is $8.5cm$

Figure 2-3 shows that the circle of confusion diameter c increases as a point is away from the focus distance D. The relationship is not linear (hyperbolic) and is not symmetrical for points in front of and behind the plane in focus.

We now study the effect of the sensor size, X. To express the amount of defocus in terms of image-space blur, we use the relative size of the circle of confusion $c' = c/X$. For sensors of different sizes, the same field of view is obtained if the relative focal length $f' = f/X$ is the same. Replacing c and f by their relative version in Eq. 2.3 we obtain

$$c' = \frac{|S - D|}{S} \cdot \frac{f'^2 X}{N(D - f'X)} \tag{2.4}$$

To a first-order approximation, the amount of defocus is proportional to the sensor size X. This confirms that for a given f-number N, a smaller sensor does

not yield as much defocus as a larger sensor. More defocus could be achieved by using a smaller f-number (larger lens aperture), but this would require bending rays that reach the periphery of the lens aperture by angles that are physically challenging to achieve. Scaling down the sensor and lens inherently scales down the amount of defocus. In this thesis, we synthesize the shallow depth of field effect by increasing the amount of defocus existing in an input image using image processing techniques.

2.1.3 View Camera and Principal Point

The view camera is the most versatile of the large-format cameras. Historical photographs were often taken with view cameras. The view camera comprises a flexible bellows, which allow photographers to control focus and convergence of parallel lines by varying the distance between the lens and the film over a large range. Figure 2-4 illustrates a view camera.

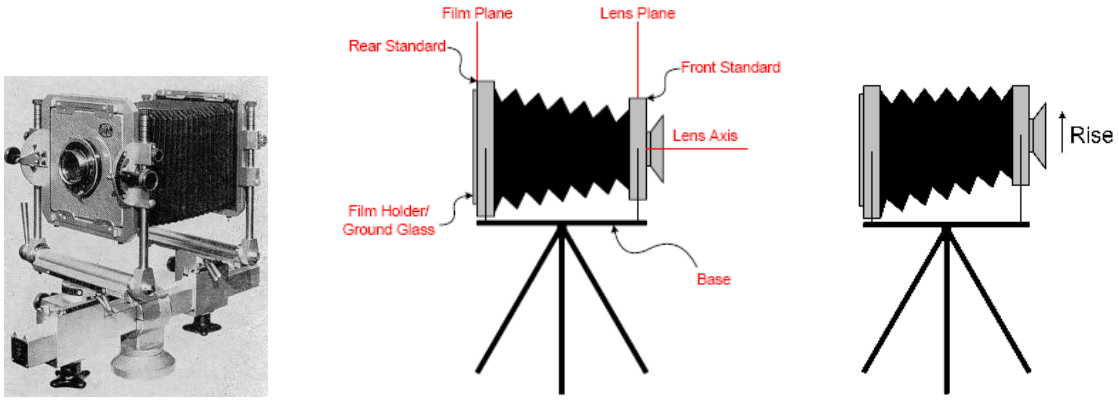


Figure 2-4: View cameras and its movement of the standard front rise. (Images from wikipedia.)

In particular, the rising front adjustment is a very important movement especially in architectural photography. The lens is moved vertically up along the lens plane in order to change the portion of the image that will be captured on the film. Figure 2-5 demonstrates the effect of rising front. The main effect of rise is to eliminate converging parallels when photographing tall buildings. If a camera is pointed at a tall building without rise movement nor tilt, the top will be cut off. If

the camera is tilted upwards to get it all in, the film plane will not be parallel to the building, and the top of the building will appear narrower than its bottom. That is, parallel lines in the object will converge in the image.

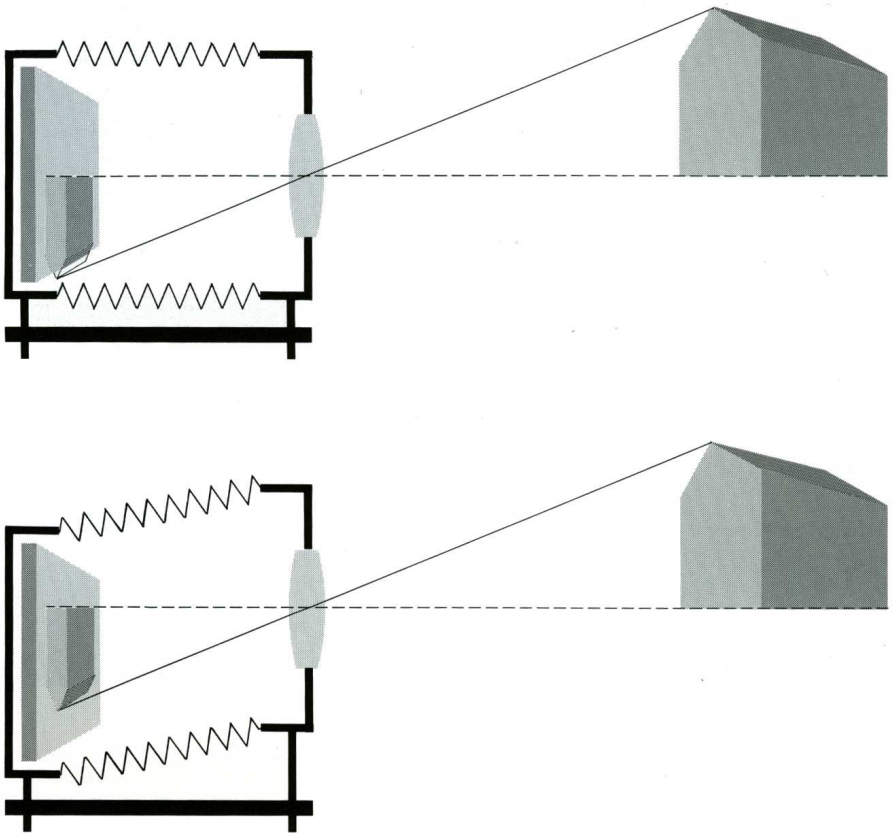


Figure 2-5: Effect of rising front. The lens is moved vertically up along the lens plane in order to change the portion of the image that will be captured on the film. As a result of rise, the principal point is not located at the image center, but at the bottom of the image. (Images from The Camera [1] by Adams)

The principal point is the intersection of the optical axis with the image plane. As a result of rise, the principal point does not locate at the image center, but at the image bottom. When we estimate the viewpoint of the reference photo, we estimate the location of the principal point as preprocessing.

2.2 Rephotography

Rephotography is the act of repeat photography; capturing a photograph of the same scene from the same viewpoint of an existing photograph typically much older. Figure 2-6 shows two examples. A photograph and its rephotograph can provide a compelling "then and now" visualization of the progress of time. When a photograph and its rephotograph match well, a digital cross-fade between the two is a remarkable artifact. A hundred years go by in the blink of an eye, and it becomes immediately evident which scene elements are preserved across time, and which have changed.

However, precise rephotography requires a careful study of the viewpoint, lens, season, and framing of the original image. Exactly matching a photograph's viewpoint "by eye" is remarkably challenging. The rephotographer must disambiguate between the six degrees of freedom of 3D translation and rotation, and the confounding similarity between the effects of camera zoom and dolly. Our viewpoint visualization technique automatically estimates the lens and viewpoint differences. Users only need to follow our real-time visualization in order to move to a specific viewpoint and location instead of manually examining parallax.

2.3 Traditional Photographic Printing

Photographic printing is the process of producing a final image on paper. Traditional photographic printing is performed in a photographic darkroom. The darkroom offers remarkable global and local control over the brightness, contrast, and sharpness of images via a combination of chemical and optical processes [9, 4]. As a result, black-and-white photographs vary in their tonal palette and range. Photographers like Adams (Fig. 2-8a) exhibit strong contrast with rich blacks, while artists like Stieglitz (Fig. 5-15a) rely more on the mid-tones. This suggests the intensity histogram as a characterization of tonal look.

However, we propose that the spatial distribution of tones must be taken into

Figure 2-6: Rephotography gives two views of the same place around a century apart. Pictures are from New York Changing [2] and Boston Then and Now [3].

Figure 2-7: Ansel Adams using a dodging tool (from The Print [4] by Adams). He locally controls the amount of light reaching the photographic paper.

account because a histogram does not deal with local contrast. The amount of texture is crucial in photographs; some artists use vivid texture over the entire image (Fig. 2-8b), while others contrast large smooth areas with strong textures in the remaining parts of the image (Fig. 2-8a). Furthermore, the human visual system is known to be more sensitive to local contrast than to low spatial frequencies. Finally, a photograph is characterized by low-level aspects of the medium such as tone (e.g. sepia toning) and grain (controlled by the film and paper characteristics). These observations drive our approach. We use decompositions of an image that afford direct control over dynamic range, tonal distribution, texture and sharpness.

(a) (b)

Figure 2-8: Typical model photographs that we use. Photo (a) exhibits strong contrast with rich blacks, and large textured areas. Photo (b) has mid-tones and vivid texture over the entire image.

Chapter 3

Related Work

Computational photography extends the capabilities of digital photography. This thesis focus on transfer of photographic characteristics including viewpoint, tonal aspects, and defocus. In this chapter, we review previous work that addresses related challenges.

3.1 Viewpoint Estimation

To the best of our knowledge, we are the first to build an interactive tool that directs a person to the viewpoint of a reference photograph. However, estimating camera positions and scene structures from multiple images has been a core problem in the computer vision community [11, 12, 8, 13].

We visualize the desired movement at capture time. One alternative to our approach is to capture a nearby viewpoint and warp it to the desired viewpoint after capture [14, 15, 16]. However, parallax and complex scene geometry can be challenging for these algorithms, and the result would be inaccurate.

Our technique is related to *visual homing* research in robotics, where a robot is directed to a desired 3D location (e.g., a charging station) specified by a photograph captured from that location. The visual homing approach of Basri et al. [17] also exploits feature matches to extract relative pose; the primary difference is that robots can respond to precise motion parameters, while humans respond

better to visualizations in a trial and error process. More recent work exists on real-time algorithms that recover 3D motion and structure [18, 19], but they do not aim to guide humans. There exist augmented reality systems [20] that ease navigation. However they assume that the 3D model is given, while the only input to our technique is an old photograph taken by an unknown camera.

We are not the first to exploit the power of historical photographs. The 4D Cities project (www.cc.gatech.edu/4d-cities) hopes to build a time-varying 3D model of cities, and Photo Tourism [21] situated older photographs in the spatial context of newer ones. However, neither project helps a user capture a new photograph from the viewpoint of a historical one.

Real-time visualization on a camera allows photographers to achieve a variety of tasks. It helps photographers focus on the right subject and change framing and settings. In addition to the traditional simple functions, advanced computer vision techniques have been embedded in recent digital cameras and mobile phones. Face detection, the Viewfinder Alignment of Adams et al. [5], feature matching and tracking on mobile phones [22, 23], and the Panoramic Viewfinder of Baudisch et al. [24] are such examples. The Panoramic Viewfinder is related to our technique, though its focus is the real-time preview of the coverage of a panorama with no parallax. The implementation of matching and tracking algorithms on mobile phones is complementary to our technique. We focus on the development of an interactive visualization method based on those tools.

3.2 Style Transfer

The frequency contents have been used to measure statistical characteristics of natural images. Field investigated the two-dimensional amplitude spectra and found regularity among natural images that the average amplitude falls as " $\frac{1}{f}$ " [25]. This property is called scale invariance [26]. We observe that natural images follow such scale invariant property, while artistic photographs do not show such invariance. In addition to the scale invariance, image statistics have been shown to

be non-stationary [27]: local statistical features vary with spatial location. In this thesis, we use the scale variance and the variations in non-stationarity to analyze and transfer photographic styles.

Tone-mapping seeks the faithful reproduction of high-dynamic-range images on low-dynamic-range displays, while preserving visually important features [28]. Our work builds on local tone mapping where the mapping varies according to the neighborhood of a pixel [29, 30, 31, 32, 33, 34]. The precise characteristics of film have also been reproduced [35, 31]. However, most techniques seek an objective rendering of the input, while we want to facilitate the exploration and transfer of particular pictorial looks.

Style transfer has been explored for the textural aspects of non-photorealistic media, e.g. [36, 37], and DeCarlo et al. stylize photographs based on saliency [38]. In contrast, we seek to retain photorealism and control large-scale effects such as tonal balance and the variation of local detail. In addition, our parametric approach leads to continuous changes supported by interactive feedback and enables interpolations and extrapolations of image look.

Our work is inspired by the ubiquitous visual equalizer of sound devices. Similarly, the modification of frequency bands can alter the "mood" or "style" of motion data [39]. The equivalent for images is challenging because of the halos that frequency decomposition can generate around edges. Our work can be seen as a two-band equalizer for images that uses non-linear signal processing to avoid halos and provides fine tonal and spatial control over each band.

Recently manual adjustment tools [40, 41] are developed and multiscale edge-preserving decomposition schemes [42, 43] are introduced. There are complementary to our technique. We aim to transfer photographic looks from model photographs.

3.3 Defocus

Defocus effects have been an interest of the Computer Vision community in the context of recovering 3D from 2D. Camera focus and defocus have been used to reconstruct depth or 3D scenes from multiple images: depth from focus [44, 45, 46, 47, 48] and depth of defocus [49, 46, 50, 51, 52, 53]. These methods use multiple images with different focus settings and estimate the corresponding depth for each pixel. They have to know the focus distance and focal length to computer the depth map. In contrast, we do not estimate the depth but the blur kernel. Recently, specially designed cameras [54, 55, 56] are introduced to infer depth or blur kernel after the picture has been taken. We want to treat this problem without the help of any special camera settings, but only with image post-processing techniques.

Image processing methods have been introduced to modify defocus effects without reconstructing depth. Eltoukhy and Kavusi [57] use multiple photos with different focus settings and fuse them to produce an image with extended depth of field. Özkan et al. [58] and Trussell and Fogel [59] have developed a system to restore space-varying blurred images and Reeves and Mersereau [60] find a blur model to restore blurred images. This is the opposite of what we want to do. They want to restore blurred images, while we want to increase existing blurriness.

Kubota and Aizawa [61] use linear filters to reconstruct arbitrarily focused images from two differently focused images. On the contrary, we want to modify defocus effects only with a single image. Lai et al. [62] use a single image to estimate the defocus kernel and corresponding depth. But their method only works on an image composed of straight lines at a spatially fixed depth.

Given an image with a corresponding depth map, depth of field can be approximated using a spatially-varying blur, e.g. [63, 64], but note that special attention must be paid to occlusion boundaries [65]. Similar techniques are now available in commercial software such as Adobe® Photoshop® (lens blur) and Depth of Field Generator Pro (*dofpro.com*). In our work we simply use these features and instead of providing a depth map, we provide a blurriness map estimated from the photo.

While the amount of blurriness is only related to depth and is not strictly the same as depth, we have found that the results qualitatively achieve the desired effect and correctly increase defocus where appropriate. Note that a simple remapping of blurriness would yield a map that resembles more closely a depth map.

Chapter 4

Computational Re-Photography

This work seeks to facilitate rephotography, the act of repeat photography of the same site. Rephotographers aim to recapture an existing photograph from the same viewpoint. However, we found that most rephotography work is imprecise and tedious because reproducing the viewpoint of the original photograph is challenging. The rephotographer must disambiguate between the six degrees of freedom of 3D translation and rotation, and the confounding similarity between the effects of camera zoom and dolly.

Our real-time interactive technique helps users reach a desired viewpoint and location as indicated by a reference image. In a pilot user study, we observed that people could not estimate depth by examining parallax. In our second user study, we showed the relative viewpoint information in 3D. Users found it hard to interpret 3D information; most had a hard time separating translation and rotation of the camera. The main contribution of our work is the development of the first interactive, computer-vision based visual guidance technique for human motion. Users only need to follow our real-time visualization displayed on a computer in order to move to a specific viewpoint and location instead of manually examining parallax.

4.1 Overview

Figure 4-1: In our prototype implementation, a laptop is connected to a camera. The laptop computes the relative camera pose and visualizes how to translate the camera with two 2D arrows. Our alignment visualization allows users to confirm the viewpoint.

Ultimately we wish to embed the whole computation onto the camera, but our current implementation has a laptop connected to a camera as shown in Figure 4-1 shows. The camera's viewfinder images are streamed out to the laptop. The laptop computes the relative camera pose and visualizes the direction to move the camera. We give an overview of how users interact with our technique and how we estimate and visualize the direction to move.

4.1.1 User interaction

We have realized an informal pilot user study to understand the challenges of rephotography. We found that users have much difficulty judging parallax and the effect of viewpoint change. In addition, they are confused by the number of degrees of freedom: camera translation, rotation, and zoom. We have experimented a number of simple visualizations that combine the reference and current images to try to facilitate rephotography, including the difference image, a linear-blend composite, and side-by-side views. In all cases, users had difficulties interpreting the displays.

This motivates our approach, which seeks to minimize the number of degrees

of freedom that the user needs to control to translation only, and displays direct guidance in the form of arrows. We only provide an alignment visualization to help with the end game when the user is close enough to the desired viewpoint. Furthermore, this alignment visualization is made easier to interpret by computationally solving for the extra degrees of freedom (rotation and zoom), automatically applying the appropriate homography warp to the current image to best match the reference.

Our approach takes as input a reference (old) image. We require the user to take a first pair of photographs of the scene from viewpoints that are distinct enough for calibration and that provide wide baseline to avoid degeneracies. In the case of an old reference photograph, the user clicks on a few correspondence points in those original images because the difference in appearance between old and new photos confuses even state-of-the art feature matching. Our interactive technique then automatically computes the difference between the current viewpoint and that of the reference photograph. It displays the required translation vector to guide the user, and refreshes several times per second. An alignment visualization between the reference and current images is also refreshed continuously. The automatic compensation for difference in orientation and zoom (within reason) allows the user to focus on parallax and the viewpoint itself.

4.1.2 Technical overview

To estimate and visualize the required translation, we leverage a number of computer-vision methods. In particular, we need to match features between images and deduce relative camera pose in real time. To achieve both real-time performance and robustness, we use an interleaved strategy where a fast but simpler method is refreshed periodically by a more involved and more robust method. Our robust but slower estimation is based on SIFT matching [6] and a robust relative pose estimation based on RANSAC [66] and Stewénius's five-point algorithm [7]. Our lightweight estimation tracks the feature points estimated trustworthy by the ro-

bust process using simple tracking [67] and updates the relative pose accordingly. Figure 4-2 shows our full pipeline.

In addition to very different appearance of historical photos, their focal lengths and principal points are unknown; historical photographs of urban and architectural scenes were often taken with view cameras where the photographer would move the optical axis off center to keep verticals parallel while taking photographs from a low viewpoint. In a preprocess, we estimate the principal point and register the reference camera to the first frame. To this end, we triangulate feature points in the first pair of photographs into 3D points. Given a few correspondence points provided by the user, we compute the pose and focal length of the reference camera with respect to the known 3D points using the Levenberg-Marquardt (LM) nonlinear optimization algorithm [68, 69]. The resulting relative camera pose between the first frame and the reference is used throughout our main estimation. In our main estimation, we estimate the camera pose relative to the first view instead of the reference. Since we know the reference camera location relative to the first-view, we can derive the relative pose between the current and reference photos.

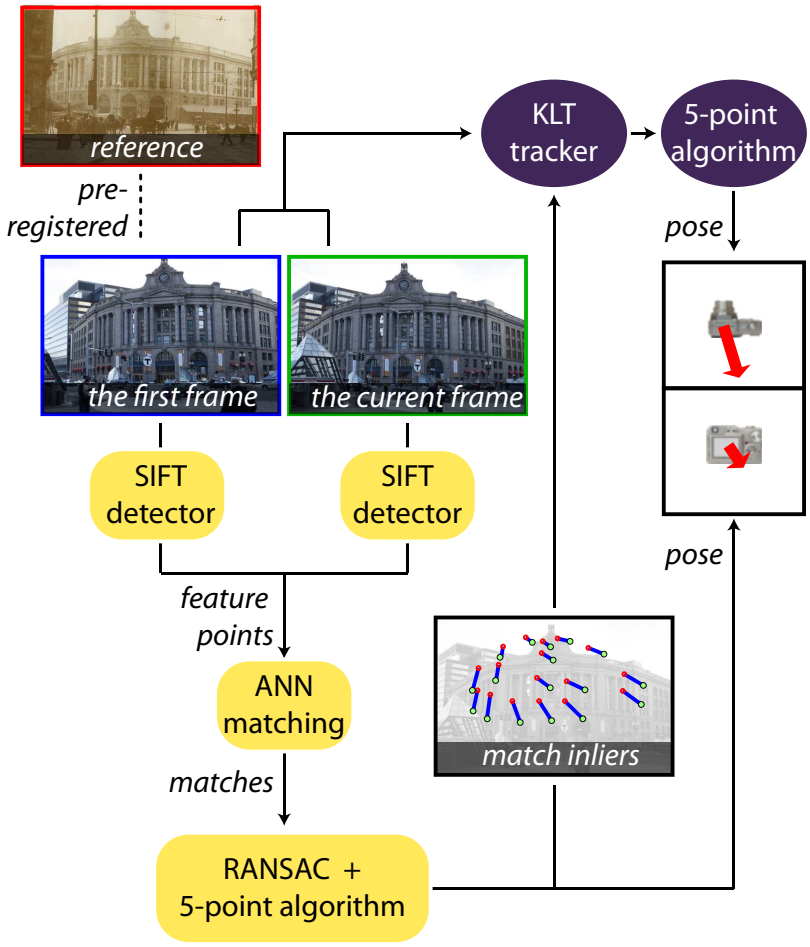


Figure 4-2: Overview of our full pipeline. In a preprocess, we register the reference camera to the first frame. In our main process, we use an interleaved strategy where a lightweight estimation is refreshed periodically by a robust estimation to achieve both real-time performance and robustness. Yellow rounded rectangles represent robust estimation and purple ellipses are for lightweight estimation. The robust estimation passes match inliers to the lightweight estimation at the end.

