國立臺灣大學電機資訊學院資訊工程學研究所

碩士論文

Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Taiwan University

Master Thesis

利用多張模糊影像校正取得環場影像

Acquisition of a Panorama from Several Blurred Images

謝若元

Jo-Yuan Hsieh

指導教授：李明穗 博士

Advisor: Ming-Sui Lee, Ph.D.

中華民國 98 年 7 月

July, 2009

誌謝

能順利走完研究所的兩年，要感謝許多人。首先我要感謝我的父母，提供我這個機會讓我接受更高一層的教育，也在我需要的時候給予適時的關心。另外也要感謝我的女朋友蔡文嘉，在我低潮與無助的時候給予我鼓勵、支持及信心，讓我一直堅持下去。最後，我想感謝大嘴學長，在給我許多建議以及處理問題的細節上都幫助我不少。另外，我的高中同學沈金緯，雖然同在台大，但其實見面的機會不多，卻還是能在我人生的道路上給予指引。能完成這篇論文，我也要特別感謝我的指導教授李明穗老師以及實驗室的每一位成員對我的支持與鼓勵，謝謝大家。

中文摘要

在這篇論文中，我們針對兩張影像來探討，一張模糊以及一張清楚且經過幾何轉換的影像。我們試著拼接兩張影像並試著去模糊化。長久以來，由於去模糊是個在不適定的情況下求逆轉換的問題，所以對影像去模糊一直是個極具挑戰性的任務。近年來不論是利用多張或是單張影像的去模糊法都被廣泛的提出討論，其中這些方法又可被分為兩類：盲去模糊法及非盲去模糊法。如果模糊核在去模糊的過程中為未知，則被稱為盲去模糊；反之，若模糊核在去模糊過程中被假設為已知前提，則稱之為非盲去模糊。為了算出模糊核，我們打算利用兩張影像中清楚的那張來幫助計算。理論上，直接將這兩張影像接在一起便可得到重疊的兩塊區域，一塊為模糊而一塊為清楚。藉由這重疊且被對在一起的兩塊區域，我們能夠算出影像的模糊核。但是實際上由於模糊的影像有些資訊已經被破壞，直接將兩張影像接在一起無法得到準確的重疊，因此我們決定先將模糊影像在拼接前先經過一次去模糊，將去模糊後的影像與清楚影像相接，並記其錄轉換參數，再將模糊影像和清楚影像以方才紀錄的參數相接，便可得到準確的重疊區塊。也因此模糊核能夠被準確的計算出來。最後便可以利用被計算出來的模糊核以非盲去模糊法來回復。

關鍵字：影像去模糊、影像拼接、全景畫。

ABSTRACT

In this thesis, we try to stitch one clear image with a blurred image relative to a geometric transformation, and then recover the blurred image in the meantime. Image deblurring has long been a challenging work since it is an ill-posed inverse problem. Deblurring methods using multiple or single image are both discussed in recent years. The deblurring is called blind if the kernel is unknown or non-blind if the kernel is known a priori. In order to estimate the blur kernel, we try to take the information from the non-blurred patch for help. By stitching a blurred image with a non-blurred image using Speeded-Up Robust Features (SURF), we can obtain the aligned overlapped patches. Ideally, we can estimate the blur kernel based on blurred/non-blurred patches. However, directly stitching blurred/non-blurred images leads to poor aligned patches. As a result, the kernel is misestimated and the image is incorrectly recovered. To solve this issue, a pre-deblurring as a pre-processing step of the blurred image is considered. We stitch the pre-deblurred image with the non-blurred image and record the transformation parameters for temporary. After that we stitch the original blurred image with the non-blurred image using the recorded parameters to get better-aligned patches. Now the two patches are much better-aligned than before so that the kernel can be correctly estimated. Finally, promising result using progressive inter-scale and intra-scale deconvolution is presented.

Index Terms — Image deblurring, image stitching, panorama.