4.2 Preprocess

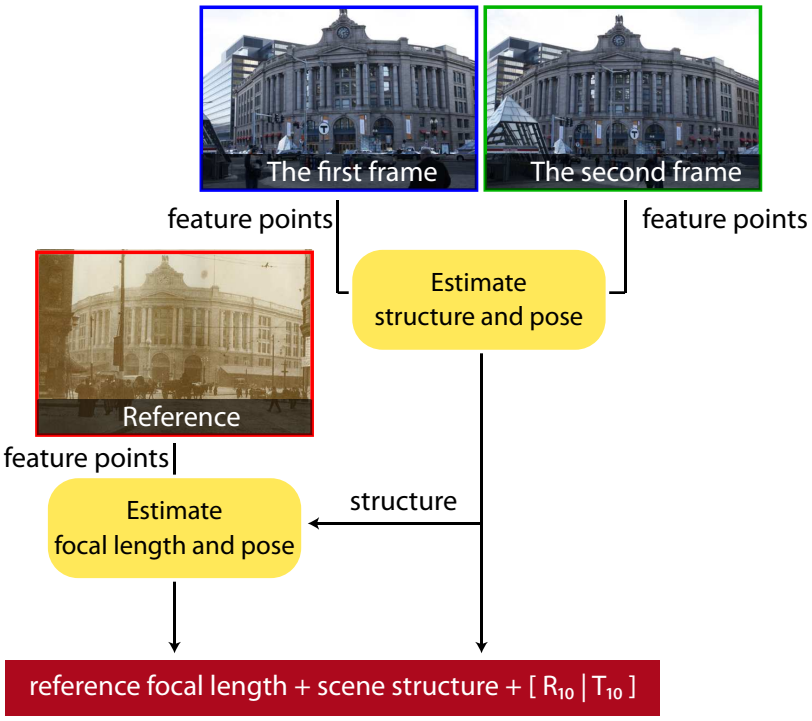


Figure 4-3: Preprocessing to register the reference camera.

At runtime, our system is given a reference image and a current frame and visualizes the translational component of the relative pose. However, there are four main challenges for viewpoint estimation for rephotography:

1. Historical images have very different appearance from current scenes because of, e.g., architectural modifications, the film response, aging, weather and time of day changes. These dramatic differences challenge even state-of-the art feature descriptors. Comparing gradients and pixel values cannot find matches between old photographs and current scenes.

2. The focal length and principal point of the reference camera are unknown, and we have only one image.

3. The resulting translational component from camera pose estimation is unit-length due to scale ambiguities; the length of the translational component is not meaningful across frames.

4. Rephotography seeks to minimize the translational component between the reference and the current. Unfortunately, pose estimation suffers from motion degeneracy where there is no translation [**?**]. Even a narrow baseline results in unstable pose estimation. The baseline refers to the distance between the camera centers.

To overcome these challenges, we use a wide-baseline 3D reconstruction. In preprocessing, we extract a 3D structure of the scene from the first two calibrated images using Structure from Motion (SfM) [13]. We calibrate the current cameras using the Camera Calibration Toolbox for Matlab [70]. With respect to the known 3D structure, we compute the pose and focal length of the old reference camera using Levenberg-Marquardt (LM) nonlinear optimization. The input to the nonlinear optimization includes 2D-3D correspondences and an estimated principal point of the reference photo. In the case of an old reference photograph, we ask users to select six to eight correct matches between the reference image and the second frame. We estimate the principal point of the reference image using its vanishing points. This preprocessing outputs the relative camera pose between the first frame and the reference, which will be used throughout our main estimation.

4.2.1 A wide-baseline 3D reconstruction

We reconstruct the 3D structure of the scene from the first and second view using triangulation [13]. This 3D information is used to estimate the reference focal length and consistent scale over frames. For a wide-baseline 3D reconstruction, we require the user to take a first pair of photographs of the scene from viewpoints that are distinct enough. That is, we ask the user to take the first frame after rotating the camera 20 degrees around the main subject from the initial guess (See Fig 4-4.) The interactive process then starts from the second photo. Consequently, the baseline between the first view and the following frames and the baseline between the first view and the desired viewpoint become wide. Using a first view guarantees wide enough baseline and improves the accuracy of the camera pose

algorithm.

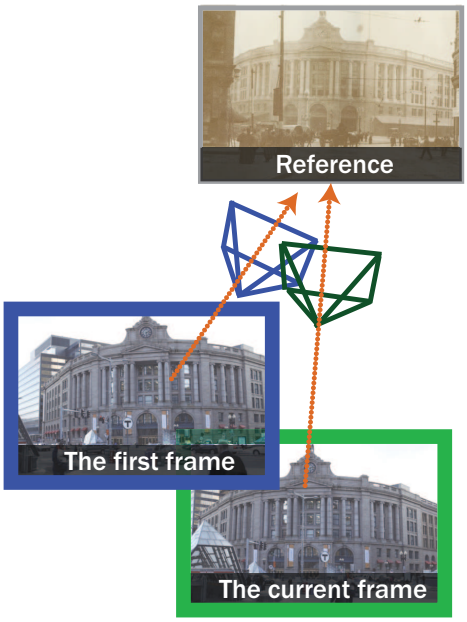


Figure 4-4: How to take the first photo rotated from where a user guesses to be the desired viewpoint.

4.2.2 Reference camera registration

We relate the reference image to the reconstructed world from the first two photos taken by the user. Given matches between the reference and a current photo, we infer the intrinsic and extrinsic parameters of the reference camera using Lourakis's LM package [71]. We assume that there is no skew. This leaves us nine degrees of freedom: one for the focal length, two for the principal point, three for rotation, and three for translation. We initialize the rotation matrix to be the identity matrix, the translation matrix to be zero, and the focal length to be the same as the current camera. We initialize the principal point by analyzing the vanishing points. We describe the details below.

Although this initialization is not close to the ground truth, we observe that the Levenberg-Marquardt algorithm converges to the correct answer since we allow only 9 degrees of freedom and the rotation matrix tends to be close to the identity matrix for rephotographying. The final input to the nonlinear optimization is a set

of matches between the reference and the first photo that users enter. Since we have already triangulated feature points in the first pair of photographs into 3D points, we know the 3D positions of the matched feature points. We compute the pose and focal length of the reference camera with respect to the known 3D locations. In addition, we re-project all 3D points from the first pair to the reference. These points are used to estimate the appropriate homography warp for our alignment visualization.

Principal point estimation

The principal point is the intersection of the optical axis with the image plane. If a shift movement is applied to preserve the verticals parallel or if the image is cropped, the principal point is not in the center of the image, and it must be computed. The analysis of vanishing points provides strong cues for inferring the location of the principal point. Under perspective projection, parallel lines in space appear to meet at a single point in the image plane. This point is the vanishing point of the lines, and it depends on the orientation of the lines. Given the vanishing points of three orthogonal directions, the principal point is located at the orthocenter of the triangle with vertices the vanishing points [13], as shown in Figure 4-5.

We ask the users to click on three parallel lines in the same direction; although two parallel lines are enough for computation, we ask for three to improve robustness. We compute the intersections of the parallel lines. We locate each vanishing point at the weighted average of three intersections. The weight is proportional to the angle between two lines [72]. We discard the vanishing point when the sum of the three angles is less than 5 degrees.

With three finite vanishing points, we initialize and constrain the principal point as the orthocenter. If we have one finite and two infinite vanishing points, we initialize and constrain the principal point as the finite vanishing point. With two finite vanishing points, we constrain the principal point to be on the vanishing line that connects the finite vanishing points.

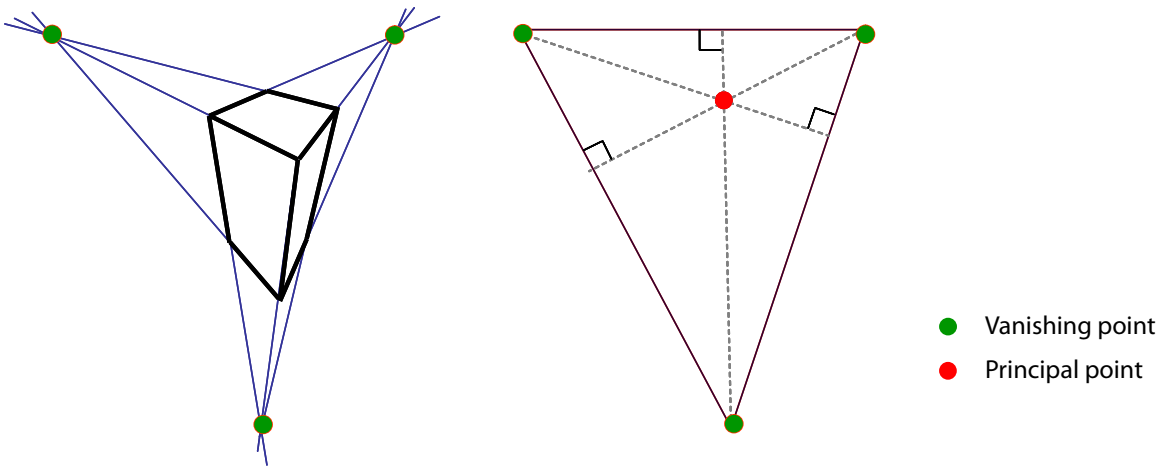


Figure 4-5: Under perspective projection, parallel lines in space appear to meet at the vanishing point in the image plane. Given the vanishing points of three orthogonal directions, the principal point is located at the orthocenter of the triangle with vertices the vanishing points

4.2.3 Accuracy and robustness analysis

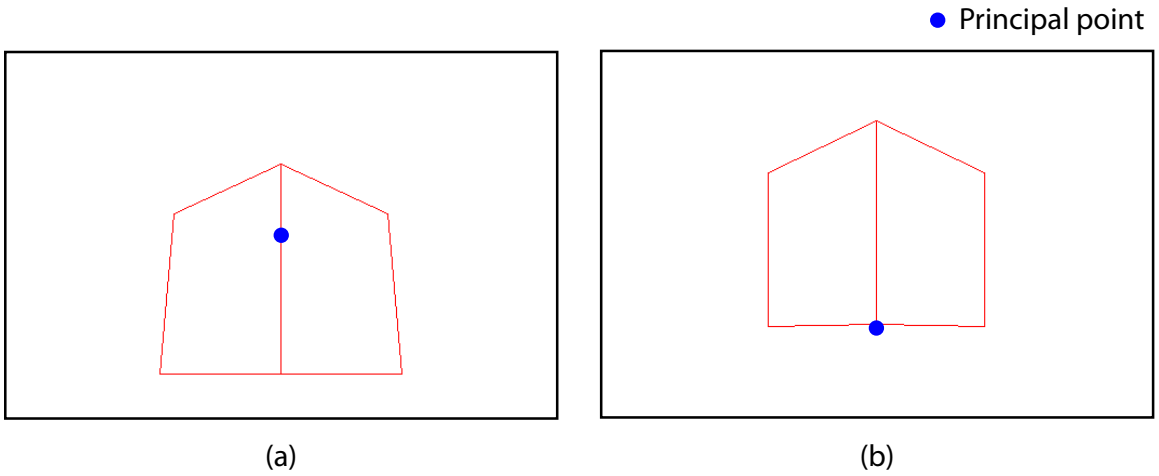


Figure 4-6: The synthetic cube images we used to test the accuracy of our estimation of principal point and camera pose. The left cube image (a) has its principal point at the image center, while the right cube image (b) moved its principal point to the image bottom.

We analyze the accuracy of our estimation of principal point and camera pose using two synthetic images. We evaluate the robustness of our estimation to user input error. Figure 4-6 shows our synthetic test cases: cube (a) has its principal point at the image center, and cube (b) has its principal point moved to the image

bottom. The cube size is 3x3x3, and the distance between the cube and the camera is around 6. The input image size is 512 $\times$ 340.

For the first test, we randomly add or subtract up-to 2 pixels to each user input for principal point estimation and pose estimation. For principal point estimation, the inputs to our principal point estimation are 18 points for three parallel lines in the three orthogonal directions. The inputs to pose estimation are 6 points. We estimate the principal point and pose 100 times and record the average error. Table 4.1 shows the result. The average viewpoint errors are 0.001 for cube (a) and 0.016 for cube (b). These are 0.02% and 0.25% of the camera distance. The average principal point errors are 0.2 pixels and 1.8 pixels respectively. These are 0.05% and 0.4% of the image size. This shows that our principal point estimation is robust against user input error.

	cube (a)	cube (b)
Viewpoint error	0.001	0.016
	0.02% of the camera distance	0.25% of the camera distance
Principal point error	0.2 pixels	1.8 pixels
	0.05% of the image size	0.4% of the image size

Table 4.1: Analysis of the robustness of our estimation against the user input error. Small errors show that our principal point estimation is robust against the user input error.

For the second test, we add errors to the 3D coordinates used for the non-linear optimization in addition to the user input error. We compare two cases: (1) the principal point is constrained by our estimation method using vanishing points, and (2) the principal point is estimated relying on Levenberg-Marquardt non-linear optimization. Table 4.2 shows the result. With our vanishing point estimation, the average errors of the estimated principal points are 17 pixels for cube (a) and 13 pixels for cube (b), and the average viewpoint errors were 0.26 and 0.24 . These are 4% and 3% of the image size, and 3% of the camera distance. In contrast, if we only rely on Levenberg-Marquardt non-linear optimization to estimate the principal point and the viewpoint, the principal point errors are 153 pixels and 126 pixels on average respectively. These are 36% and 30% of the image size, more

than 9 times larger than the errors using vanishing points. The average viewpoint errors are 3.75 and 3.8 respectively. These are almost 50% of the camera distance. Levenberg-Marquardt nonlinear optimization is a local descent approach and relies on good initialization. When significant measurement noise is present in the initialization, it might converge to a wrong local minimum. In addition, the projection errors are not discriminative enough to determine the viewpoint and the principal point at the same time. There exist ambiguities between changing the principal point and moving the camera. This is reduced by the vanishing point method.

	Principal point error		Viewpoint error	
	cube (a)	cube (b)	cube (a)	cube (b)
WITH our principal point constraint	17 pixels	13 pixels	0.26	0.24
	$< 4\%$ of the image size		3% of the camera distance	
NO principal point constraint	153 pixels	126 pixels	3.75	3.80
	$> 30\%$ of the image size		$> 50\%$ of the camera distance	

Table 4.2: Our principal point constraint using vanishing point estimation enables an accurate estimation of the viewpoint.

Finally we analyze the effect of varying the focal length while changing the camera distance. As a result, the size of the projected cube stays the same, but camera rotation and principal point modification become harder to disambiguate. The focal lengths used are 400, 600, 800, and 1000. 400 is equivalent to 20mm, and 1000 is equivalent to 50mm. The errors increase as the focal length and the camera distance increase. The principal point errors are 13, 27, 45, and 66 pixels respectively. These are 3%, 6%, 11%, and 15% of the image size. The viewpoint errors are 0.4, 0.6, 1.15, and 1.86. These are 5%, 5%, 7%, and 9% of the camera distance. The more we zoom, the more ambiguous the estimation becomes. This is related to the fact that the projection error is less discriminative for a photo taken by a telephoto lens, because the effect of a 3D rotation and that of a 2D translation become similar.

In our preprocessing, we reconstruct the scene structure using the first and second views. With respect to the known 3D structure, we compute the pose and

focal length of the reference camera using Levenberg-Marquardt (LM) nonlinear optimization. The input to the nonlinear optimization includes 2D-3D correspondences. In the case of an old reference photograph, we ask the user to clicks on 6-8 matches. This preprocessing outputs the relative camera pose between the first frame and the reference, which will be used throughout our main estimation.

4.3 Robust Camera Pose Estimation

In our robust estimation process, we estimate the camera pose relative to the first frame instead of the reference. Due to large appearance differences, it is difficult even for humans to find matches between the reference and new photos. Since we know the reference camera location relative to the first view $[R_{10}|T_{10}]$, we can derive the relative pose between the current and reference photos. For each frame n, we compute the relative camera pose $[R_{1n}|T_{1n}]$ between the first and the current frame. The translational component T_{0n} of the relative camera pose between the reference image and the current scene is

$$T_{0n} = T_{1n} - R_{1n} * R_{10}^{\top} * T_{10}. \tag{4.1}$$

In our full pipeline, we interleave this robust estimation with a lightweight estimation. We present details in Section 4.4.

4.3.1 Correspondence Estimation

To find matches between the first and current frames, we use SIFT [6] feature points. SIFT is designed to be invariant to scale changes and linear brightness changes. It is also partially invariant to viewpoint changes. For speed, we use a GPU implementation [73]. Input images have around one megapixels and we downsample images by two times for speed-up. For the downsampled images, SIFT detects around one thousand feature points. We use an approximate searching method, ANN [74] to find correspondences. We set the threshold of the second

ratio test [6] quite strict at 0.6 to keep only trustworthy correspondences.

4.3.2 Essential Matrix Estimation

We compute relative camera pose between the first view and the current frame. Since we calibrate the user's camera using the Camera Calibration Toolbox for Matlab [70] as preprocessing, we only need to estimate the essential matrix that relates the calibrated images. We use Stewénius's five-point algorithm [7], which estimates the essential matrix between two calibrated cameras in real-time. We run MSAC (m-estimator sample consensus) to find inliers and the best fitting essential matrix. MSAC is similar to RANSAC, but it modifies the cost function that outliers are given a fixed penalty while inliers are scored on how well they fit the data. The accuracy of MSAC is close to MLESAC (maximum likelihood consensus) without the loss of speed [75]. We fix the number of iterations at 1000. We determine inliers and the best fitting essential matrix using the symmetric epipolar distance [13]. Our threshold is 0.01 considering that we use normalized point coordinates.

4.3.3 Scale Estimation

We compute the scale of the translational component of the relative pose by comparing the 3D reconstructed in the current frame and the 3D reconstructed in the pre-processing. We measure the scale of the 3D world based on the median distance from the camera to the point clouds. The scale of the world is inversely proportional to the camera distance. We scale the translational vector accordingly. This scaling makes the length of our arrow visualization meaningful and consistent across frames.

4.3.4 Rotation Stabilization

Users can use our alignment visualization to confirm that they have reached the desired viewpoint. We automatically resolve the camera's rotational difference in

the alignment visualization. We warp the current scene using an infinite homography [13]. The infinite homography is more restricted than a general homography. It assumes square pixels. Our alignment visualization becomes useful when the user is close to the desired viewpoint. It allows users to focus on translating the camera in the right direction without striving to hold the camera in the right orientation. We use Brown et al.'s algorithm [76] to compute the infinite homography. We find the infinite homography that fits all the epipolar geometry inliers with the least square error.

4.4 Real-time Camera Pose Estimation

We want to provide robust results but interact with users in real-time. Our robust estimation generates reliable results but its computation is expensive and takes seconds. To provide real-time feedback, we interleave our robust estimation with a lightweight estimation, which is not as robust but inexpensive. In our lightweight estimation, we do not update the correspondences but track the most recent set of inliers using feature tracking and recompute the relative camera pose in one iteration.

We use Birchfield's KLT implementation [77] to track feature points. It contains the affine consistency check [67] and performs a multiscale tracking where it refines the feature point locations from coarse to fine resolution.

Our robust estimation and lightweight estimation are interleaved as shown in Figure 4-7. Our robust estimation detects feature points, finds matches, and estimates a new set of inliers and an epipolar geometry using robust statistics. This takes around two seconds while our lightweight estimation runs at more than 10 frames per second. This interleaved process allows the accuracy of the inliers to be preserved and provides users with a real-time update.

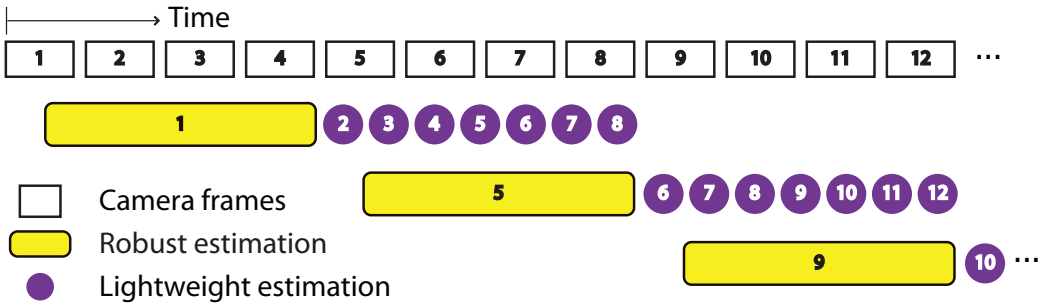


Figure 4-7: Our interleaved pipeline. Our robust estimation and lightweight estimation are interleaved using three threads: one communicates with the camera, the other conducts our robust estimation, and another performs our lightweight estimation. At the end of each robust estimation, a set of inliers is passed to the lightweight estimation thread. Numbers in this figure indicate the camera frame numbers. Note that for simplicity, this figure shows fewer frames processed per robust estimation.

4.4.1 Interleaved Scheme

Our interleaved pipeline is implemented as three threads: one communicates with the camera, the other conducts our robust estimation, and another performs our lightweight estimation. At the end of each robust estimation, a set of inliers is passed to the lightweight estimation thread. We store subsequent fames of the key frame where the robust estimation computes inliers. When the light estimation is refreshed with a inlier set from the robust estimation, it starts tracking from the next frame of the key frame instead of the current camera frame. Since the lightweight estimation uses optical flow to track points, there should not be a large gap between the key frame where the inliers are computed and the first frame where tracking starts. When the inlier set is refreshed with new robust estimation result; users can observe a one-second delay. However, this is negligible compared to the whole rephotographying process, and it does not affect the user performance or resulting rephoto quality. Our interleaved version operates as in Figure 4-8.

4.4.2 Sanity Testing

For each resulting pose, we examine three sanity tests to make sure our visualization is reliable.

0. Register the reference camera
1. Robust estimation starts. Estimate correspondences.
2. Estimate camera pose.
3. Estimate the scale of the translation.
4. Check if the robust estimation result passes sanity testing.
   If yes, proceed to the next step. Otherwise repeat from Step 1.
5. Visualize the direction to move. The robust estimation ends.
6. Multi-threading starts. Thread A repeats robust estimation
   from Step 1, while Thread B performs a lightweight estimation.
7. Thread B tracks inliers found in Step 2 and estimates camera
   pose using only one iteration.
8. Estimate the scale of the translation.
9. Check if the lightweight estimation result passes sanity testing.
   If yes, proceed to the next step.Otherwise repeat from Step 7.
10. Visualize the direction to move.
11. Repeat from Step 7 until Thread A finishes Step 5 and updates
    the set of inliers.

Figure 4-8: The flow chart of our interleaved scheme.

We compare the 3D structure reconstructed from each frame with our initial 3D reconstruction from the first two images. We measure the 3D error of all points and ignore the pose estimation if the median of the 3D error is more than $10\ \%$. Most of the time, the median error is less than $5\ \%$.

In addition, we check if the current camera pose result is consistent with previous ones. We found that a simple filter works, although the Kalman filter [78] would generate a good result as well. We measure the mean and the standard deviation of the camera locations at the previous ten frames and confirm that the current estimated camera location is within 4 standard deviations from the mean. We assume the camera motion is smooth and the pose variation is small. The above two tests typically detect a wrong answer once in hundreds frames.

Finally, we test a structure degeneracy where the inliers are all from one plane. We find the best fitting homography using RANSAC with 1.5 pixel average mapping errors within 500 iterations. If the number of homography inliers is more than 70 % of the epipolar geometry inliers, we ignore the pose estimation result. Since we use a large-enough baseline, this error does not occur in general.

When our estimation result fails to pass the above tests, we simply do not update the visualization. Since wrong answers do not occur often, this does not affect the user experience too much.

4.5 Visualization

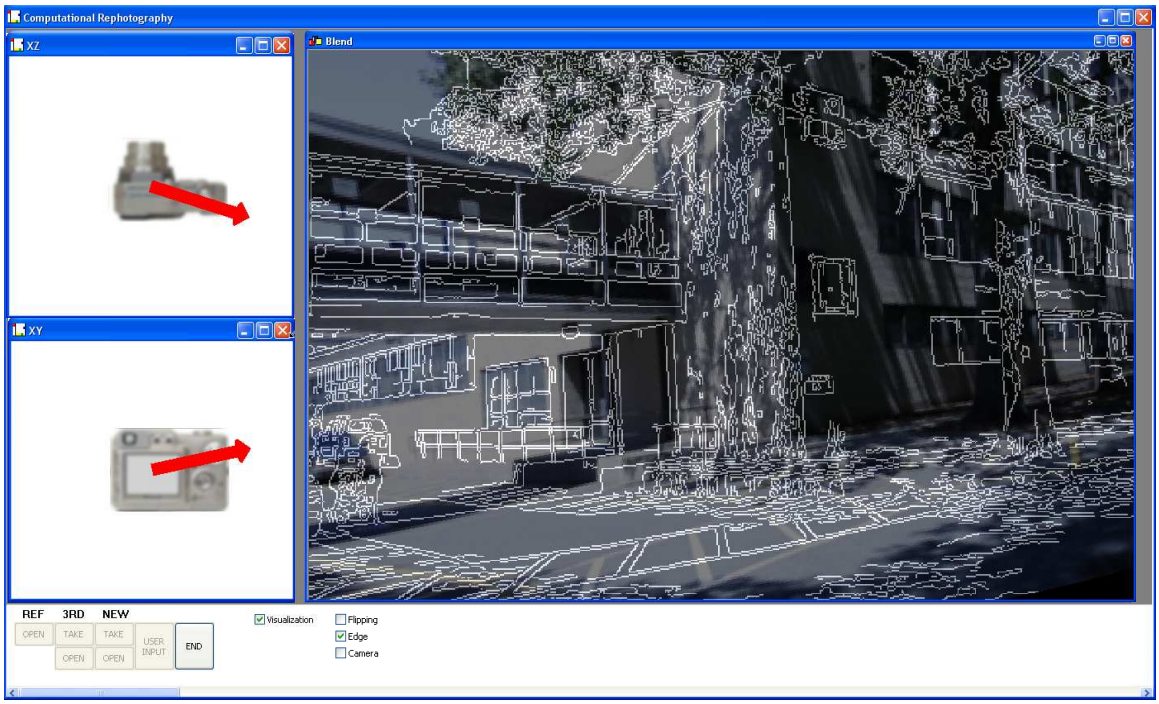


Figure 4-9: The screen capture of our visualization. It includes two 2D arrows and an edge visualization. The primary visualization is the two 2D arrows. The top arrow is the direction seen from the top view and the bottom arrow is perpendicular to the optical axis. An edge visualization is next to the arrow window. The user can confirm that he has reached the desired viewpoint when the edges extracted from the reference are aligned with the rephoto result in our edge visualization. We show a linear blend of the edges of the reference image and the current scene after homography warping.

Comparing the reference and current image side by side does not provide precise information about viewpoint difference. In our pilot user study, we provided a linear-blend of the reference and current image, and users could not estimate the desired viewpoint by examining the pixel difference. In a subsequent test, we showed the relative pose information in 3D (See Fig. 4-10(a).) Still we found that

it was hard for users to interpret 3D information. In our final visualization design, we visualize the relative camera pose in two 2D planes: one is the direction seen from the top view and the other is perpendicular to the optical axis, as shown in figure 4-9. In our final user studies, users found our arrow visualization easy to learn and follow.

In addition, we visualize the alignment between the reference and current photos to let users to refine and confirm the final viewpoint. Due to the large appearance differences, a linear-blend of the old reference photo and the current scene does not show whether they are aligned or not. We experimented with three visualizations: an edge visualization, a flipping visualization, and a visualization with a reference camera projected onto the current frame.

In an edge visualization, we overlay the edges extracted from the reference image over the current frame. In a flipping visualization, users can flip between the reference photo and the current frame. In both edge and flipping visualizations, we warp the current frame using a best-fitting infinite homography [76, 13], which is not useful for big parallax but good for small translation. As the translational component becomes zero, the rotational component is resolved by homographies. Finally, we project the reference camera onto the current frame.

During user studies, we let users to choose among three visualization. All the users chose the edge visualization to provide them feedback. Users used the flipping visualization only at the final viewpoint to confirm the viewpoint. Users did not find the projected reference camera useful.

Figure 4-9 shows our final visualization design. Our explicit arrow visualization guides users how to move the camera. At the final viewpoint, they can use our edge visualization to refine and confirm the viewpoint. In addition, we leave the other two alignment visualization techniques as options.

4.6 Results

In our prototype, we estimate relative pose using the output we get from the camera viewfinder. We use a Canon 1D Mark III live view, which outputs 5-10 frames per second. Each robust estimation takes about 2 seconds on a 2.4GHz laptop with NVIDIA GeForce 8600M GT, while a lightweight estimation tracks inliers, estimates the relative pose, and visualizes the arrows at 10-20 frames per second. With multi-threading, GPU-SIFT takes one second and the approximate nearest neighbor (ANN) takes one second.

4.6.1 Evaluation

We have performed multiple pilot user studies before finalizing the design of our user interface.

First pilot user study

In our first pilot user study, we wanted to test whether humans would be able to estimate the viewpoint differences by simply comparing two photos.

Procedure We asked users to estimate the viewpoint of a reference photo by comparing it with the output of the camera viewfinder, while they moved the camera. We provided two users with three different visualization techniques: the reference and current image side by side, a linear-blend of the reference and current image, and a color coded linear blend of the reference in red and current image in blue. We asked questions to users at the end.

Results and conclusions Comparing the reference and current image side by side did not seem to provide information about viewpoint differences. Although users preferred the linear-blend among three visualization, still users could not estimate the desired viewpoint by examining parallax. This leads to our first visualization design.

Second pilot user study

In our first visualization design, we showed the relative pose information in 3D and updated the camera pyramid every 3 seconds (See Fig. 4-10(a).) In a pilot user study, we wanted to test whether users would be able to take more accurate rephotos using our 3D pose visualization than using a linear-blend visualization.

Procedure Given the reference photo taken by the same camera, we asked six users to reach the viewpoint where the reference photo was taken. Note that this was easier than an actual rephotography: there was not a large appearance difference, and users did not need to estimate the focal length. We tested 4 indoor scenes. We measured the accuracy of rephotos by comparing the pixel differences between the reference and resulting rephotos.

Results and conclusions We observed that neither our visualization nor a linear-blend visualization helped users to take accurate rephotos. Figure 4-10 shows the resulting rephotos.

In terms of the pixel differences, users made less errors with our visualization, 70% of the errors with a linear-blend. However, we realized that comparing pixel differences was not a good metric. We decided to measure the distance from the users' final camera location to the ground-truth in the next studies.

In our camera pyramid visualization, users found it hard to interpret 3D information; most had a hard time separating translation and rotation. In addition, users asked for a real-time feedback.

Third pilot user study

In the next user study, we tried to guide the user with respect to the distance to the scene (z). In addition to showing 3D camera pyramids, we told the user how far he was from the desired viewpoint relative to a scene point, but users did not find it useful. In general, users preferred a simple visualization; having one visualization

(a) Our camera pyramid visualization (b) Rephoto using our technique (c) Rephoto using a linear blend

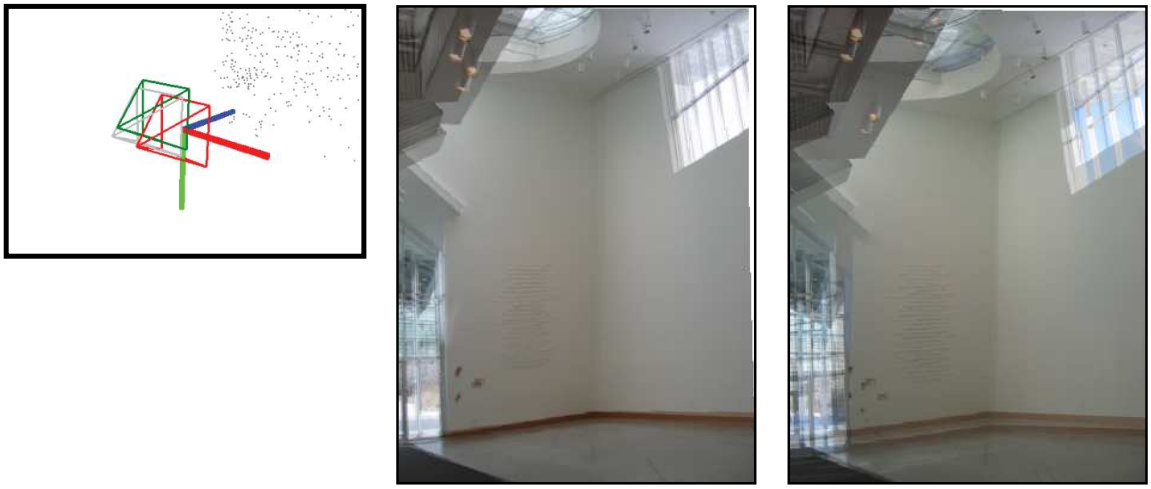


Figure 4-10: User study results with our first visualization. We displayed the relative camera pose using 3D camera pyramids (a). The red pyramid showed the reference camera, and the green one was for the current camera. Although the rephoto using our technique (b) was better than that using a linear-blend (c), neither helped users to take accurate rephotos.

window helped users focus on the task. Users became tired and lost when they had to jump between different visualization windows.

First final user study

In our final estimation and visualization, we compute relative camera pose in real-time and show the direction to move using two 2D arrows (See Fig. 4-9.) We have conducted two additional user studies to validate our technique.

In the first final user study, we wanted to compare our arrow visualization technique with a linear-blend visualization. In addition, we included the focal length estimation: users had to manually estimate the focal length with a linear-blend estimation, while our visualization automatically resolved the focal length difference.

Procedure Given the reference photo taken by a different camera, we asked four users to reach the viewpoint where the reference photo was taken within 3 mins. We tested two indoor scenes. Each participant experienced both scenes and each

	Our method	Linear blend	Remark
avg. (m)	(a) 0.47	(b) 6.03	(a)/(b) = 7.8%
std. (m)	0.14	2.83	
P-value	0.02		

Table 4.3: User study errors. With our method, users made less than 8 % of error than with a linear blend. The P-value is smaller than 0.05. This means that this result is statistically meaningful and significant.

scene paired with only a single technique for that participant. We marked the reference camera location on the map and measured the distance from the users' final camera location to the ground-truth. We did not ask users to choose the first and second viewpoints. They were fixed among all the users.

Results and conclusions Table 4.3 shows the average distance between the ground-truth and the final locations where four users took the rephotos for two test cases. With our method, users made less than 8 % of error than with a linear blend. Users found that our 2D arrows were easy to learn and follow. Figure 4-11 compared two rephoto results using both techniques. In every test case, users took more accurate rephotos with our arrow visualization than with a linear blend visualization.

Second final user study

In our final user study, we wanted to test our user interaction schemes including providing a wide-baseline, clicking on matches, and comparing two photos with large appearance differences. We compared the accuracies of the resulting rephotos using our technique with those with a naïve visualization. In particular, we sought to compare the accuracy of the viewpoint localization using our technique with those using a naïve visualization.

Procedure We compared our technique with a naïve visualization. We added appearance differences to the reference photos: we transferred the tonal aspects from old photos to the reference photos [79], as shown in figure 4-12. As a result, the reference photos had large appearance differences from current photos. Given

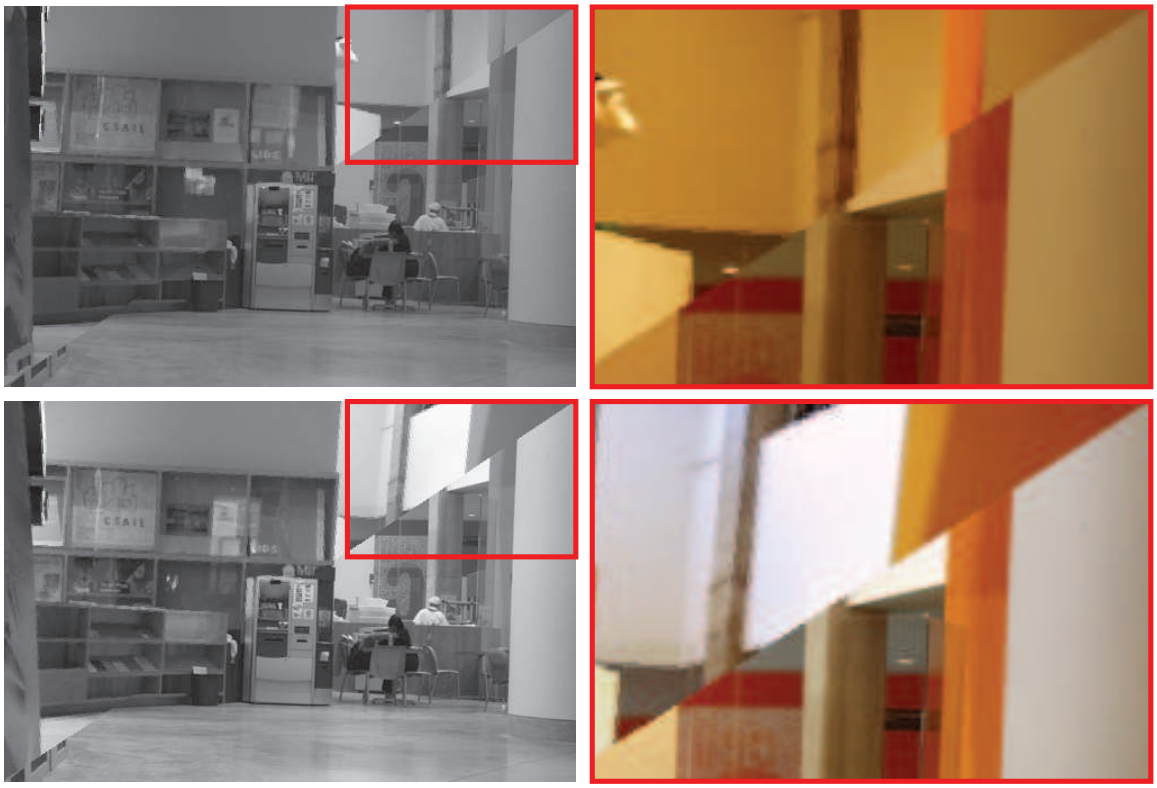


Figure 4-11: User study results in the indoor scenes. Split comparison between the reference image and users' rephotos after homography warping. The result at the top is from a user using our method, and the one at the bottom is from a user using a linear blend visualization. Notice that the result at the top is aligned better.

the reference photo taken by a different camera, we asked six users to reach the viewpoint where the reference photo was taken within 10 mins. We tested three outdoor scenes. Each participant experienced all the scenes and each scene paired with only a single technique for that participant.

For both methods, users start from the same initial location. With our technique, we only fixed the first viewpoint and asked users to choose the second viewpoint. In addition, users provided correspondences between the reference and the second frame by clicking six matches. In a naïve visualization method, we showed both linear blend and side-by-side visualization of the reference and current frame, since a linear-blend suffered from large appearance differences between the reference and current photos. Before each user study, we provided users with a quick tutorial of both methods.

Figure 4-12: In the final user study, the reference photos had old-looking appearances.

Results and conclusions In every test case, users took more accurate rephotos with our arrow visualization than with a naïve visualization with a linear blend and a side-by-side. Figure 4-13 and 4-14 compare the rephotos taken with our technique and those using a naïve visualization. The remaining parallax in the rephoto results using a naïve visualization is quite large, while users resolved parallax with our technique.

Table 4.4 shows the average distance between the ground-truth and the final locations where six users took the rephotos for three test cases. The error with our method is 40% of the error with a linear blend. The distance difference became smaller than the indoor cases. In the indoor scenes, the parallax was subtle, but in

	Our method	Linear blend	Remark
avg. (m)	(a) 1.8	(b) 4.4	(b)/(a) = 2.5
std. (m)	0.6	2.9	

Table 4.4: User study errors. The error with a linear blend is 2.5 times larger than the error with our technique.

the outdoor scenes, users could notice some important cues such as that buildings are occluded or not. Still many people could not figure out how to move the camera to resolve the parallax. With a naïve blend, users had to estimate the location and focal length of the reference camera by themselves. With our method, users needed only to follow our arrow visualization while our technique automatically estimated the location and focal length of the reference camera.

Figure 4-13: User study results. Left to right: (a) the reference photo and users' rephotos using our technique (b) and a naïve visualization (c) after homography warping. The second row show split comparison between the reference image and users' rephotos, and the last row has their zoomed-in. With our method, users took more accurate rephotos.

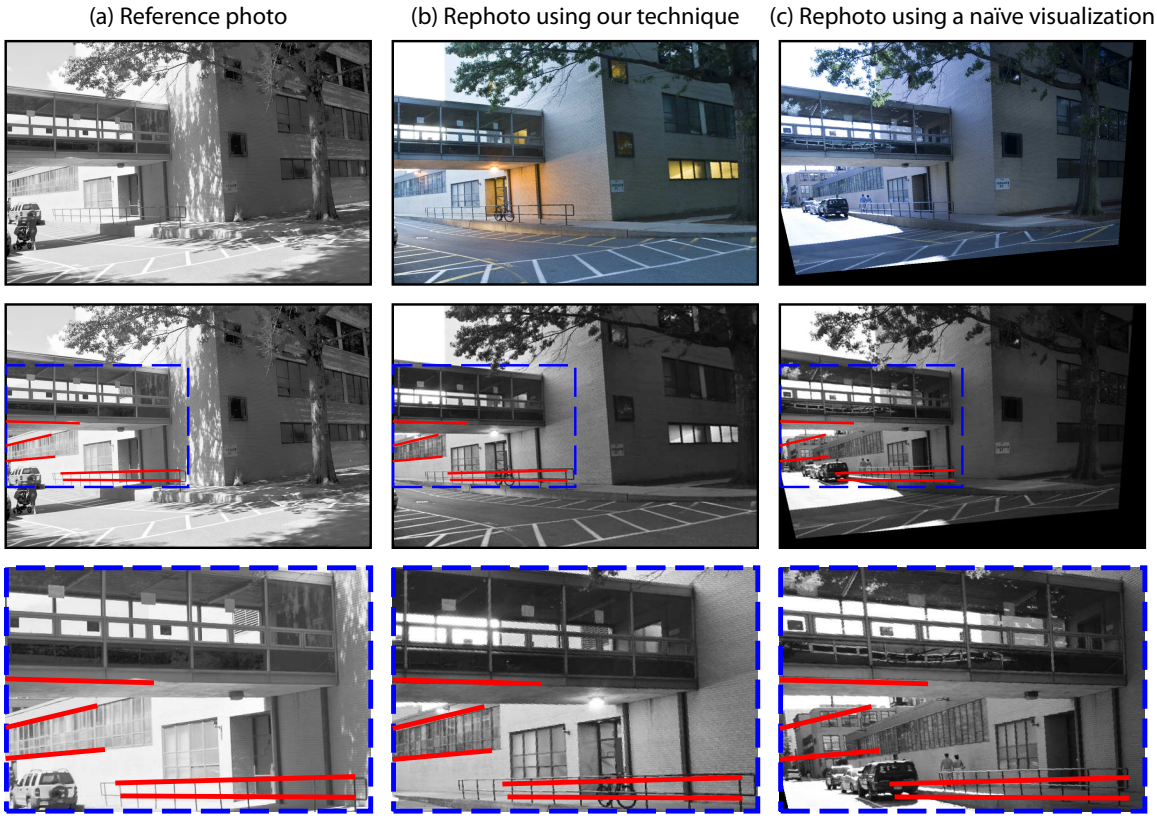


Figure 4-14: User study results. Left to right: (a) the reference photo and users' rephotos using our technique (b) and a naïve visualization (c) after homography warping. The last row shows two times zoomed-in blend of rephotos and outline features from (a) in red. The outline features from (a) match outline features in (b) but not in (c). This shows that users take more accurate rephotos using our technique than using a naïve visualization.

(a) Reference photos (b) Our rephoto results (c) Professional rephotos without our technique

Figure 4-15: Results. Left to right: the reference images, our rephoto results, and professional manual rephotos without our method. This shows the accuracy of our results.

4.6.2 Results on historical photographs

Figure 4-15, 4-16, 4-17, and 4-18 show our rephoto results of historical photographs taken by unknown cameras. It usually took 15-30 minutes to reach the desired viewpoint. This took a long time because we had to walk 50-100m with the laptop and tripod and cross roads. In figure 4-15, a zoomed in linear blend shows the accuracy of our results. In Figure 4-18, we apply style transfer from the reference to the rephotos [79]. By matching the tonal aspects, it becomes even more evident which scene elements are preserved and which have changed across time. Faithful rephotos reveal the change of roofs, windows, and neighborhood.

(a) Reference photos (b) Our rephoto results (c) Zoomed-in blend of (b) and features from (a)

Figure 4-16: Results. Left to right: the reference images, our rephoto results, three times zoomed-in blend of (b) and outline features from (a) in red. The outline features from (a) match outline features in (b). This shows the accuracy of our results.

4.6.3 Discussion

The bulk of the laptop currently limits portability and we hope that open digital cameras with additional processing power will enable rephotography directly from the camera.

Our relative pose estimation works best when there is sufficient parallax between the images. When nearing the viewpoint, the user must rely on the alignment blend, which limits final precision. Our technique requires a reasonable number of feature points (around 20) and suffers from uniform targets. The scene must present enough 3D structure to make viewpoint estimation well posed. If the scene is mostly planar, a homography can match any pair of views and the viewpoint

cannot be inferred.