CONTENTS

LIST OF FIGURES

Chapter 1 Introduction

1.1 Motivation

Conventional handheld digital cameras have made an exploration recent years. People are used to take pictures around and share with others on the Internet. The most common issue is image blur due to camera shake. Shake could occur when taking pictures on a moving vehicle or under long exposure time. Unfortunately, many photographer capture ephemeral moments that cannot be recapture again. Panoramas are popular in recent years thanks to the development of reliable distinctive image features. In such case image blur is highly prone to occur due to the camera shake since one needs to take a series of photographs with moving camera. Image deblurring has long been a challenging work as it is an ill-posed inverse problem and deblurring using one single image is more challenging. In panorama, we observed that there are overlapped regions in two stitched images. If one image is blurred by camera shake, we can take the information from non-blurred overlapped region to help deblur the blurry image. As a result, directly stitching of blurred/non-blurred images to get overlapped information is our first try. Nonetheless, directly stitching blurred/non-blurred images is not a good idea.

1.2 Problem Statement

A naïve solution is to simply stitch two images with one blurred and one non-blurred, and then extract the overlapped information to help deblur the blurred one. However, there is a fundamental problem that causes the following deblurring work fail. Direct stitching of blurred/non-blurred images actually produces poor stitching result. Once there is a large error in stitching step, the overlapped regions are misaligned. Incorrect aligned regions induce terrible results. Hence, a pre-deblurring step is required for a better stitching to get better aligned regions. The pre-processing step adopts a single image kernel estimation method to calculate the blur kernel for the use of deblurring. After the image is deblurred, we again stitch pre-deblurred/non-blurred images together. Remember, the goal of pre-deblurring is to get better-aligned patches of blurred/non-blurred pair, not pre-deblurred/non-blurred pair. Therefore, we extract the transformation parameters in the pre-deblurred/non-blurred images stitching step first. Then we stitch blurred/non-blurred images by the recorded parameters so that the following deblurring steps work well.

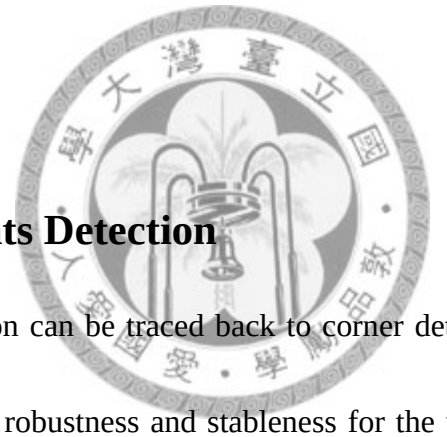
Finally, blurred image is recovered by use of overlapped patches in image stitching and stitching is again performed using the final output image.

1.3 Thesis Organization

The organization of the thesis is as follows. Chapter 2 introduces how interest points are detected and what the blurring model is. Then we review the related work about image deblurring; both blind and non-blind image deconvolution. Chapter 3 briefly explains the methods we adopted for image stitching and image deblurring. Chapter 4 first gives an overview of our system then describes our work on two cases for comparison. Chapter 5 presents some experiments results which tested on 4 types of geometric transformations with two different kernels. Finally, we conclude our thesis and have a discussion in chapter 6.

Chapter 2 Related Work

In this chapter, we discuss the related work of two categories, interest point detection and image deblurring. For the former, we introduce what interest points are and different approaches to detect them. For the latter, we discuss the degrading model of image blur and the most common artifacts in digital photography caused by camera shake. Then we give an overview about how to recover an non-blurred image from one or more blurred images.



2.1 Interest Points Detection

Interest point detection can be traced back to corner detection in computer vision which were obtained with robustness and stableness for the purpose of object tracking or recognition. Traditional corner detection such as Moravec corner detection and Harris corner detection [5] were proposed. Recently a more popular and robust interest point detection is proposed such as SIFT [7].

2.1.1 Moravec Corner Detection

The main idea of Moravec corner detector is based on the sum of squared difference (SSD) in a local shifting window in four directions (horizontal, vertical and

on the two diagonals). If the SSD varies largely in all directions then this points can be seen as a good feature. Hence, the minimum of SSDs in the window is selected among four directions as its strength. The interest point is detected if it is a local maximum. This approach contains several drawbacks:

1. The response map is noisy because of box filter (a binary window function). Since response map is not a smoothed image, there may be many local maxima.
2. Only four directions are considered.
3. Strong response to edge because only minimum of SSDs among directions is considered.

Thus, Harris made some improvements of Moravec corner detection.

2.1.2 Harris Corner Detection

Focused on the three drawbacks, some improvements are reached:

1. Use Gaussian mask rather than box filter to eliminate noise.
2. Consider all small shifts by Taylor's expansion rather than only 4 directions.
3. A new measurement using eigenvalue analysis with an ellipse instead of minimum of SSDs.

The new measurement of corner response requires the computation of eigenvalue

of a matrix M described in [5]. Therefore, a more efficient way to measure the corner response is to calculate the difference of $det(M)$ and $(trace(M))^2$.

2.1.3 Scale Invariant Feature Transform (SIFT)

Although the Harris corner is invariant to rotation and intensity change, it is not invariant to scale change which limits its applications. In recent years scale invariant feature transform (SIFT) was proposed by Lowe *et al.* [7] which further improves the Harris corner detector. In order to be invariant to scale change, SIFT is implemented in scale space. SIFT transforms an image to a set of feature points. There are four steps in SIFT:

1. Extrema detection in scale space using Difference of Gaussian (DoG), where DoG is actually an approximation of Laplacian of Gaussian (LoG).
2. Accurate keypoints localization by removing low contrast points and edge responses.
3. For each interest point, SIFT uses the information around it to generate a reproducible orientation which is assigned against rotation deformation.
4. Local image description based on the location, scale and orientation extracted above.

Once the description is constructed, a 128-dimension vector is formed for every interest

point. This high dimensional vector is highly distinctive for applications such as image stitching, object recognition, etc. On the contrary, the speed of matching features is dropped due to high dimensional vectors.

2.2 Image Deblurring

Motion blur from camera shake is a common artifact in digital photography. In many situations it is hard to hold the camera still when taking pictures around. In panorama, we need to take series of pictures for a big view which is prone to camera shake. Recovering a non-blurred image from a single image or multiple images has been discussed recent years. Most research focus on shift-invariant kernel which reduces image deblurring to image deconvolution. The observed blurred image B is the convolution of an unknown latent image I with an unknown kernel K:

$$B = I \otimes K + n,$$

where $\otimes$ is the convolution operator and n is the additive noise. Usually n is ignored. The deblurring methods can be categorized into two types: blind deconvolution and non-blind deconvolution.

2.2.1 Blind Image Deconvolution

It is obviously a more challenging work when both the blur kernel and latent

images are unknown. Basically blind deconvolution works only for low frequency blur kernels, e.g., smooth shape of motion. Recent work like Fergus *et al*. [4] used natural image statistics as a prior together with a sophisticated variational Bayes inference algorithm to estimate the kernel. This approach is not robust enough, however, it requires only one single image for kernel estimation. We adopted it as a pre-processing step in our system. Some techniques make the problem more tractable with additional inputs. Methods taking multiple images as inputs to perform blur kernel estimation have been proposed in [10]. The correlation among blurred images is utilized based on the assumption that all blurred images came from the same latent image. Even though good results are achieved using multiple images, they are limited to simple directional motion. Another method using a pair of blurred/noisy images proposed by Yuan *et al*. [14] takes the information from noisy image to help recover the blurred image. An iterative residual deconvolution is developed to significantly reduce deconvolution artifacts.

2.2.2 Non-blind Image Deconvolution

The blur kernel is assumed known a priori so the main task becomes recovering the latent image. A common method is Richardson-Lucy (RL) deconvolution [8][11], which recovers the latent image iteratively based on the modeling of image noise using Poisson distribution. Deconvolution artifacts, such as ringing effects or color speckles,

are inevitable because of high frequency loss in the blurred image. Regularization on deblurring tries to suppress the artifacts, however, edges are meanwhile suppressed. A total variation regularized Richardson-Lucy [2] is proposed to suppress ringing while preserving edges, but only gray-level images with simple geometric figures are shown in their results. Yuan *et al*. [15] proposed a progressive deconvolution in scale space successively suppress the ringing effects while preserving large-scale edges. A further refinement to add detail layer produce promising result.