We share a number of limitations with traditional photography: if the desired viewpoint is not available or the scene is occluded and cannot be seen at the desired viewpoint, rephotography is impossible. Nevertheless, our technique can still help users realize that the viewpoint is no longer available.

Audio feedback is a natural extension to our visualization that we can explore in the future.

(a) Reference photos (b) Our rephoto results (c) Split comparison: (a) and (b)

Figure 4-17: Results. Left to right: the reference photos, our rephoto result, split comparison between the reference and our rephoto. This shows that our technique enables users to take a faithful rephoto.

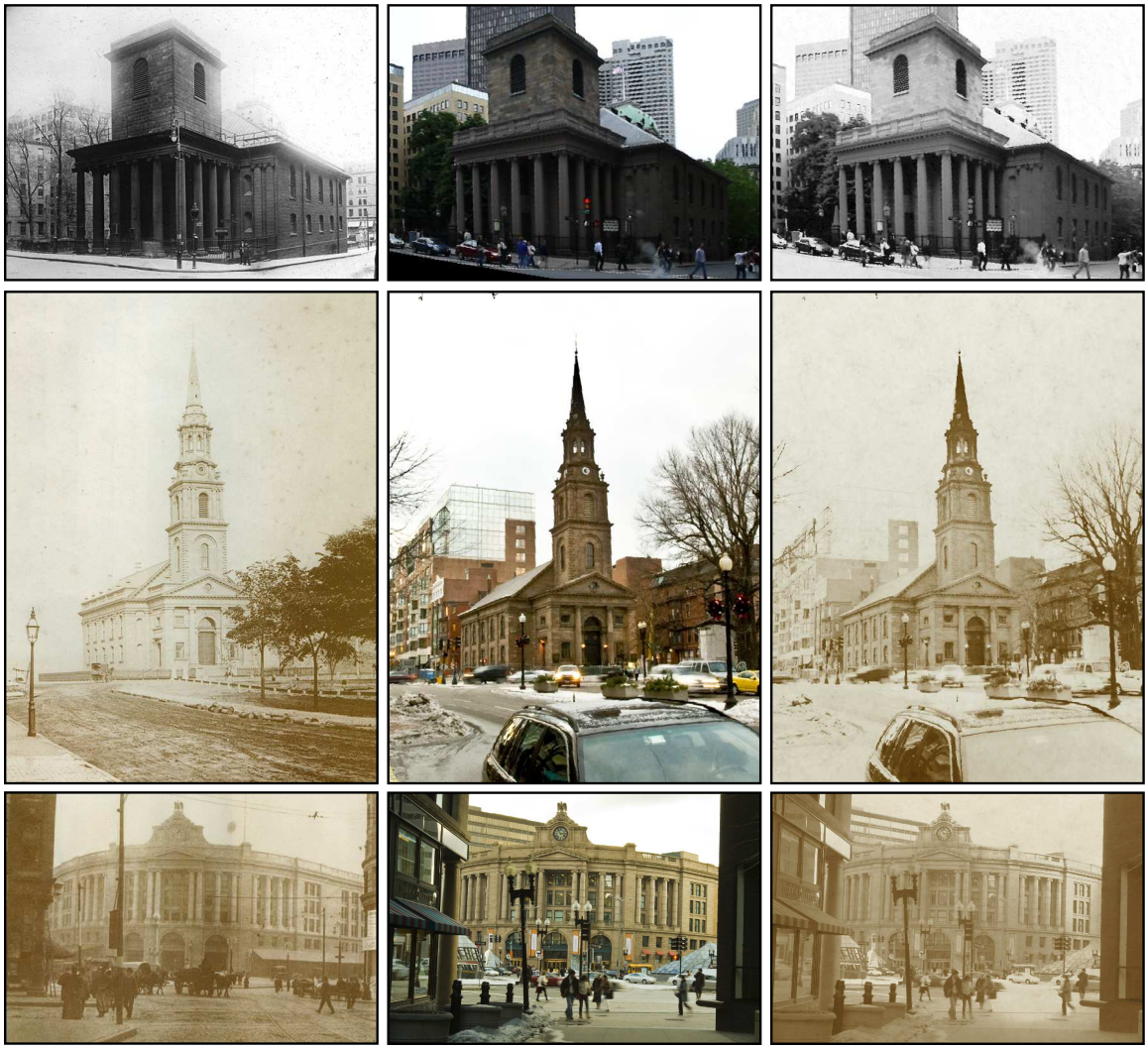


Figure 4-18: Results with style transfer. Left to right: the reference photos, our rephoto results, and our rephotos with styles transferred from the reference photos.

4.7 Conclusions

We have shown a technique that allows users to reach a target location and take rephotographs with ease. We make this real-time guidance possible by interleaving a robust pose estimation with a lightweight real-time estimation. We visualize the direction to move using two 2D arrows, which users found intuitive and easy to follow. We believe that computational techniques open exciting possibilities for user assistance through scene analysis directly on the camera, such as our computational rephotography.

Our robust estimation relies on a wide-baseline 3D reconstruction. Having a wide baseline between the first-vie and the current frame makes pose estimation more robust and removes the need to resolve motion degeneracy cases. The reconstructed 3D from the first- and second-view is used to estimate the focal length of the reference photo. In addition, the 3D reconstruction allows for consistent and meaningful scale estimation across frames, which is important for consistent visualization.

Chapter 5

Style Transfer

Photographers seek to obtain a certain "look" for their pictures to convey a mood or an æsthetic. This is particularly significant for black-and-white photography where strikingly distinctive styles can be achieved. Our technique offers direct control over the "look" of an image by transferring style from a model photograph onto an input one. Our method is based on a two-scale non-linear decomposition of an image and handles global and local contrast separately. This is inspired by traditional photography, where the darkroom offers remarkable global and local control over the brightness, contrast, and sharpness of images via a combination of chemical and optical processes [4, 9].

We decompose both input and model into two layers and modify each layer according to its histogram. To transfer the spatial variation of local contrast, we introduce a new edge-preserving textureness that measures the amount of local contrast. We recombine the two layers using a constrained Poisson reconstruction. Finally, additional effects such as soft focus, grain and toning complete our look transfer.

Our main contribution is to transfer photographic look between images. Our results demonstrate the relevance and robustness of the features we manipulate. In addition, our method provides direct control through the curve interface, which is equally powerful, though perhaps more suited to advanced users.

5.1 Image Statistics

Before we introducing our work, let us discuss the relevance of frequency contents to photographic look. Statistical characterization has been used to measure regularities and differences among so-called "natural" images. Statistical characteristics of natural images are often measured in the frequency domain. Among natural images, Field found scale invariance that the average amplitude falls as " $\frac{1}{f}$ " [25, 26]. We observe that this statistical regularity is mostly shared by casual photographs, but not by artistic photographs. Figure 5-1 shows the normalized average amplitude across scale. That is, each average amplitude is multiplied by its frequency. Natural images observe the scale invariant property, while artistic photographs do not show such invariance.

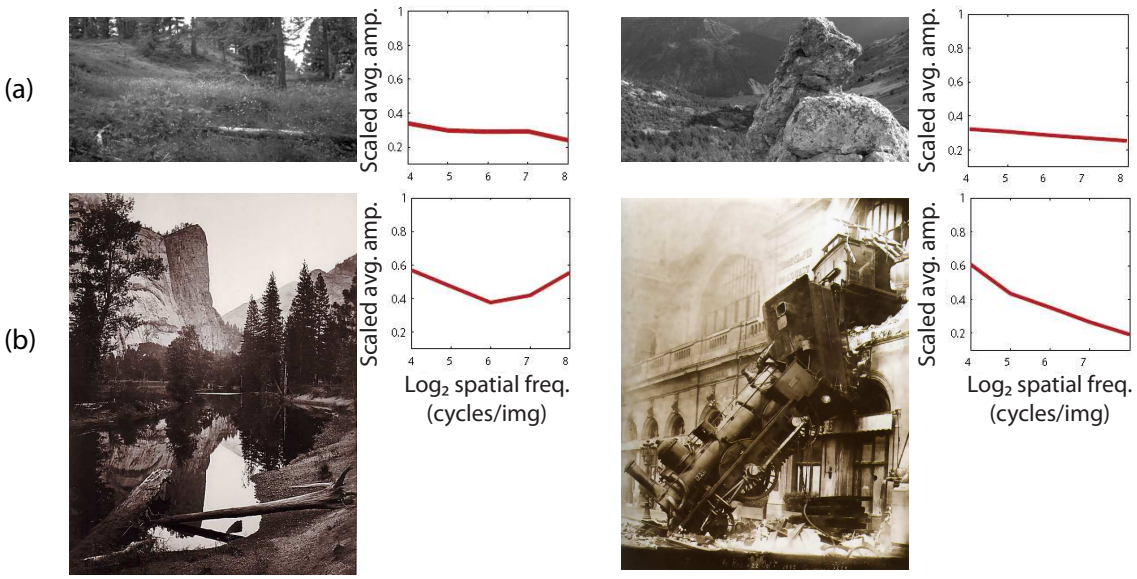


Figure 5-1: The normalized average amplitude across scales: on the left is the low frequency and the right is the high frequency. (a) The average amplitude in casual photographs is uniformly distributed across frequencies. (b) The average amplitude in the artistic photographs shows a unique distribution with a U shape or a high slope.)

In addition, image statistics are non-stationary, and local statistical features vary with spatial location [27]. Frequency content is useful to show the spatial

distribution of local contrast. Figure 5-2 shows windows of the normalized amplitude spectra. The local spectral signatures are obtained by taking normalized spectrum of each window, where we remove boundary effects by using a smooth windowing function. We use a hamming window in practice. The degree of the non-stationarity varies in different images. In this work, we use the scale variance and the variations in non-stationarity to analyze and transfer photographic styles.

5.2 Overview

Image statistics and traditional photographic printing suggest that aspects such as the intensity distribution at different scales, spatial variations, and the amount and distribution of detail are critical to the look of a photograph. This inspires our use of a two-scale decomposition to control global contrast and the spatial variation of local contrast. We quantify the look of an image using histograms over this decomposition, which affords both interactive control using a curve interface, and the ability to automatically transfer visual properties between images. In the latter, histograms of the components of a model image are forced upon a new input. Because we explore strong stylistic variations, we tend to perform larger modifications to the input than tone mapping. In particular, some looks require an increase in local contrast, which can produce halos if traditional techniques are used. We introduce a gradient constraint that prevents undesirable modifications. Finally, we post-process the image to achieve various effects such as soft focus, paper grain, and toning. Figure 5-3 summarizes our pipeline.

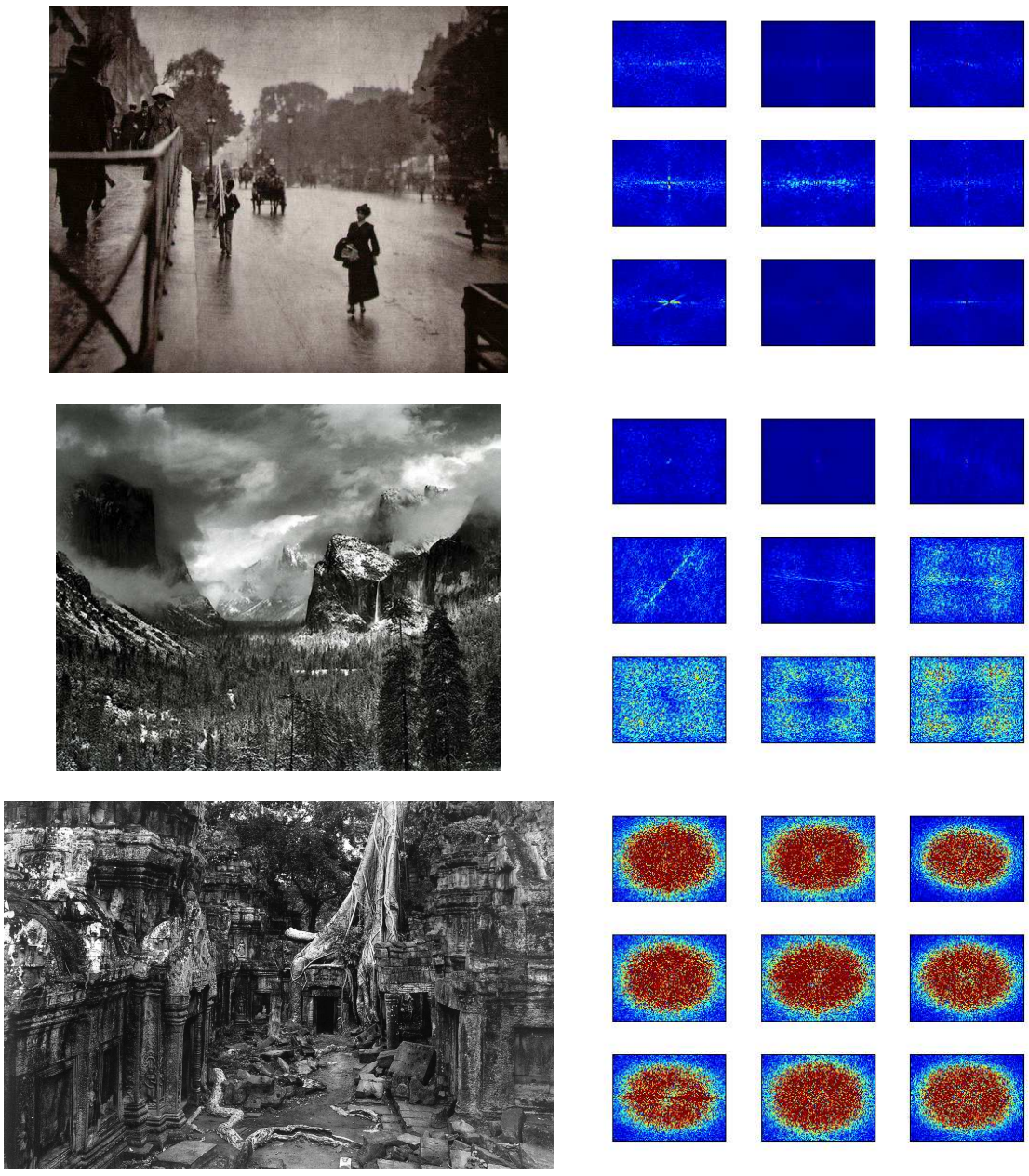


Figure 5-2: The local spectral signatures show non-stationarity. The local spectral signatures are obtained by taking normalized spectrum of each window : each amplitude is multiplied by its frequency. The degree of the non-stationarity varies in different images. The color close to red means a high value and that close to blue means a low value.

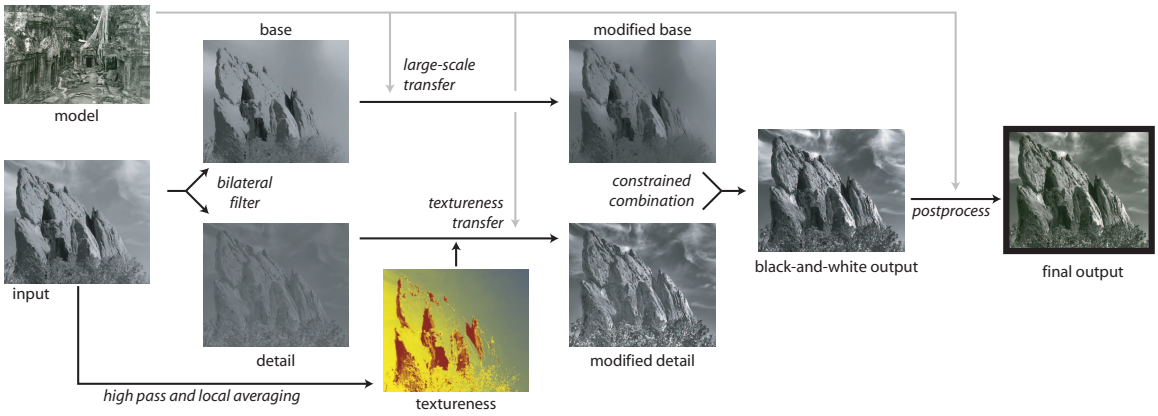


Figure 5-3: Overview of our pipeline. The input image is first split into base and detail layers using bilateral filtering. We use these layers to enforce statistics on low and high frequencies. To evaluate the texture degree of the image, we introduce the notion of *textureness*. The layers are then recombined and post-processed to produce the final output. The model is Kenro Izu's masterpiece shown in Figure 2-8b.

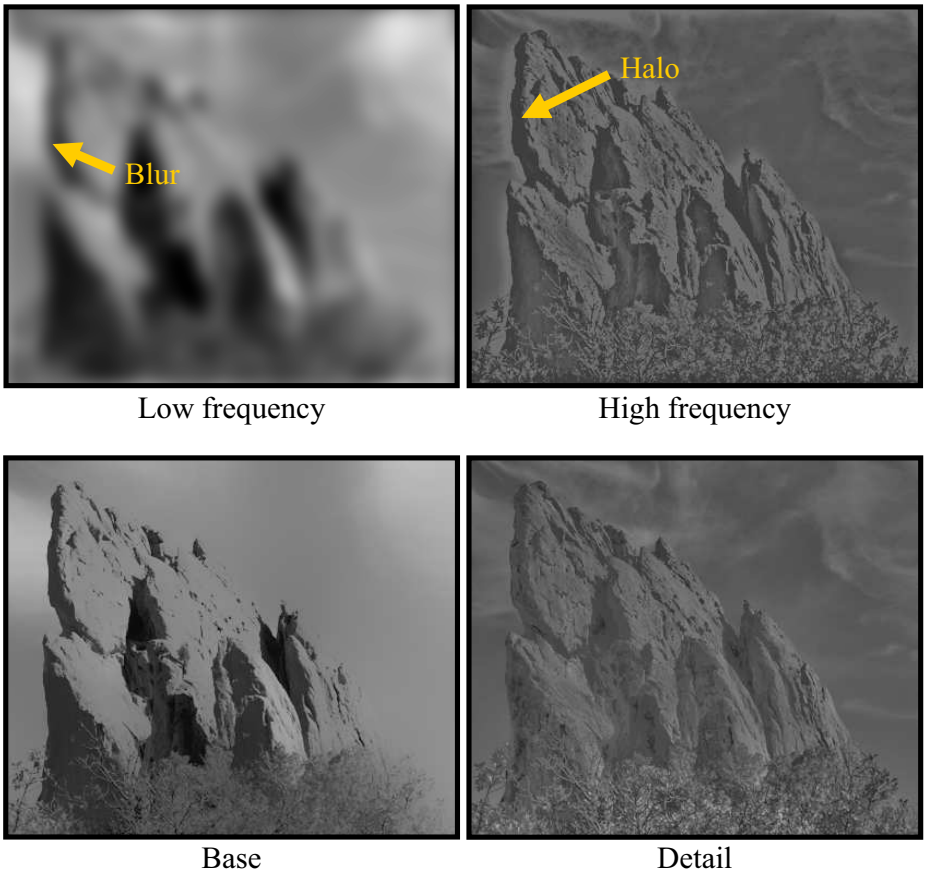


Figure 5-4: The output of the Gaussian blur contains low frequency contents, and the residual has high frequency components. However, this linear filtering results in haloes and artifacts around edges. In contrast, the output of the bilateral filter (base) and its residual (detail) preserve the edge information.

5.3 Edge-preserving Decomposition

To manipulate global and local contrast separately, we decompose both input and model photographs into two layers. We want one layer to contain global contrast and the other layer to characterize local contrast. A naïve solution is to use a Gaussian blur. The filtered output contains global contrast, and the residual has local contrast. However, a linear filter such as a Gaussian blur does not preserve edges, as shown in Figure 5-4. This results in haloes and artifacts in the result as we modify each layer independently.

Therefore, we use an edge-preserving filter, the bilateral filter, which Durand and Dorsey [32] use for the tone mapping. The bilateral filter smoothes the image

everywhere except at strong edges. This prevents haloes and artifacts. We use the output and the residual of the bilateral filter to control global and local contrast respectively. The output is called the base layer, and the residual is called the detail layer [32].

With $g_\sigma(x) = \exp(-x^2/\sigma^2)$, a Gaussian function, the bilateral filter of image I at pixel $\mathbf{p}$ is defined by:

$$bf(I)_\mathbf{p} = \frac{1}{k} \sum_{\mathbf{q} \in I} g_{\sigma_\mathrm{s}}(\|\mathbf{p} - \mathbf{q}\|)\ g_{\sigma_\mathrm{r}}(|I_\mathbf{p} - I_\mathbf{q}|)\ I_\mathbf{q} \tag{5.1a}$$

$$\text{with:} \quad k = \sum_{\mathbf{q} \in I} g_{\sigma_\mathrm{s}}(\|\mathbf{p} - \mathbf{q}\|)\ g_{\sigma_\mathrm{r}}(|I_\mathbf{p} - I_\mathbf{q}|) \tag{5.1b}$$

The choice of σ_s and σ_r is crucial. σ_s controls the spatial neighborhood. We find that $\sigma_\mathrm{s} = \min(\text{width}, \text{height})/16$ consistently produces good results. σ_r determines the influence of the range difference, and it should differentiate important edges from textures. We rely on the gradient norm to estimate the edge amplitude in the input. $\sigma_\mathrm{r} = p_{90}(\|\nabla I\|)$ achieves consistently good results, where p_{90} denotes the 90th percentile. These settings are robust to spatial and intensity scales.

Since contrast is a multiplicative effect, we perform our decomposition in the logarithmic domain. In practice, we use Paris's fast bilateral filter [80]. We define the base layer B and detail layer D from the input image I (where I, B and D have log values):

$$B = bf(I) \qquad \text{and} \qquad D = I - B \tag{5.2}$$

5.3.1 Gradient Reversal Removal

Durand and Dorsey [32] note that artifacts can occur when edges are not sharp. They introduce a "fix" that detects uncertain pixels and uses a smoothed base layer, but they highlight that this solution is not entirely satisfying. The problem is more acute in our case because we may *increase* the amount of detail (by a factor as high

as 6 in some examples), which requires a reliable halo-free detail layer.