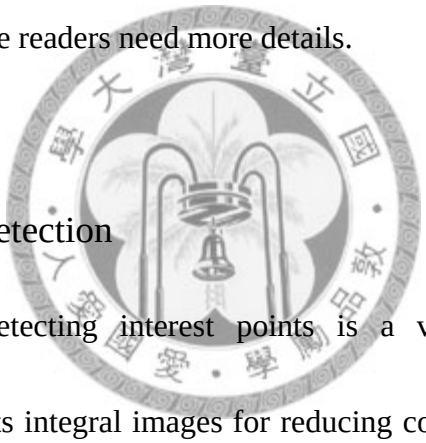
Chapter 3 Background Knowledge

We need to stitch images to get a panorama and also deblur the blurry image to recover it. Therefore there are two essential steps in our system, image stitching and image deblurring. For more self contained, we introduce the methods we adopted as background knowledge of our work. In image stitching, we adopted Speeded-Up Robust Feature (SURF) [1] for feature based image stitching. The estimation of transformation parameters utilizes a weighted least square with Expectation Maximization (EM) algorithm. Two approaches are adopted in image deblurring step, one for kernel estimation and one for image deconvolution. A least square estimation using Tikhonov regularization proposed by Yuan *et al*. [14] provides a good estimated kernel for image deblurring. Progressive inter-scale and intra-scale image deconvolution proposed by Yuan *et al*. [15] works well in suppressing ringing artifacts while preserving large scale edges. It is applied as non-blind image deconvolution based on the estimated blur kernel. Two components for image stitching are described in 4.1 and 4.2, other two components of image deblurring are introduced in 4.3 and 4.4.

3.1 Speeded-Up Robust Feature (SURF)

To stitch images, feature points detection in images is required. We estimate the

transformation between images based on the detected feature points. After that one image is transformed using the estimated parameters to stitch with the other. Each feature point is actually a description described by the descriptor. Before that, we need to find the candidate of the feature point first. Conceptually, there are two steps, detection and description, for extracting feature points. Matching is applied on two sets of feature points for further application. In total, these are detection, description, and matching steps. The figures we presented in this section are directly taken from [1] for the ease of referencing if the readers need more details.



3.1.1 Interest Point Detection

The approach for detecting interest points is a very basic Hessian-matrix approximation which adopts integral images for reducing computation time drastically. We first briefly introduce the concept of integral image and Hessian-matrix based interest points then describe how scale space is presented.

A. Integral Images

Integral images allow fast computation of box type convolution filters. Every pixel of an integral image I_Σ constructed from image I is the sum of all pixel values within the rectangle formed by the origin and $\mathbf{x}$ at a location $\mathbf{x} = (x, y)$.

$$I_{\Sigma}(\mathbf{x}) = \sum_{i=0}^{i\leq x}\sum_{j=0}^{j\leq y} I(i,j)$$

Once the integral image is computed, it takes only three additions to compute the sum of a rectangle area as shown in Fig. 3.1.

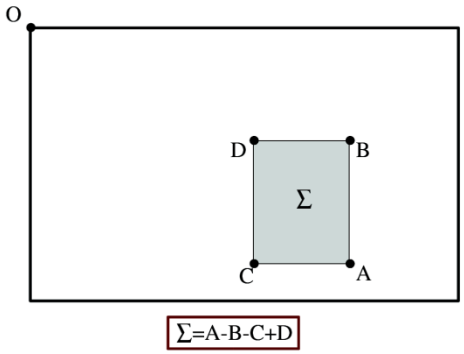


Fig. 3.1 It takes only three additions to calculate the sum of values within a rectangular region using integral images.

B. Hessian Matrix Based Interest Points

The interest point detector is based on the Hessian matrix. Given a point $\mathbf{x} = (x, y)$, the Hessian matrix $\mathcal{H}(\mathbf{x}, \sigma)$ at location $\mathbf{x}$ in scale σ is defined as:

$$\mathcal{H}(\mathbf{x}, \sigma) = \begin{bmatrix} \mathcal{L}_{xx}(\mathbf{x}, \sigma) & \mathcal{L}_{xy}(\mathbf{x}, \sigma) \\ \mathcal{L}_{yx}(\mathbf{x}, \sigma) & \mathcal{L}_{yy}(\mathbf{x}, \sigma) \end{bmatrix},$$

where $\mathcal{L}_{xx}(\mathbf{x}, \sigma)$ is the convolution of the Gaussian second order derivative $\frac{\partial^2}{\partial x^2} g(\sigma)$ and image I at location $\mathbf{x}$, similarly for $\mathcal{L}_{xy}(\mathbf{x}, \sigma)$ and $\mathcal{L}_{yy}(\mathbf{x}, \sigma)$. Here a box filter is constructed to approximate the Gaussian second order derivative, as shown in Fig. 3.2. Thus, computation time is largely reduced by using integral images and the

approximated Gaussian second order derivative.