We address this by directly constraining the gradient of the decomposition to prevent reversal. We force the detail derivatives $\partial D/\partial x$ and $\partial D/\partial y$ to have the same sign as the input derivatives and an amplitude no greater than them. For this, we build a gradient field $\mathbf{v} = (x_{\mathbf{v}}, y_{\mathbf{v}})$:

$$x_{\mathbf{v}} = \begin{cases} 0 & \text{if} \quad \text{sign}\left(\partial D/\partial x\right) \neq \text{sign}\left(\partial I/\partial x\right) \\ \partial I/\partial x & \text{if} \quad |\partial D/\partial x| > |\partial I/\partial x| \\ \partial D/\partial x & \text{otherwise} \end{cases} \tag{5.3}$$

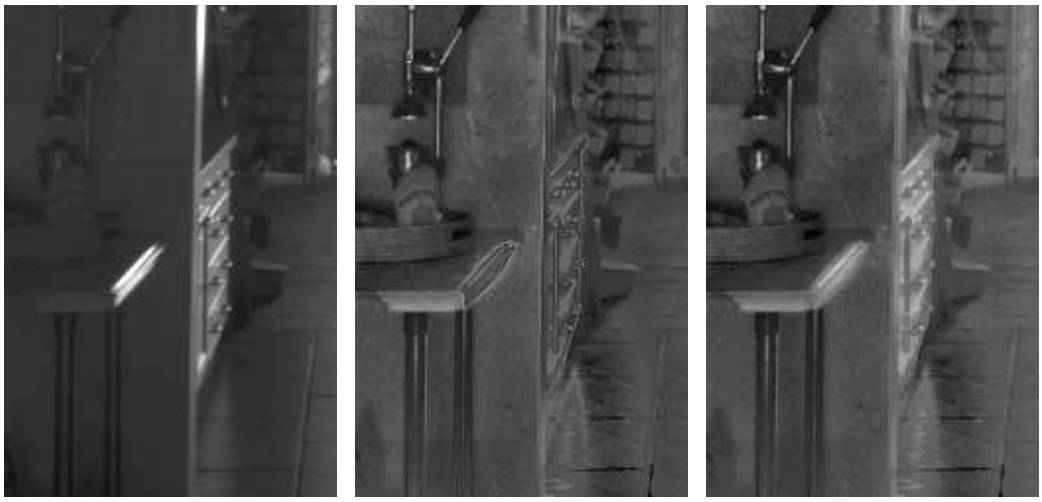
The y component $y_{\mathbf{v}}$ is defined similarly. The corrected detail layer is obtained by solving the Poisson equation (Eq. 5.4.) This builds the detail layer D with a gradient ∇D as close as possible to $\mathbf{v}$, in the least square sense.

$$\partial D/\partial t = \Delta D - \text{div}\left(\mathbf{v}\right) \tag{5.4}$$

We update the base layer accordingly: $B = I - D$. This approach results in a high-quality detail layer because it directly addresses gradient reversal and preserves other subtle variations (Fig. 5-5).

5.4 Global Contrast Analysis and Transfer

The base layer contains global contrast. We transfer global contrast using histogram matching in the base layer. We deduce a remapping curve from the input and model base histograms. We apply the remapping curve to the input base layer. Each pixel is transformed according to the remapping curve. As a result, the histogram of the output base is the same as that of the model base. Figure 5-6 illustrates this histogram matching process.



(a) input image (b) uncorrected detail (c) corrected detail

Figure 5-5: The bilateral filter can cause gradient reversals in the detail layer near smooth edges. Note the problems in the highlights (b). We force the detail gradient to have the same orientation as the input (c). Contrast is increased in (b) and (c) for clarity.

5.5 Local Contrast Analysis and Transfer

Our local contrast transfer is independent of the global contrast transfer. The main contribution of our work is a technique that manipulates the amount of high-frequency content and its spatial variation. This contrasts with tone mapping approaches that usually do not modify the detail layer.

This step involves additional challenges compared to the base transform. First, we show that the detail layer does not capture all the high frequency content of the image. Second, we need to modify the spatial variation of detail without creating artifacts. In particular, we introduce a new technique to measure and modify local frequency content in an edge-preserving manner.

5.5.1 Detail Transfer using Frequency Analysis

While the bilateral filter provides a decomposition that facilitates halo-free manipulation, the edge-preserving term g_{σ_r} results in substantial high-frequency content in the base layer (Fig. 5-7). While the choice of different parameters or more ad-

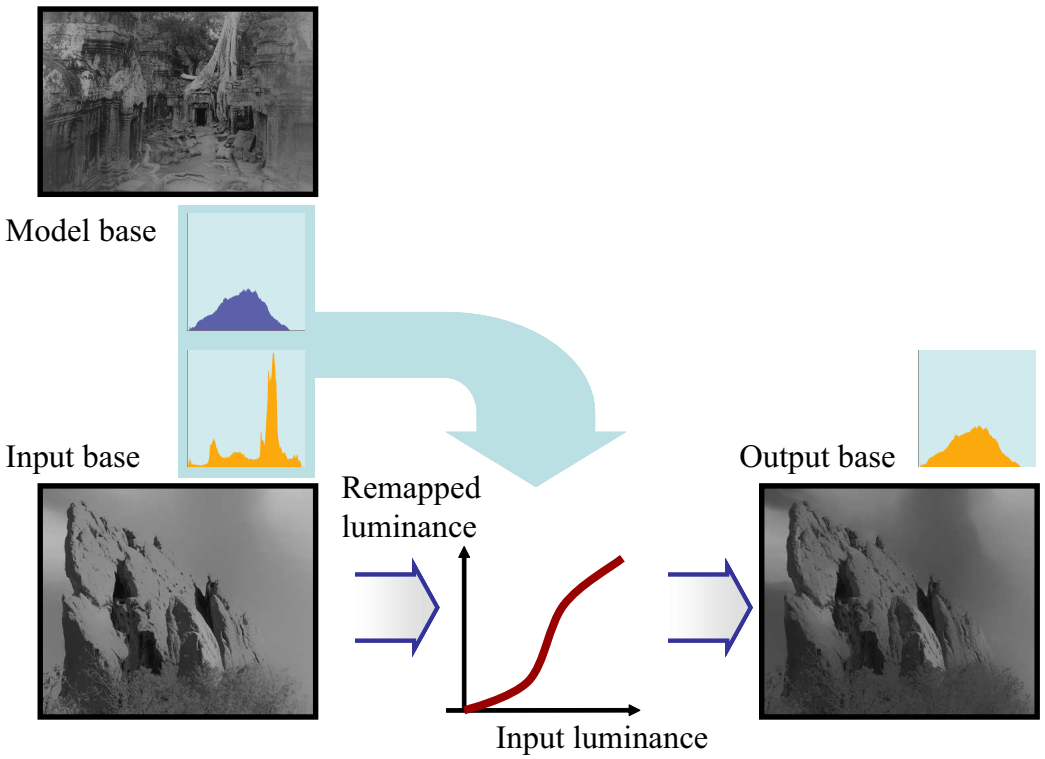


Figure 5-6: Histogram matching. The remapping curve is deduces from the input and model base histograms. Each pixel is transformed according to the remapping curve. For the remapping curve, the horizontal axis is the luminance of the input base. The vertical axis is the luminance of the output base.

vanced filters [81] can affect this issue, the very nature of such filter calls for high-frequency content in the base. In particular, the influence of the range Gaussian g_{σ_r} results in that patterns that are high-frequency but high-contrast will mostly be in the base. While this is not an issue for tone mapping where the detail is unaffected, it is critical for our detail management. On the other hand, the manipulation of the detail layer is a safe operation that does not lead to the halo artifacts caused by linear image processing.

Our solution combines linear frequency *analysis* with the *transfer* of the detail layer obtained from our nonlinear filter. We analyze the amount of texture (or high frequency) using a high pass filter applied to *both* the detail and the base layer. This ensures that all the frequency content is taken into account. We use this information to decide how the detail layer should be modified. In a nutshell, we get the best of the two approaches: reliable analysis of the high-pass filter, and the

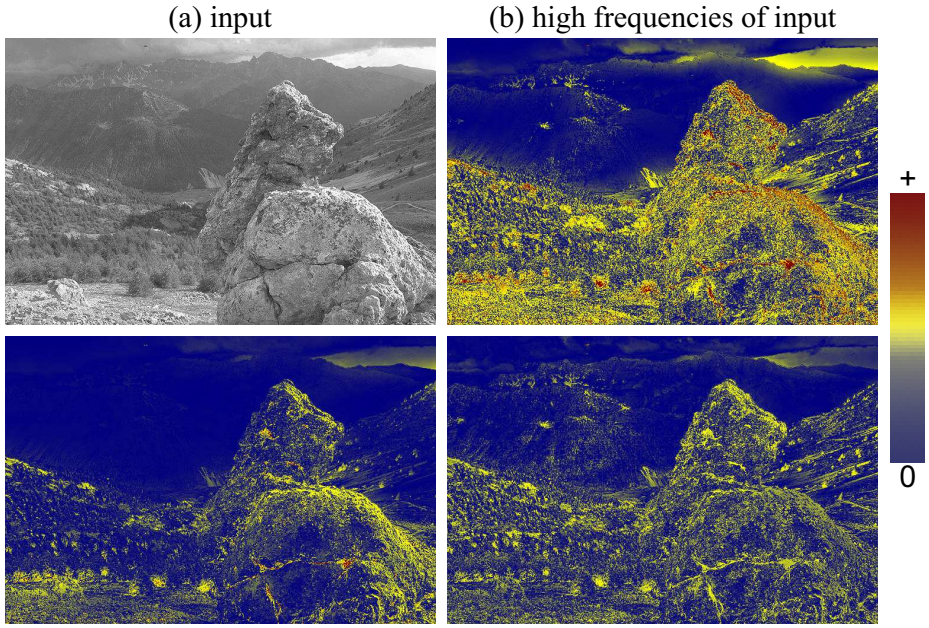


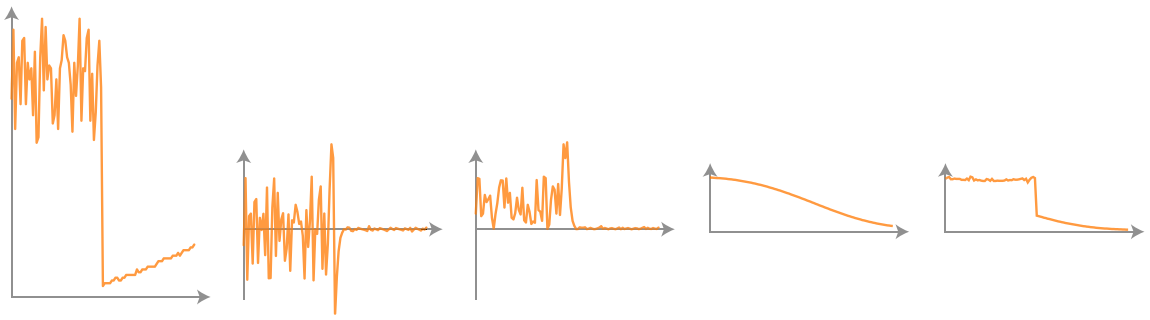
Figure 5-7: Because of the preserved edges, the high frequencies of an image (b) appear both in the base layer (c) and in the detail layer (d). This phenomenon has to be taken into account to achieve an appropriate analysis.

safe manipulation of the detail layer.

5.5.2 Textureness

We seek to characterize the spatial variation of local contrast. We build on the notion of power maps, e.g. [82] and activity map [34] where the local average of the amplitude of high frequencies is used. Figure 5-8 illustrates our computation of textureness for a 1D example where the left part has a high level of local contrast while the right part is smooth.

First, we compute a high-pass version H of the image using the same cutoff σ_s. Note that the local average of such a high-pass image is by definition zero: the low frequencies are removed. This is why we consider the *magnitude* (or absolute value) of H (Fig. 5-8c). Power maps or activity maps are then defined as the local average – obtained via low-pass filtering – of this magnitude (Fig. 5-8d). Such maps provide good characterization of highly-textured vs. smooth regions and



(a) input I (b) high frequencies H (c) absolute values $|H|$ (d) activity/power map: low pass of $|H|$ (e) textureness: cross bilateral filter of $|H|$ and I

Figure 5-8: Textureness of a 1D signal. To estimate the textureness of the input (a), we compute the high frequencies (b) and their absolute values (c) . Finally, we locally average these amplitudes: Previous work based on low-pass filter (d) incurs halos (Fig. 5-9) whereas our cross bilateral filtering yields almost no halos (e) .

the local level of detail can be altered by modifying the detail layer accordingly.

Unfortunately, such spatially-varying manipulation of detail can lead to artifacts at the boundary between highly detailed and smooth regions (Fig. 5-9). This is because the amount of detail on one side of the boundary influences the estimate on the other side, and the manipulation suffers from a halo effect similar to that observed in linear frequency decomposition of image intensity. This problem is the same as the one addressed by edge-preserving decomposition, except that we are dealing with a less spatially localized quantity, the magnitude of high frequency $|H|$. Strong edges are hard to characterize in $|H|$, which is why we define textureness using a *cross-bilateral filter* [83, 84] where the intensity image defines the edge-preserving term to filter $|H|$. More precisely, our textureness is defined as

$$T(I)_{\mathbf{p}} = \frac{1}{k} \sum_{\mathbf{q} \in |H|} g_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)\, g_{\sigma_r}(|I_{\mathbf{p}} - I_{\mathbf{q}}|)\ |H|_{\mathbf{q}} \tag{5.5a}$$

$$\text{with:} \quad k = \sum_{\mathbf{q} \in I} g_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)\, g_{\sigma_r}(|I_{\mathbf{p}} - I_{\mathbf{q}}|) \tag{5.5b}$$

We set this cross filter with the same σ_r as for the base-detail computation, but with a larger σ_s (8 times larger in practice) to ensure smooth textureness variations

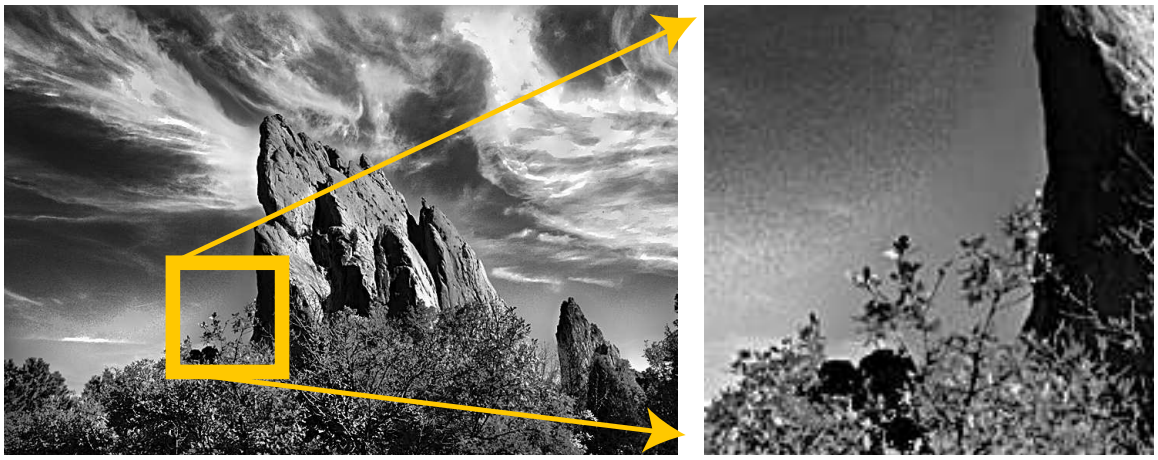


Figure 5-9: Using a Gaussian filter to locally average the high frequency amplitudes yields halos around strong edges. To prevent this defect, we use an edge-preserving filter.

on uniform regions (discontinuities can still happen at edges). Figure 5-10 shows how our textureness map captures the spatial variation of local contrast over the image.

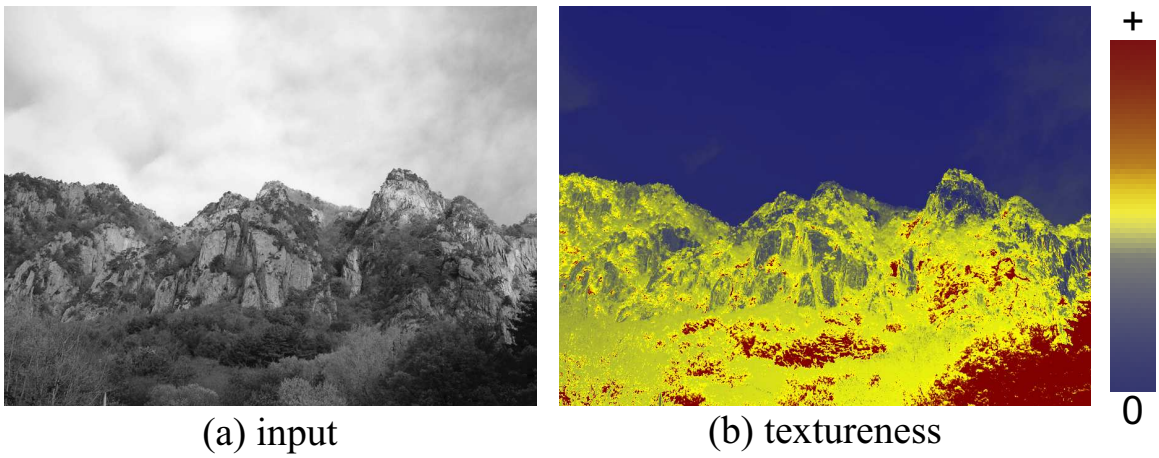
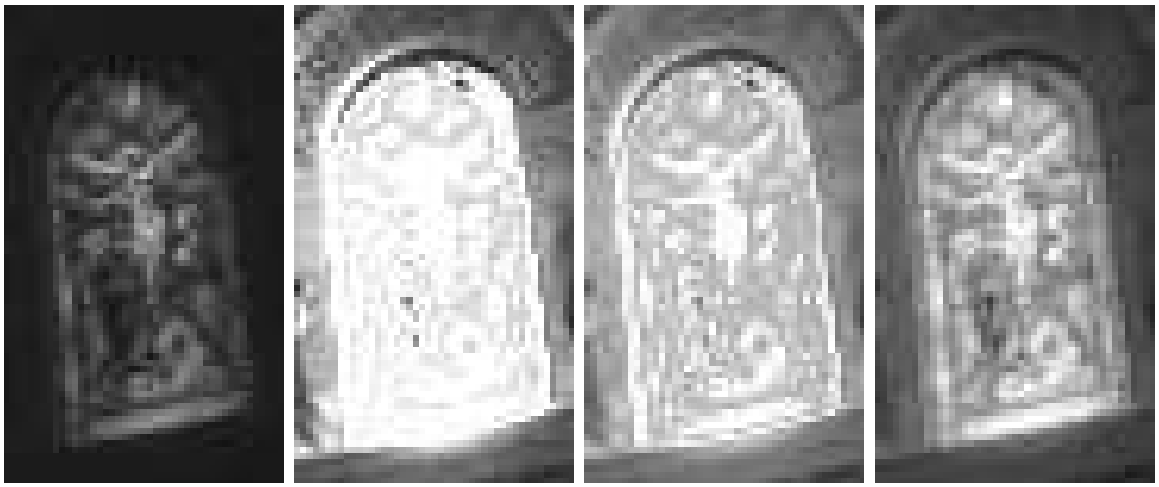


Figure 5-10: Our measure of textureness indicates the regions with the most contrasted texture.

Textureness Transfer The input I and model M have textureness maps $T(I)$ and $T(M)$, respectively. Using histogram transfer, we enforce the histogram of $T(M)$ onto $T(I)$ to build the desired textureness map T'. To prevent halos, we modify only the detail layer D to approximate T'. We scale the values of D by a ratio ρ to



(a) input (b) unconstrained (c) histogram matching (d) gradient constraint

Figure 5-11: Without constraints, the result may lose valuable details (b) because the highlight are saturated. Enforcing the model histogram brings back the intensity values within the visible range (c). Finally, constraining the gradients to preserve some of the original variations (a) produces high quality details (d).

match T' values while accounting for the textureness of the base B' modified by the tonal balance of the previous section:

$$\rho_{\mathbf{p}} = \max\left(0, \frac{T'_{\mathbf{p}} - T(B')_{\mathbf{p}}}{T(D)_{\mathbf{p}}}\right) \tag{5.6}$$

We do not apply negative ratios, thus preventing gradient reversals. Although this computation is done pixel-wise, we found that the textureness maps are smooth enough to ensure a smooth transformation. We linearly recombine the layers to produce the output: $O = B' + \rho D$.

5.6 Detail Preservation

As illustrated by Figure 5-11b, the previous result ($O = B' + \rho D$) may result in saturated highlights and shadows. These bright and dark regions are nevertheless of higher importance for photographers who aim for crisp details everywhere. We preserve these details in two steps.

First, we enforce the intensity histogram of the model M to the current output O, which brings back the values within the displayable range. Second, we modify the gradient field to ensure that no details are removed or overly emphasized. Similarly to our shock removal, we build a gradient field $\mathbf{v}$ that satisfies these constraints. We aim at preserving a portion α of the variations of the input image, and we prevent the gradient being increased by a factor greater than β to avoid over-emphasizing noise. We define:

$$x_{\mathbf{v}} = \begin{cases} \alpha\, \partial I/\partial x & \text{if} \quad |\partial O/\partial x| < \alpha\, |\partial I/\partial x| \\ \beta\, \partial I/\partial x & \text{if} \quad |\partial O/\partial x| > \beta\, |\partial I/\partial x| \\ \partial O/\partial x & \text{otherwise} \end{cases} \tag{5.7}$$

The y component $y_{\mathbf{v}}$ is defined similarly, and the image is reconstructed with the Poisson technique. All that remains is to set α and β. We use percentiles to define $\phi = [p_{95}(O) - p_5(O)]/[p_{95}(I) - p_5(I)]$, which robustly estimates the contrast change induced by our processing. We then use a constant $\alpha = \phi/4$, and we make β depend on intensity in order to avoid increasing noise. We use a smooth-step function $\nu_\tau(x) = 0$ if $x < \tau$, 1 if $x > 2\tau$, and $1 - [1 - (x - \tau)^2/\tau^2]^2$ otherwise. Setting $\beta = 1 + 3\nu_\tau\phi$ performs consistently well with $\tau = 0.1$. As a result, we successfully preserve the richness of the input images as shown on Figure 5-11.

5.7 Additional Effects

While our focus is on the management of the tonal palette and the variation of detail, we have also developed simple filters to control low-level aspects of the look of a photograph.

Soft Focus and Sharpness The level of sharpness of a picture is a strong aspect of style as exemplified by soft-focus effects. To characterize sharpness, we use difference-of-Gaussian filters and analyze three octaves of the current output O.

We set the parameters so that the highest band captures the wavelengths shorter than $\lambda_{\mathrm{h}} = \min(\text{width}, \text{height})/256$. For each band B_i^O, we evaluate the sharpness of the most contrasted edge with the 95th percentile $p_{95}(\left|B_i^O\right|)$. We divide this number by $p_{95}(O) - p_5(O)$ to make this measure invariant to intensity. The use of percentiles makes this estimation robust. To summarize, our sharpness estimator is a triplet of numbers $(\zeta_1, \zeta_2, \zeta_3)$ defined as $\zeta_i^O = p_{95}(\left|B_i^O\right|)/(p_{95}(O) - p_5(O))$. We compute the same measures for the model M and scale the bands B_i^O of the output by a factor ζ_i^M/ζ_i^O to transfer sharpness. See Figure 5-12, 5-15 and 5-19. In particular, in Figure 5-12, the intermediate frequencies are attenuated more than the highest frequencies, achieving a "soft-yet-sharp" rendition which is a convincing approximation of the effect produced by a soft-focus lens.

Film Grain and Paper Texture Some photographs exhibit a characteristic appearance due to the paper which they are printed on or because the film grain is visible. We reproduce this effect in two steps. First, since the grain is not part of the image content, we remove it from the model image with a bilateral filter on the luminance values, using $\sigma_{\mathrm{r}} = p_{75}(\|\nabla M\|)$. Then, we crop a sample from the residual (detail) of this bilateral filter in a uniform region. We generate a grain layer using texture synthesis [85] (Fig. 5-12, and 5-15).

Color and Toning To handle color images, we can use the original a and b channels in the CIE-LAB color space. a and b can be used directly, or they can be scaled by L_O/L_I where L_I and L_O are the luminance of the input and current output. The latter alters color saturation and is useful for HDR images because their chromaticity is often out of the displayable gamut [33, 34]. Figures 5-14 and 5-19 show color renditions.

We produce toned pictures (e.g. sepia) using a one-dimensional color map. We use the Lab color space to build the functions $a(L)$ and $b(L)$ from the model by averaging a and b for the pixels with a given L. These functions are then applied to the L values of the current result (Fig. 5-12).

5.8 Results

We demonstrate our technique using models by different artists on a variety of inputs, including pictures by beginners using point-and-shoot cameras, photos by more advanced amateurs using SLRs, and high-dynamic-range images (Fig. 5-12).

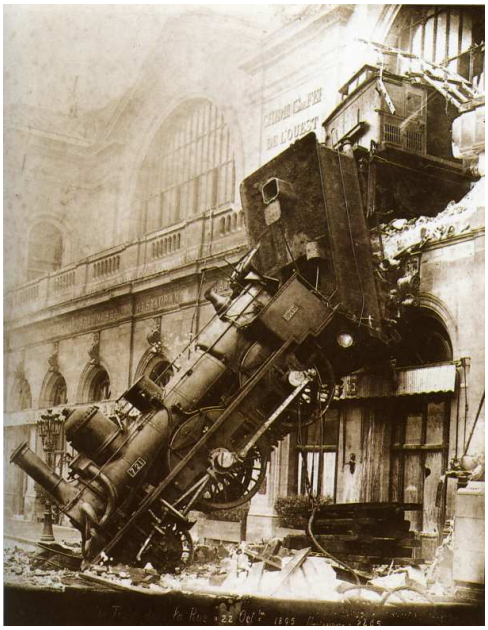
Computation time varies roughly linearly with the number of pixels, thanks to the fast bilateral filter and a multigrid implementation of gradient reconstruction. For example, the full pipeline for a one megapixel image takes about six seconds on a 2.6GHz Opteron PC, and a four megapixel takes 23 seconds. However, note that we cache intermediate results such as the base, detail, and textureness map, which enables interactive feedback when using the user interface.

In addition, results from downsampled images are faithful previews (Fig. 5-13) because our parameters are scale invariant, which enables fast interaction before a final computation at full resolution.

Our implementation enables interactive adjustment of the parameters through controls such as sliders for scalar parameters and, for the remapping function of the base layer, a spline interface inspired by the "curve" tool of photo-editing software. These adjustments can be saved and reused on subsequent inputs. We have also found that the interactive control is a great way to refine the result of an automatic transfer (Fig. 5-14).

Figure 5-15, 5-16, 5-17, and 5-18 show a comparison of our results with a straightforward histogram matching from the model to the input. Histogram matching ignores the notion of texture and therefore overly increases or decreases the picture detail. In comparison, our technique yields results that are both more faithful to the model and higher quality, with rich shadows and detailed highlights.

Discussion The main cause of failure of our approach is poor input quality. In particular JPEG artifacts and noise can be amplified by our detail manipulation (Fig. 5-20). Apart from this, meaningful input/model couples (two landscapes, two trees, etc) consistently yield faithful transfers, close to our expectations. On



(a) model (608x785)

(b) HDR input (512x768)

(c) direct histogram matching

(d) our result

Figure 5-12: Our system can seamlessly handle HDR images . We can turn a sharp picture (b) into a soft grainy and toned photograph (d). We have toned the histogram-transferred version (c) to prevent biased comparison due to different color cast. The model (a) is Accident at the Gare Montparnasse from the Studio Lévy and Sons, 1895. The input (b) is courtesy of Paul Debevec, USC

(a) low resolution (b) high resolution, twice of (a)

Figure 5-13: Results from lower resolution (a) provides quick previews and allow for interactive adjustments before rendering high resolution results (b). Limited differences are visible on the smallest details (*e.g.* in the background) because they are not well sampled in the low-resolution image.

(a) input image (800x424) (b) result from model (c) result after user adjustment

Figure 5-14: This rendition was obtained in two steps. We first used Kenro Izu's picture shown in Figure 2-8b as a model (b). Then, we manually increased the brightness and softened the texture to achieve the final rendition (c) that we felt is more suitable for the scene.

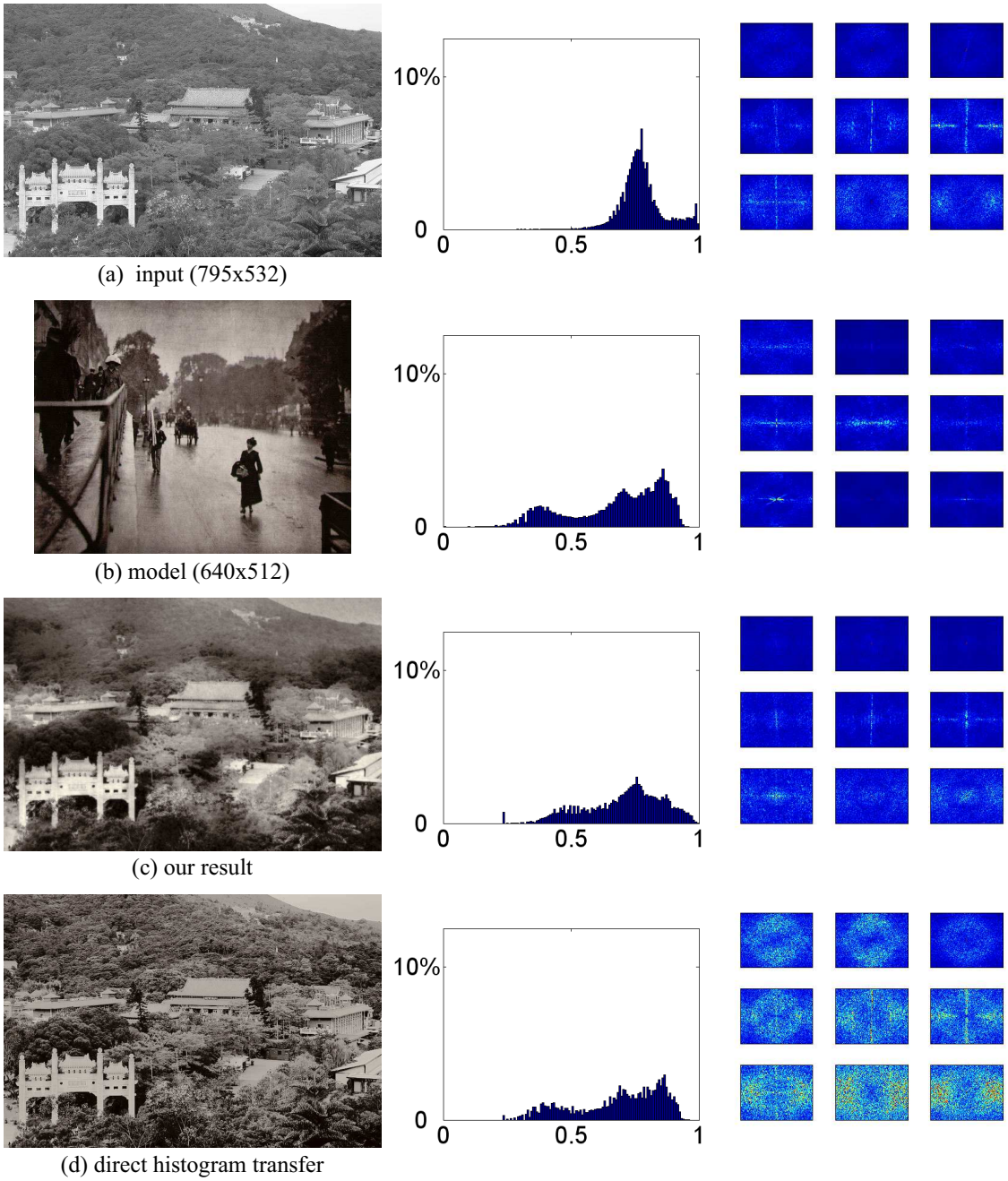
(a) model (640x512)

(b) input (795x532)

(c) direct histogram matching

(d) our result

Figure 5-15: A simple histogram matching from the model (a) to the input (b) increases the texture level of the image (c) whereas the model has little texture. In comparison, we successfully reduce the texture and the sharpness to achieve large uniform gray regions similar to those in the model. The model is Snapshot by Alfred Stieglitz.



(a) input (795x532)

(b) model (640x512)

(c) our result

(d) direct histogram transfer

Figure 5-16: Histogram and frequency comparison. Our result (c) faithfully transfers local contrast from the model photo (b). In contrast, a direct histogram matching (d) increases the high frequency contents by spreading the luminance values. In the local spectral signatures, the color close to red means a high value and that close to blue means a low value.

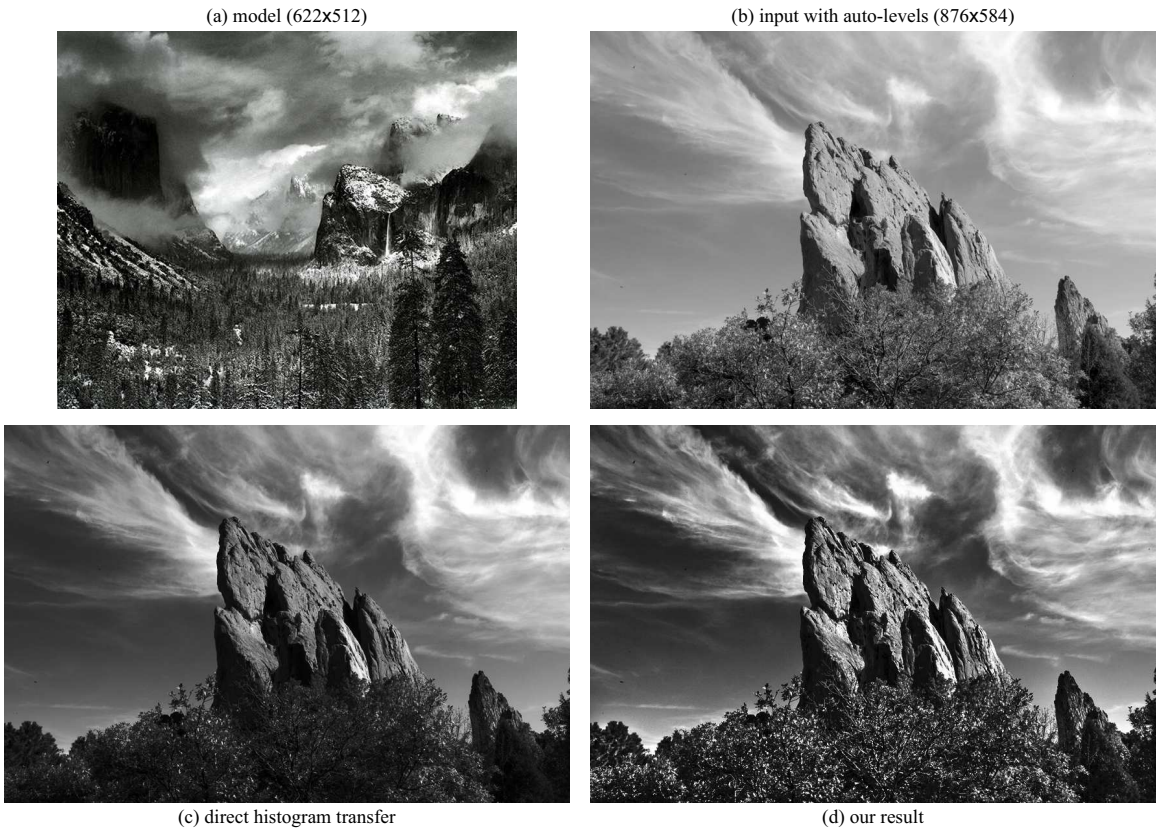
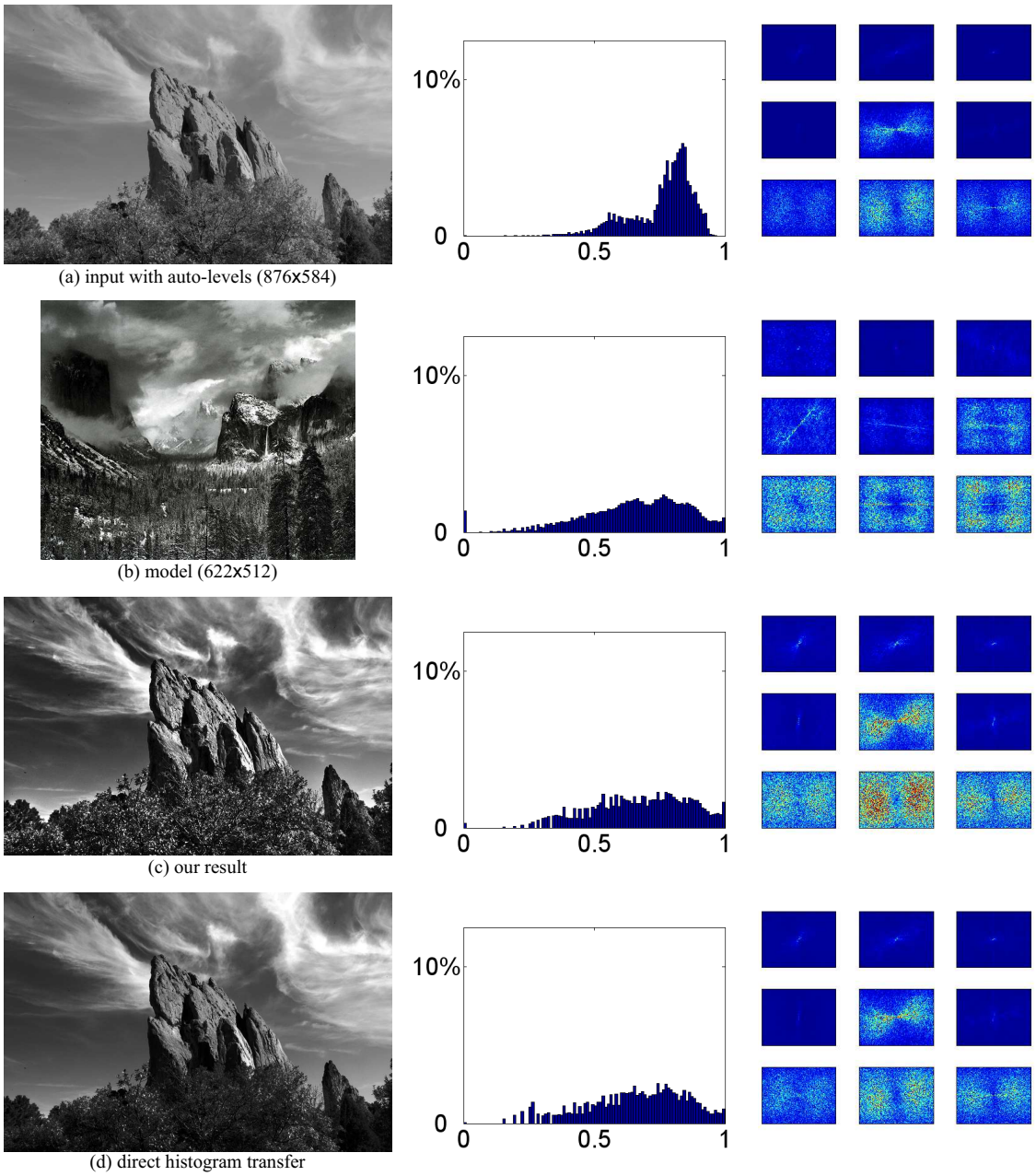


Figure 5-17: Our approach is able to reproduce the level of texture observed in Adams' masterpiece (a) to achieve a compelling rendition (d). In comparison, Adobe® Photoshop® "auto-level" tool spans the image histogram on the whole intensity range. This reveals the small features of a picture but offers no control over the image look (b). And, a direct histogram transfer only adjusts the overall contrast and ignores the texture, thereby producing a dull rendition (c).



(a) input with auto-levels (876x584)

(b) model (622x512)

(c) our result

(d) direct histogram transfer

Figure 5-18: Histogram and frequency comparison. A direct histogram matching (d) increases the high frequency contents by spreading the luminance values, but not as much as our result (c). In the local spectral signatures, the color close to red means a high value and that close to blue means a low value.

(a) input image (1200x900) (b) our result

Figure 5-19: For color images, we process the luminance channel of the image and keep the original chrominance channels. In this example, the details are enhanced while the overall contrast and sharpness are increased. We used Adams' picture (Fig. 5-17a) as a model.

more surprising pairs (*e.g.* a flower and a landscape), the process does not generate artifacts and the achieved mood is often pleasing, although one can always argue about the æsthetic quality of some results. Portraits are probably the most challenging type of input, and detail enhancement can lead to unflattering result because skin defects can be emphasized. It is then best to turn this feature off.

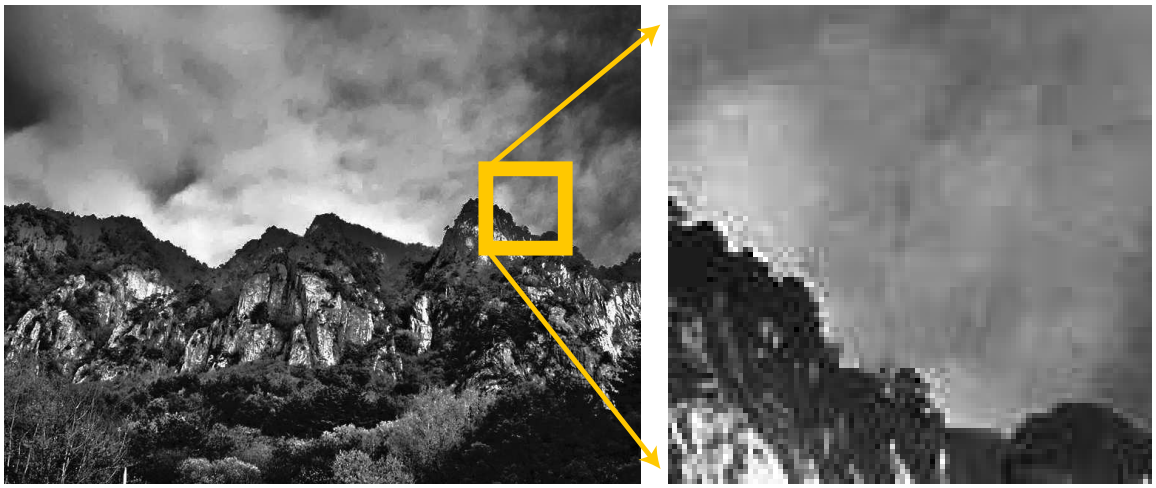


Figure 5-20: Our technique suffers from imperfections such as JPEG artifacts. In this example, the artifacts in the sky are not visible in the input image (Fig. 5-10a) but appear clearly after processing.

5.9 Conclusions

We have presented an approach to manipulate the tonal look of digital photographs. Using a combination of non-linear edge-preserving decomposition and linear analysis, we control both the large-scale tonal palette and the detail over an image. In particular, we manipulate the spatial variation of high-frequencies using a new textureness map that performs an edge-preserving analysis and manipulation of the high-frequency content. We have introduced a gradient constraint that preserves image content and prevents gradient reversal and halos.

Our method can be used to transfer the look of a model photograph or can be directly controlled using a simple interface. It allows for the exploration of a variety of styles and achieves high-quality results that are consistent from low-resolution previews to high-resolution prints.

This work opens several areas of future research. It should be combined with approaches to control the color components of pictorial style. While early experiments with videos have shown that our technique itself is stable, we have found that the biggest challenge is the fluctuation created by auto-exposure, autofocus and the variation of motion blur when the camera moves.

Chapter 6

Defocus Magnification

Sharp foreground with blurred background is preferred in many types of photography such as portraits. But point-and-shoot cameras have small lenses and sensors, which fundamentally limits their ability to defocus the background and generate shallow depth of field. We present an image-processing technique that magnifies existing defocus given a single photo.

For a given field of view and subject distance, depth of field is directly related to the physical diameter of the lens aperture. This means that compact cameras that rely on smaller sensors – and therefore on smaller lenses – yield less defocus and cannot blur the background the way a large-aperture single-lens reflex (SLR) lens can (Fig. 6-2). While a smaller amount of defocus (larger depth of field) can be desirable, for example in landscape or macro photography, it is often a serious limitation for portraits and creative photography. Users of compact cameras often complain that their portraits do not look "artistic" and lack the clarity afforded by defocused backgrounds. In fact, the quality of a blurry background, called *bokeh*, has a real cult following among some photographers.

Our technique takes a single input image where the depth of field is too large and increases the amount of defocus present in out-of-focus regions. That is, our goal is opposite to that of work that seeks to create images that are sharp everywhere.

Our approach first estimates the spatially-varying amount of blur over the im-

(a) input (b) defocus map (c) our result with magnified defocus

Figure 6-1: Our technique magnifies defocus given a single image. Our defocus map characterizes blurriness at edges. This enables shallow depth of field effects by magnifying existing defocus. The input photo was taken by a Canon PowerShot A80, a point-and-shoot camera with a sensor size of 7.18×5.32 mm, and a 7.8 mm lens at f/2.8.

(a) large sensor (22.2 x 14.8 mm) (b) small sensor (7.18 x 5.32 mm)

Figure 6-2: Given the same field of view and the same f-number (f/2.8), a large sensor (a) yields more defocus than a small sensor (b) does.

age, and then uses an off-the-shelf image-based technique to increase defocus. We first estimate the size of the blur kernel at edges, building on the method by Elder and Zucker [86], and then propagate this defocus measure over the image with a non-homogeneous optimization. Using our defocus map, we can magnify the existing blurriness, which means that we further blur blurry regions and keep sharp regions sharp.

Note that in contrast to more difficult problems such as depth from defocus, we do not require precise depth estimation and do not need to accurately disambiguate smooth regions of the image, since such regions are not much affected by extra blur due to defocus. The fundamental ambiguity between out-of-focus edges and originally smooth edges is out of the scope of our work. We also do not need to disambiguate between objects in front and behind the plane of focus. We sim-

ply compute the amount of blur and increase it. While our method does not produce outputs that perfectly matches images captured with a larger-aperture lens, it qualitatively reproduces the amount of defocus. We refer interested readers to Appendix 2.1.2 where we review thin-lens optics and defocus.

6.1 Overview

For each pixel, we estimate the spatially-varying amount of blur. We call our blur estimation the *defocus map*. We estimate the defocus map in two steps. First, we estimate the amount of blur at edges. Then, we propagate this blur measure to the rest of the image.

We model an edge as a step function and the blur of this edge as a Gaussian blurring kernel. We adapt the method by Elder and Zucker [86], which uses multiscale filter responses to determine the size of this kernel. We add a cross-bilateral filtering step [83, 84] to remove outlier estimates.

We propagate the blur measure using non-homogeneous optimization [87]. Our assumption is that blurriness varies smoothly over the image except where the color is discontinuous. We propagate blurriness measure to the neighbors with similar intensity and color.

We can use our defocus map to magnify defocus effects. We blur each pixel according to its estimated blurriness. If we double our defocus map, it doubles defocus effects as if the image is taken with an aperture that is twice as large. In this paper, our results are generated using Adobe® Photoshop® lens blur with our defocus map as a depth map.

6.2 Blur Estimation

The amount of blur can be estimated reliably only in areas of an image that has significant frequency content. This is why we focus on edges. However, we need to extract and analyze edges with various levels of blurriness, which makes the technique by Elder and Zucker [86] particularly appropriate. We refine their technique by introducing the explicit fitting of a blurred edge model that is more robust than the original technique. Also, our refinement step reduce outliers due to blurry features such as soft shadows.

6.2.1 Detect blurred edges

Following Elder and Zucker, we model an edge as a step function in intensity, and the blur of this edge as a Gaussian blurring kernel:

$$g(x, y, \sigma_b) \quad = \quad \frac{1}{2\pi\sigma_b^2} \exp(-(x^2 + y^2)/2\sigma_b^2) \tag{6.1}$$

where σ_b denotes the scale of the blur, and is what we want to estimate.

For each pixel, Elder and Zucker determine the right scale for edge detection using the noise thresholds. More details can be found below. We used $\sigma_1 \in \{64\ 32\ 16\ 8\ 4\ 2\ 1\ 0.5\}$ pixels and $\sigma_2 \in \{32\ 16\ 8\ 4\ 2\ 1\ 0.5\}$ pixels. We apply a strict threshold, $s_n = 2.5$ and $\alpha_I = 0.0001\%$, to achieve very reliable blur estimation.

Elder and Zucker's edge detector Elder and Zucker [86] detect edges with various levels of blurriness. To determine the right scale for edge detection, they compute the minimum reliable scale for each pixel, based on the noise thresholds . They locate edges by testing nonzero gradient and zero-crossing of second derivative at the minimum reliable scale.

For each pixel, Elder and Zucker compute its multiscale responses to the steerable Gaussian first derivative filters and steerable second derivative of Gaussian

filters and compute the gradient using the steerable Gaussian first derivative basis filters:

$$g_1^x(x,y,\sigma_1) = \frac{-x}{2\pi\sigma_1^4}\exp(-(x^2+y^2)/2\sigma_1^2) \quad (6.2a)$$

$$g_1^y(x,y,\sigma_1) = \frac{-y}{2\pi\sigma_1^4}\exp(-(x^2+y^2)/2\sigma_1^2) \quad (6.2b)$$

where σ_1 is the scale of the first derivative Gaussian estimator. A weighted sum of these two filter responses is used to compute the gradient direction θ that maximizes the gradient magnitude.

They compute the second derivative in the direction θ using a steerable second derivative of Gaussian operator:

$$g_2^x(x,y,\sigma_2) = \frac{(x/\sigma_2)^2-1}{2\pi\sigma_2^4}\exp(\frac{-(x^2+y^2)}{2\sigma_2^2}) \quad (6.3a)$$

$$g_2^y(x,y,\sigma_2) = \frac{(y/\sigma_2)^2-1}{2\pi\sigma_2^4}\exp(\frac{-(x^2+y^2)}{2\sigma_2^2}) \quad (6.3b)$$

$$g_2^{xy}(x,y,\sigma_2) = \frac{xy}{2\pi\sigma_2^6}\exp(\frac{-(x^2+y^2)}{2\sigma_2^2}) \quad (6.3c)$$

$$g_2^\theta(x,y,\sigma_2) = \cos^2(\theta)g_2^x(x,y,\sigma_2) + \sin^2(\theta)g_2^y(x,y,\sigma_2) \quad (6.3d)$$

$$- 2\cos(\theta)\sin(\theta)g_2^{xy}(x,y,\sigma_2) \quad (6.3e)$$

where σ_2 is the scale of the second derivative of Gaussian filter.

They test the reliability of filter responses by setting a threshold for each scale. The thresholds are derived from the sensor noise level s_n. In the following equations, c_1 denotes the threshold for Gaussian first derivative filter in a function of σ_1 and c_2 denotes the threshold for the second derivative of Gaussian filter in a

function of σ_2.

$$c_1(\sigma_1) = \frac{s_n\sqrt{-2\ln\sigma_p}}{2\sqrt{2\pi}\cdot\sigma_1^2} \tag{6.4a}$$

$$c_2(\sigma_2) = \frac{s_n\sqrt{2}\cdot erf^{-1}(\alpha_p)}{4\sqrt{\pi/3}\cdot\sigma_2^3} \tag{6.4b}$$

$$\text{with: } \alpha_p = 1-(1-\alpha_I)^{1/n} \tag{6.4c}$$

where n is the number of pixels. The thresholds are computed statistically based on the standard deviation of the sensor noise s_n and a false positive tolerance α_I. At the minimum reliable scale, pixel filter responses are larger than the threshold of the scale.

6.2.2 Estimate blur

In their technique, Elder and Zucker estimate the amount of blur by measuring the distance d between second derivative extrema of opposite sign in the gradient direction. This directly follows from the analytical derivation of a perfect step edge convolved with a Gaussian, as shown in their paper [86].

However, we have found that, for real images, the localization of the second-derivative extrema of the edge using the zero-crossing of the third derivative is not robust, which is acknowledged in their article. This leads to errors in the estimation of the blur amount. Therefore, instead of measuring the distance between actual extrema, we fit the multiscale models of the second derivative Gaussian filter response to the pixel responses and find the distance with a least square fitting error. Given the estimated distance, we compute the size of blur kernel σ_b using Equation 6.5 (d). This provides us with a sparse set of blur measures BM at edge pixels in the image.

We fit the response model using a brute-force strategy. We fit the response model with a number of values for distance d (Fig. 6-3) to a window around the edge pixel and along the gradient direction. Elder and Zucker use an edge pixel at the dark side of the edge. But we found that using both bright and dark sides of

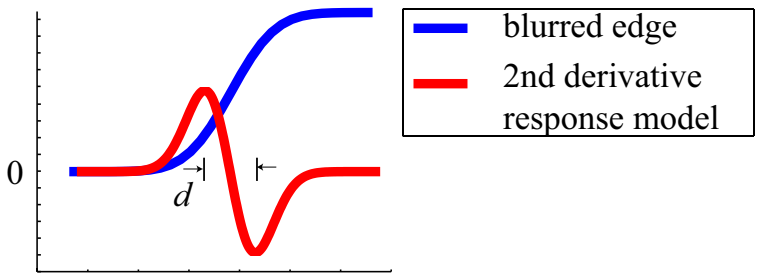


Figure 6-3: The model for the distance between second-derivative extrema. We numerically fit this response model with various d around the edge pixel and along the gradient direction to find the distance d with a least square fitting error.

the edge generates more reliable defocus maps. We use window sizes from 3×3 to 71×71. Given a blurred step edge along the y axis of amplitude A and blur parameter σ_b, the expected response to the second derivative filter is modeled by:

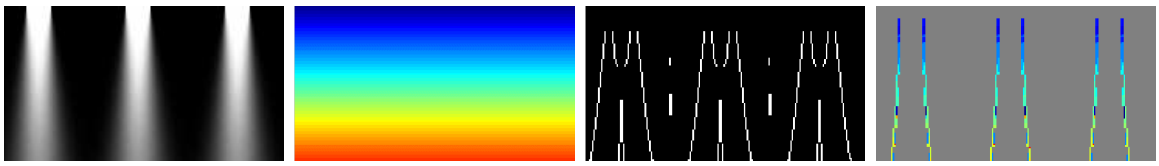
$$
\begin{aligned}
r_2^x(x, y, \sigma_2) &= Au(x) * g_2^x(x, y, \sigma_b^2 + \sigma_2^2) && \text{(6.5a)} \\
&= \frac{-Ax}{\sqrt{2\pi}(\sigma_b^2 + \sigma_2^2)^{3/2}} \exp(-x^2/2(\sigma_b^2 + \sigma_2^2)) && \text{(6.5b)} \\
&= \frac{-Ax}{\sqrt{2\pi}(d/2)^3} \exp(-x^2/2(d/2)^2) && \text{(6.5c)} \\
\text{with: } (d/2)^2 &= \sigma_b^2 + \sigma_2^2 && \text{(6.5d)}
\end{aligned}
$$

where $u(x)$ is a step function. We derive A from the local extrema within each window.

Figure 6-4 shows that our approach can successfully estimate blur measures while the zero-crossing of the third derivative cannot localize the second derivative extrema.

6.2.3 Refine blur estimation

Depth of field effects are not the only cause of edge blurriness in images and phenomena such as soft shadows and glossy highlights can result in erroneous estimates of defocus.



(a) input (b) actual blur sigma (c) the zero-crossing of the third derivative (d) blur measure using our approach

Figure 6-4: The zero-crossing of the third derivative (c) is greatly affected by neighboring edges and cannot localize the second derivative extrema. In contrast, our approach (d) can estimate the blur sigma that is close to the actual blur sigma (b). The input (a) is generated using the blur sigma (b). In the blur measure, the color close to red means blurry and that close to blue means sharp.

We suppress the influence of these outliers by smoothing the blur measure with an edge-preserving filter. We apply cross bilateral filtering [83, 84] to our sparse set of blur measures, BM. The cross-bilateral filtering output is a weighted mean of its neighbors where the weights decrease with the distance in space and with the range difference of a reference image.

In addition to the original cross bilateral filtering weights, we use a sharpness bias, $b(BM) = \exp(-BM/2)$. The sharpness bias corrects blur measures in soft shadows and glossy highlights that are higher than they are supposed to be.

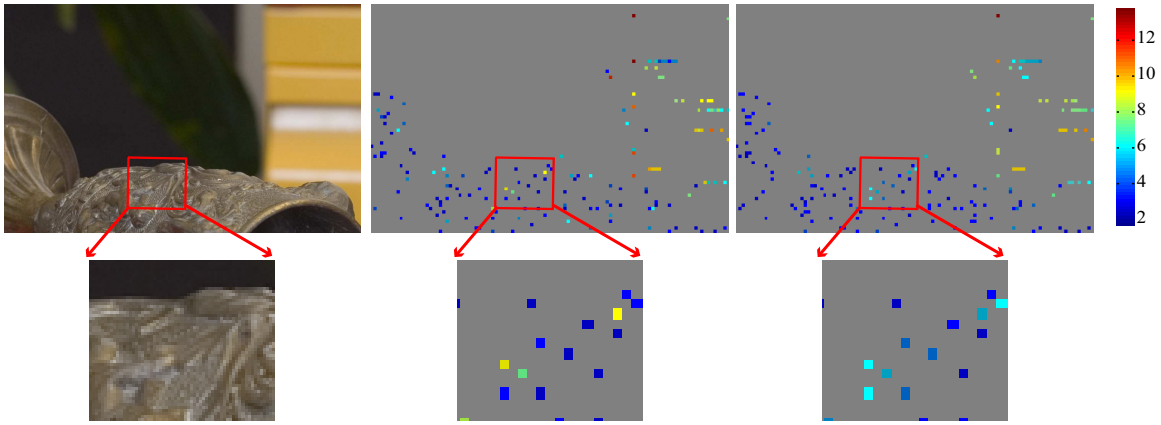
With $g_\sigma(x) = \exp(-x^2/2\sigma^2)$, a Gaussian function, we define the biased cross bilateral filtering of a sparse set of blur measures, BM at an edge pixel $\mathbf{p}$ as the following:

$$bCBF(BM)_{\mathbf{p}} = \frac{1}{k} \sum_{\mathbf{q} \in BM} w_{\mathbf{pq}}\, b(BM_{\mathbf{q}})\, BM_{\mathbf{q}} \tag{6.6a}$$

$$\text{with:} \quad w_{\mathbf{pq}} \propto \sum_{\mathbf{i} \in \{R,G,B\}} g_{\sigma_\mathrm{s}}(\|\mathbf{p}-\mathbf{q}\|)\; g_{\sigma_\mathrm{r}}\left(|C_i(\mathbf{p}) - C_i(\mathbf{q})|\right) \tag{6.6b}$$

$$\text{and} \quad k = \sum_{\mathbf{q} \in BM} w_{\mathbf{pq}}\, b(BM_{\mathbf{q}}) \tag{6.6c}$$

where σ_s controls the spatial neighborhood, and σ_r the influence of the intensity difference, and k normalizes the weights. We use the RGB color channels of the original input image as the reference and set $\sigma_r = 10\%$ of the image range and $\sigma_s =$



(a) input

(b) blur measure before the cross bilateral filtering

(c) blur measure after the cross bilateral filtering

Figure 6-5: Blur measure before and after the cross bilateral filtering. The cross bilateral filtering refines outliers such as yellow and green measures (b), which mean blurry, in the focused regions to be blue measures (c), which means sharp. The blur measures are downsampled using nearest neighbor for better illustration.

10% of the image size. This refinement process does not generate much change but refines a few outliers as shown in Figure 6-5. The cross bilateral filtering refines outliers such as yellow and green measures (b) in the focused regions to be blue (c).

6.3 Blur Propagation

Our blur estimation provides blur kernels only at edges and we need to propagate this blur measure. We use non-homogeneous optimization [87] and assume that the amount of defocus is smooth when intensity and color are smooth.

6.3.1 Propagate using optimization

Our propagation is inspired by the *colorization* paper by Levin et al.[87]. We impose the constraint that neighboring pixels $\mathbf{p}, \mathbf{q}$ have similar blurriness if they have similar intensities and colors. We minimize the difference between the blurriness $B(\mathbf{p})$ and a weighted average of blurriness of neighboring pixels:

$$E(B) = \sum (B(\mathbf{p}) - \sum_{\mathbf{q} \in N(\mathbf{p})} w_{\mathbf{pq}} B(\mathbf{q}))^2 \tag{6.7a}$$

$$+ \sum \alpha_{\mathbf{p}} \, (B(\mathbf{p}) - BM(\mathbf{p}))^2 \tag{6.7b}$$

$$\text{with:} \quad w_{\mathbf{pq}} \propto \sum_{\mathbf{i} \in \{R,G,B\}} exp(\frac{-(C_i(\mathbf{p}) - C_i(\mathbf{q}))^2}{2\sigma_{i_{\mathbf{p}}}^2}) \tag{6.7c}$$

where $\sigma_{\mathbf{p}}$ is the standard deviation of the intensities and colors of neiboring pixels in a window around $\mathbf{p}$. The window size used is 7×7. We have experimented both with setting the second term as hard constraints vs. as a quadratic data term, and have found that the latter is more robust to potential remaining errors in the blur measure.

We solve this optimization problem by solving the corresponding sparse linear system. Figure 6-6 shows the defocus map for various values of α. We use $\alpha = 0.5$ for edge pixels.

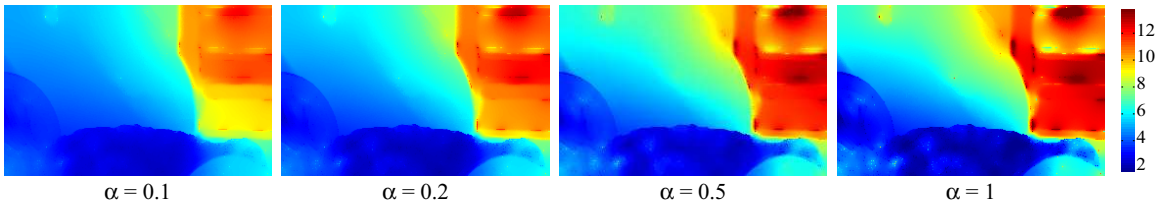


Figure 6-6: Defocus map with various α. α controls the balance between the smoothness penalty term and data term in Equation 6.7. We use $\alpha = 0.5$ for edge pixels and $\alpha = 0$ for non-edge pixels, which do not have values. In this plot, red means blurry region and blue means sharp regions. The input image is Figure 6-5.

6.4 Results

We have implemented our blur estimation using Matlab. Our defocus map enables defocus magnification. We rely on Photoshop's lens blur to compute the defocused output. We crop the upper and lower 5% of the defocus map and clamp its minimum value to 0. In addition, we apply Gaussian blur to the defocus map to use it as a depth map. The Gaussian blur radius is set to 0.5% of the image size.

Using our defocus map, we can simulate the effect of doubling the aperture size. Figure 6-8 compares two input defocus maps of two images with the f-number 8 (a) and 4 (b). As we double the defocus map (c) of the f/8 image, we obtain a result similar to the defocus map (d) of the f/4 image. While the simulated defocused map (e) is not exactly the same as the real map (d), the output image with magnified defocus (f) is visually close to the f/4 photograph (b).

In Figure 6-7, we show the results of using our defocus map to magnify the existing defocus effects in the original images. The results preserve the sharpness of the focused regions but increase the blurriness of the out-of-focus regions.

In addition, while our defocus map is not really a depth map, it is sometimes possible to use it to refocus a photograph resembling the effect of Ng et al. [54] and Isaksen et al. [88]. Figure 6-9 shows a result where our defocus magnification is applied with a virtual focusing distance. Before we apply lens blur, we performed deconvolution using our defocus map. The result looks as if the foreground is focused.

Figure 1 and 6-9 and the two rows in the middle of Figure 6-7 were taken by a Canon PowerShot A80, a point-and-shoot camera with a sensor size of 7.18×5.32 mm, and a 7.8 mm lens at f/2.8. Figure 6-5 and 6-8 were taken by a Canon 1D Mark II with a sensor size of 28.7×19.1 mm and a Canon EF 85mm f/1.2L lens. The first input of Figure 6-7 was taken by a Nikon D50 with a sensor size of 23.7×15.6 mm and a 180.0 mm lens at f/4.8. The two rows at the bottom of Figure 6-7 are from *bigfoto.com*.

6.4.1 Discussion

Our defocus maps are different from their actual depth maps mostly in smooth regions of the image that are not much affected by extra blur due to defocus. For example, the gradients in human skin are interpreted as blurry regions. However, such artifacts do not cause visual defects in the results with magnified defocus. You can notice some of these issues in Figure 6-7.

A limitation of our technique is that occlusion boundaries that separate sharp foreground and blurry background are sometimes erroneously blurred (e.g. the top of the Teddy bear in Fig. 6-1)

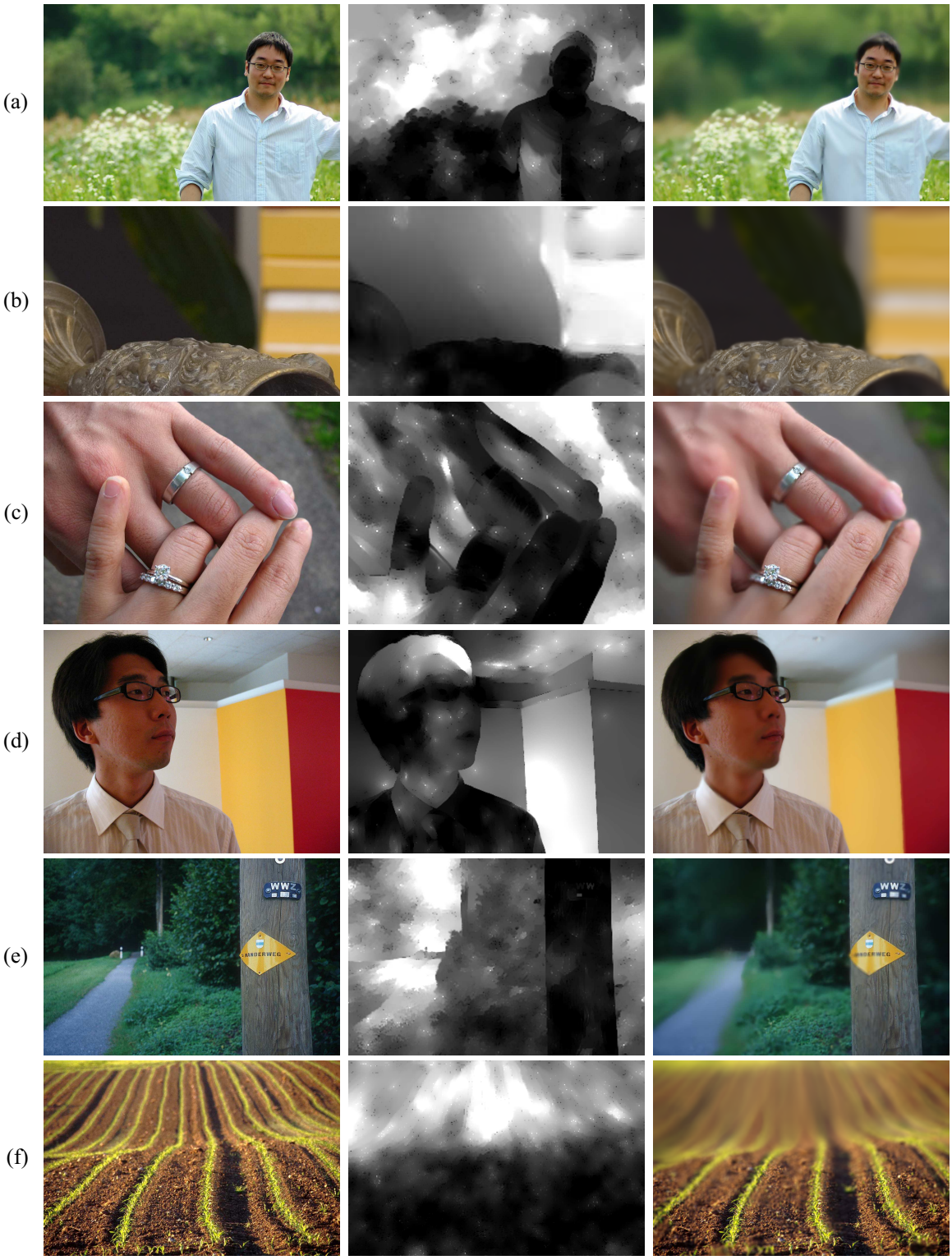
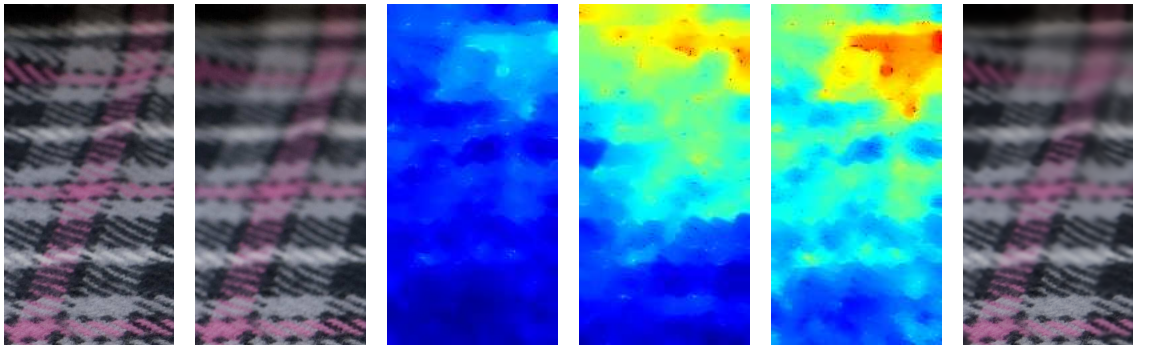
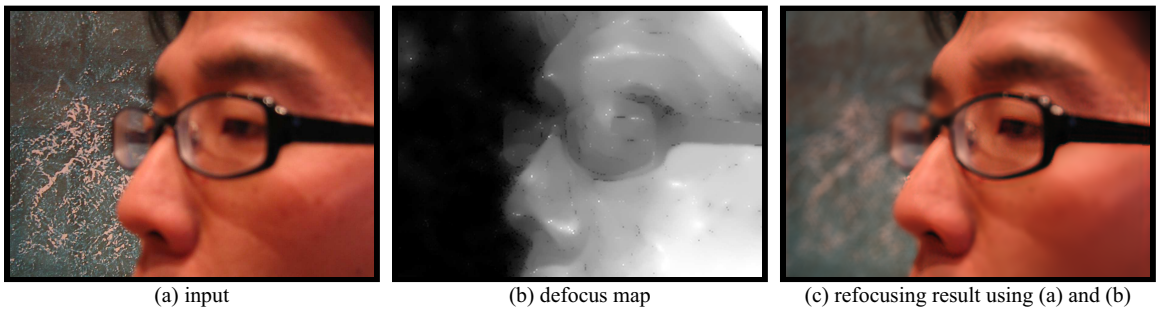


Figure 6-7: Results. The original images, their defocus maps, and results blurred using our approach. The inputs were taken by (a) a Nikon D50 with a sensor size of 23.7×15.6 mm and a 180.0 mm lens at f/4.8, (b) a Canon 1D Mark II with a sensor size of 28.7×19.1 mm and a Canon EF 85mm f/1.2L lens, and (c, d) a Canon PowerShot A80, a point-and-shoot camera with a sensor size of 7.18×5.32 mm, and a 7.8 mm lens at f/2.8. The two at the bottom are from *bigfoto.com*.



(a) f/8 input (b) f/4 input (c) f/8 defocus map (d) f/4 defocus map (e) doubling (c) (f) our synthesized result using (a) and (e)

Figure 6-8: Doubled defocus. Doubling the defocus map generates a effect of doubling the aperture size. As we double the defocus map (c) of the f/8 image, we obtain a result similar to the defocus map (d) of the f/4 image. While the simulated defocused map (e) is not exactly the same as the real map (d), the output image with magnified defocus (f) is visually close to the f/4 photograph (b).



(a) input (b) defocus map (c) refocusing result using (a) and (b)

Figure 6-9: Using our defocus map, we can synthesize refocusing effects. We perform deconvolution using our defocus map (b) and apply lens blur. The result (c) looks as if the foreground is focused. The input photo was taken by a Canon PowerShot A80, a point-and-shoot camera with a sensor size of 7.18×5.32 mm, and a 7.8 mm lens at f/2.8.

6.5 Conclusions

We have presented an image-processing technique to magnify the amount of defocus due to lens aperture. Given a single image, we estimate the size of the blur kernel at edges and propagate the blur measure to the overall image. We use a multiscale edge detector and model fitting to estimate the size of blur kernel. We propagate the blur measure assuming that blurriness is smooth where intensity and color are similar.

Unlike more difficult problems such as depth from defocus, we do not need to generate an accurate depth map and do not need to disambiguate textureless regions. Our defocus map focuses on edges and texture regions that are visually affected by defocusing and approximates textureless regions without causing visual defects.

In future work, we want to extend this work to video inputs where the effect of motion blur needs to be distinguished from depth of field. Finally, we also want to further study occlusion boundaries, a traditional issue for depth of field effects.

Chapter 7

Conclusions

In this dissertation, we have presented three contributions that analyze and transfer photographic viewpoint and appearance including computational rephotography, style transfer, and defocus magnification. In computation re-photography, we propose an assistive camera that analyzes the scene and guides users. We develop a real-time pose estimation and visualization to guide users to a desired viewpoint. In style transfer, we transfer global and local contrast from a model to an input. To this end, we introduce an edge-preserving textureness. In defocus magnification, we generate synthetic shallow depth of field effects by analyzing and magnifying existing defocus given a single image.

7.1 Future work

In future work, we want to extent our image enhancement technologies to advanced processing technologies for videos. Although digital image capture, processing, and sharing have become pervasive, creating a video has not been common yet. It is difficult to capture good footage that is well composed, beautifully lit, and endearing. In addition, video editing requires painstaking work and advanced skill. As a result, videos often become long, boring, and hard to explore. The development of post-processing and searching technologies will make it easy to edit, share, and browse videos. In particular, we want to extend our defocus

magnification to video inputs. To process videos, we need to separate motion blur from defocus.

In addition, we want to study occlusion boundaries. Occlusion boundaries such as corners are very important cues for scene understanding and matching. Current feature descriptors rely on local appearances, and they cannot find matches between actual corners. Occlusion boundary detector and descriptor will improve the performance of feature matching.

Moreover, we would like to explore various composite of old and new photos. We want to use the estimated structure to modify viewpoints, structures, colors, and illumination. This is related to image-based modeling, rendering, and lighting.

Finally, we envision that cameras will become more interactive and embed more computation. We want to embed real-time scene analysis and user interaction onto cameras. The camera will guide users to take better pictures with the subject in focus, and with better lighting and framing. These techniques need to understand users' preferences, goals, and skills.

Bibliography

[1] Ansel Adams. *The Camera*. Little, Brown and Company, 1995.

[2] Douglas Levere, Bonnie Yochelson, and Paul Goldberger. *New York Changing: Revisiting Berenice Abbott's New York*. Princeton Architectural Press, 2004.

[3] Elizabeth McNulty. *Boston Then and Now*. Thunder Bay Press, 2002.

[4] Ansel Adams. *The Print*. Little, Brown and Company, 1995.

[5] Andrew Adams, Natasha Gelfand, and Kari Pulli. Viewfinder alignment. *Comput. Graph. Forum*, 27(2):597–606, 2008.

[6] D. Lowe. Distinctive image features from scale-invariant keypoints. In *IJCV*, volume 20, pages 91–110, 2003.

[7] H. Stewénius, C. Engels, and D. Nistér. An efficient minimal solution for infinitesimal camera motion. In *CVPR*, Zurich, Switzerland, June 2007.

[8] Richard I. Hartley. Estimation of relative camera positions for uncalibrated cameras. In *ECCV*, pages 579–587, 1992.

[9] Tim Rudman. *The Photographer's Master Printing Course*. Focal Press, 1994.

[10] E. Hecht. *Optics*. Addison-Wesley, Reading, MA, 2002.

[11] Olivier Faugeras. *Three-dimensional computer vision: a geometric viewpoint*. MIT Press, Cambridge, MA, USA, 1993.

[12] Anders Heyden and Gunnar Sparr. Reconstruction from calibrated cameras Ů a new proof of the kruppa-demazure theorem. *J. Math. Imaging Vis.*, 10(2):123–142, 1999.

[13] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[14] Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 279–288, New York, NY, USA, 1993. ACM.

[15] T. Werner, R. D. Hersch, and V. Hlavac. Rendering real-world objects using view interpolation. In *ICCV '95: Proceedings of the Fifth International Conference on Computer Vision*, page 957, Washington, DC, USA, 1995. IEEE Computer Society.

[16] Peter Sand and Seth Teller. Video matching. *ACM Transactions on Graphics*, 23(3):592–599, August 2004.

[17] Ronen Basri, Ehud Rivlin, and Ilan Shimshoni. Visual homing: Surfing on the epipoles. *International Journal of Computer Vision*, 33(2):117–137, September 1999.

[18] M. Pollefeys, D. Nistér, J. M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. J. Kim, P. Merrell, C. Salmi, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewénius, R. Yang, G. Welch, and H. Towles. Detailed real-time urban 3d reconstruction from video. *IJCV*, 2008.

[19] A.J. Davison, I.D. Reid, N.D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *PAMI*, 29(6):1052–1067, June 2007.

[20] Michael Scheuering, Christof Rezk-Salama, Helmut Barfufl, Armin Schneider, and Guenther Greiner. 2002.

[21] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. *ACM Transactions on Graphics*, 25(3):835–846, July 2006.

[22] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond, and D. Schmalstieg. Pose tracking from natural features on mobile phones. In *IEEE/ACM Internatialn Symposium on Mixed and Augmented Reality*, pages 125–134, 2008.

[23] Gabriel Takacs, Vijay Chandrasekhar, Natasha Gelfand, Yingen Xiong, Wei-Chao Chen, Thanos Bismpigiannis, Radek Grzeszczuk, Kari Pulli, and Bernd Girod. Outdoors augmented reality on mobile phone using loxel-based visual feature organization. In *ACM international conference on Multimedia information retrieval*, 2008.

[24] Patrick Baudisch, Desney Tan, Drew Steedly, Eric Rudolph, Matt Uyttendaele, Chris Pal, and Richard Szeliski. Panoramic viewfinder: providing a real-time preview to help users avoid flaws in panoramic pictures. In *OZCHI '05: Proceedings of the 19th conference of the computer-human interaction special interest group (CHISIG) of Australia on Computer-human interaction*, pages 1–10, 2005.

[25] David J. Field. Relations between the statistics of natural images and the response preperties of cortical cells. 4(12):2379–2394, December 1987.

[26] Daniel L. Ruderman. The statistics of natural images. *Network: Comput. Neural Syst.*, 5:517–548, 1994.

[27] Antonio Torralba and Aude Oliva. Statistics of natural image categories. *Network: Comput. Neural Syst.*, 14:391–412, January 2003.

[28] Erik Reinhard, Greg Ward, Sumanta Pattanaik, and Paul Debevec. *High Dynamic Range Imaging*. Morgan Kaufmann Publishers, 2005.

[29] S. Pattanaik, J.A. Ferwerda, M.D. Fairchild, and D. Greenberg. A multiscale model of adaptation and spatial vision for realistic image display. In *Proc. of ACM SIGGRAPH conf.*, 1998.

[30] Jack Tumblin and Greg Turk. LCIS: a boundary hierarchy for detail-preserving contrast reduction. In *Proc. of ACM SIGGRAPH conf.*, 1999.

[31] Erik Reinhard, Michael Stark, Peter Shirley, and James Ferwerda. Photographic tone reproduction for digital images. *ACM Trans. on Graphics*, 21(3), 2002. Proc. of ACM SIGGRAPH conf.

[32] Frédo Durand and Julie Dorsey. Fast bilateral filtering for the display of high-dynamic-range images. *ACM Trans. on Graphics*, 21(3), 2002. Proc. of ACM SIGGRAPH conf.

[33] Raanan Fattal, Dani Lischinski, and Michael Werman. Gradient domain high dynamic range compression. *ACM Trans. on Graphics*, 21(3), 2002. Proc. of ACM SIGGRAPH conf.

[34] Yuanzhen Li, Lavanya Sharan, and Edward H. Adelson. Compressing and companding high dynamic range images with subband architectures. *ACM Trans. on Graphics*, 24(3), 2005. Proc. of ACM SIGGRAPH conf.

[35] Joe Geigel and F. Kenton Musgrave. A model for simulating the photographic development process on digital images. In *Proc. of ACM SIGGRAPH conf.*, 1997.

[36] Aaron Hertzmann, Charles E. Jacobs, Nuria Oliver, Brian Curless, and David H. Salesin. Image analogies. In *Proc. of ACM SIGGRAPH conf.*, 2001.

[37] Iddo Drori, Daniel Cohen-Or, and Hezy Yeshurun. Example-based style synthesis. In *Proc. of IEEE conf. on Comp. Vision and Pattern Recognition*, 2003.

[38] D. DeCarlo and A. Santella. Stylization and abstraction of photographs. *ACM Trans. on Graphics*, 21(3), 2002. Proc. of ACM SIGGRAPH conf.

[39] Armin Bruderlin and Lance Williams. Motion signal processing. In *Proc. of ACM SIGGRAPH conf.*, 1995.

[40] Dani Lischinski, Zeev Farbman, Matt Uyttendaele, and Richard Szeliski. Interactive local adjustment of tonal values. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, pages 646–653, New York, NY, USA, 2006. ACM.

[41] Jiawen Chen, Sylvain Paris, and Frédo Durand. Real-time edge-aware image processing with the bilateral grid. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, page 103, New York, NY, USA, 2007. ACM.

[42] Zeev Farbman, Raanan Fattal, Dani Lischinski, and Richard Szeliski. Edge-preserving decompositions for multi-scale tone and detail manipulation. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2008)*, 27(3), August 2008.

[43] Raanan Fattal, Maneesh Agrawala, and Szymon Rusinkiewicz. Multiscale shape and detail enhancement from multi-light image collections. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 26(3), August 2007.

[44] B.K.P. Horn. Focusing. *Memo*, (10), May 1968.

[45] T. Darrell and K. Wohn. Pyramid based depth from focus. In *IEEE CVPR*, 1988.

[46] John Ens and Peter Lawrence. An investigation of methods for determining depth from focus. *IEEE Transactions on PAMI*, 15(2):97–108, 1993.

[47] S. K. Nayar and Y. Nakagawa. Shape from focus. *IEEE Transactions on PAMI*, 16(8):824–831, 1994.

[48] Samuel W. Hasinoff and Kiriakos N. Kutulakos. Confocal stereo. In *ECCV*, pages 620–634, 2006.

[49] A. P. Pentland. A new sense for depth of field. *IEEE Transactions on PAMI*, 9(4):523–531, 1987.

[50] Masahiro Watanabe and Shree K. Nayar. Rational filters for passive depth from defocus. *IJCV*, 27(3):203–225, 1998.

[51] Paolo Favaro and Stefano Soatto. Learning shape from defocus. In *ECCV*, pages 735–745, London, UK, 2002. Springer-Verlag.

[52] H. Jin and P. Favaro. A variational approach to shape from defocus. In *ECCV*, May 2002.

[53] Paolo Favaro and Stefano Soatto. A geometric approach to shape from defocus. *IEEE Transactions on PAMI*, 27(3):406–417, 2005.

[54] Ren Ng, Marc Levoy, Mathieu Bredif, Gene Duval, Mark Horowitz, and Pat Hanrahan. Light field photography with a hand-held plenoptic camera. April 2005.

[55] Anat Levin, Rob Fergus, Frédo Durand, and William T. Freeman. Image and depth from a conventional camera with a coded aperture. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, page 70, New York, NY, USA, 2007. ACM.

[56] Ashok Veeraraghavan, Ramesh Raskar, Amit Agrawal, Ankit Mohan, and Jack Tumblin. Dappled photography: mask enhanced cameras for heterodyned light fields and coded aperture refocusing. *ACM Trans. Graph.*, 26(3):69, 2007.

[57] Helmy A. Eltoukhy and Sam Kavusi. A computationally efficient algorithm for multi-focus image reconsction. In *SPIE Electronic imaging*, volume 3813, June 2003.

[58] Mehmet K. Özkan, A. Murat Tekalp, and M. Ibrahim Sezan. Pocs-based restoration of space-varying blurred images. *IEEE Transactions on IP*, 3(4):450–454, 1994.

[59] H. Joel Trussell and Sergei Fogel. Identification and restoration of spatially variant motion blurs in sequential images. *IEEE Transactions on IP*, 1(1):123–126, 1992.

[60] Stanley J. Reeves and Russell M. Mersereau. Blur identification by the method of generalized cross-validation. *IEEE Transactions on IP*, 1(3):301–311, 1992.

[61] Akira Kubota and Kiyoharu Aizawa. Reconstructing arbitrarily focused images from two differently focused images using linear filters. *IEEE Transactions on IP*, 14(11):1848–1859, November 2005.

[62] Shang-Hong Lai, Chang-Wu Fu, and Shyang Chang. A generalized depth estimation algorithm with a single image. *IEEE Transactions on PAMI*, 14(4):405–411, 1992.

[63] Michael Potmesil and Indranil Chakravarty. A lens and aperture camera model for synthetic image generation. In *SIGGRAPH '81: Proceedings of the 8th annual conference on Computer graphics and interactive techniques*, pages 297–305, New York, NY, USA, 1981. ACM Press.

[64] B.A. Barsky, D.R. Horn, S.A. Klein, J.A. Pang, and M. Yu. Camera models and optical systems used in computer graphics: Part ii, image based techniques. In *ICCSA*, 2003.

[65] Brian A. Barsky, Michael J. Tobias, Derrick P. Chu, and Daniel R. Horn. Elimination of artifacts due to occlusion and discretization problems in image space blurring techniques. *Graph. Models*, 67(6):584–599, 2005.

[66] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981.

[67] Jianbo Shi and Carlo Tomasi. Good features to track. In *CVPR*, pages 593–600, 1994.

[68] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *The Quarterly of Applied Mathematics*, 2, 1944.

[69] Donald Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics*, 11, 1963.

[70] JeanYves Bouguet. Camera calibration toolbox for matlab. http://www.vision.caltech.edu/bouguetj/calib_doc/, 2007.

[71] M.I.A. Lourakis. levmar: Levenberg-marquardt nonlinear least squares algorithms in C/C++. [web page] `http://www.ics.forth.gr/~lourakis/levmar/`, Jul. 2004.

[72] B. Caprile and V. Torre. Using vanishing points for camera calibration. *International Journal of Computer Vision*, 4(2):127–139, March 1990.

[73] Sudipta N Sinha, Jan-Michael Frahm, Marc Pollefeys, and Yakup Genc. Gpu-based video feature tracking and matching. In *EDGE, workshop on Edge Computing Using New Commodity Architectures*, 2006.

[74] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *JACM: Journal of the ACM*, 45, 1998.

[75] P. H. S. Torr and A. Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, pages 138–156, 2000.

[76] M. Brown, R.I. Hartley, and D. Nister. Minimal solutions for panoramic stitching. *CVPR*, pages 1–8, June 2007.

[77] Stan Birchfield. Klt: An implementation of the kanade-lucas-tomasi feature tracker. http://www.ces.clemson.edu/ stb/klt/, 2007.

[78] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transaction of the ASME – Journal of Basic Engineering*, 1960.

[79] Soonmin Bae, Sylvain Paris, and Frédo Durand. Two-scale tone management for photographic look. *ACM Transactions on Graphics*, 25(3):637–645, July 2006.

[80] Sylvain Paris and Frédo Durand. A fast approximation of the bilateral filter using a signal processing approach. In *Proc. of Eur. Conf. on Comp. Vision*, 2006.

[81] Prasun Choudhury and John Erwin Tumblin. The trilateral filter for high contrast images and meshes. In *Proc. of Eurographics Symp. on Rendering*, 2003.

[82] Sara Lee Su, Frédo Durand, and Maneesh Agrawala. De-emphasis of distracting image regions using texture power maps. In *Proc. of Int. Workshop on Texture Analysis and Synthesis*, 2005.

[83] Elmar Eisemann and Frédo Durand. Flash photography enhancement via intrinsic relighting. *ACM Transactions on Graphics*, 23(3), 2004. Proceeding of ACM SIGGRAPH conference.

[84] Georg Petschnigg, Maneesh Agrawala, Hugues Hoppe, Richard Szeliski, Michael F. Cohen, and Kentaro Toyama. Digital photography with flash and no-flash image pairs. *ACM Transactions on Graphics*, 23(3), 2004. Proceeding of ACM SIGGRAPH conference.

[85] David J. Heeger and James R. Bergen. Pyramid-based texture analysis/synthesis. In *Proc. of ACM SIGGRAPH conf.*, 1995.

[86] James H. Elder and Steven W. Zucker. Local scale control for edge detection and blur estimation. *IEEE Transactions on PAMI*, 20(7):699–716, 1998.

[87] Anat Levin, Dani Lischinski, and Yair Weiss. Colorization using optimization. *ACM Transactions on Graphics*, 23(3):689–694, 2004. Proceeding of ACM SIGGRAPH conference.

[88] Aaron Isaksen, Leonard McMillan, and Steven J. Gortler. Dynamically reparameterized light fields. *ACM Transactions on Graphics*, (3), 2000.