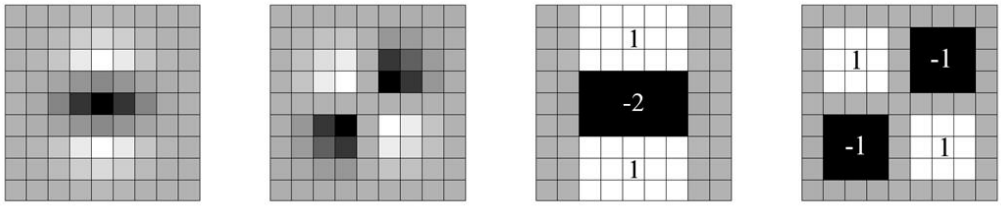


Fig. 3.2 Left to right: the Gaussian second order partial derivative in *yy*- ($\mathcal{L}_{yy}$) and *xy*-direction ($\mathcal{L}_{xy}$); the approximated version of $\mathcal{L}_{yy}$ and $\mathcal{L}_{xy}$. The gray area are zero.

C. Scale Space Representation

The interest points need to be detected at different scales against the variance of scale change. All interest points will be localized in the image and over scales. Usually, scale spaces are implemented as an image pyramid. Since SURF utilizes box filters and integral images, filters of any size on the original image are applied instead of iteratively scaling down the input image.

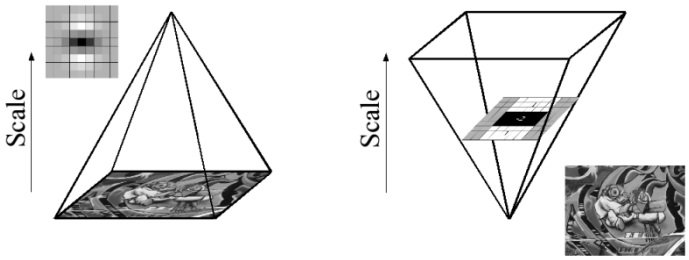


Fig. 3.3 Left: Iteratively reducing the image size in traditional image pyramid.

Right: Increasing the filter size while keep the original image unchanged.

In fact, the scale space consists of several octaves. An octave is a series of Hessian response maps obtained by convolving the same input image with filters of increasing size. In total, the scaling factor increases to a number of 2 when an octave is complete. Finally, an interest point is detected if it is a local maximum.

3.1.2 Interest Point Description

The interest point descriptor describes the distribution of gradients around the interest point neighborhood. The description is built on the distribution of first order Haar wavelet responses in *x* and *y* direction. First, a reproducible orientation is assigned to an interest point according to the information from a circular region around it. Then a square region is constructed to as a SURF descriptor.

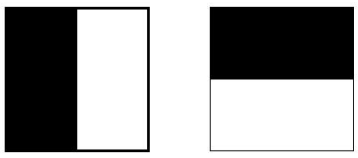


Fig. 3.4 Haar wavelet masks used to compute the response in *x* and *y* directions.

Dark area represents -1 while white area represents +1.

A. Orientation Assignment

In order to be invariant to image rotation, a reproducible orientation needs to be calculated for the interest points. Haar wavelet responses in *x* and *y* directions are

calculated in a circular neighborhood around the interest point. As shown in Fig. 3.4, integral images are again utilized for efficient computing. After the wavelet responses are obtained and weighted by a Gaussian, the responses of directions *x* and *y* are combined as 2D vectors (i.e., points). A dominant orientation is selected as follows. As shown in Fig. 3.5, we calculate the sum of all responses within a sliding orientation window. Horizontal and vertical responses are summed in the window to form an orientation vector. The orientation of the interest point is selected by the longest vector over all windows.

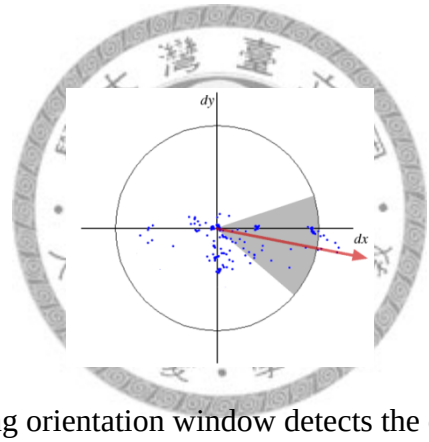


Fig. 3.5 A sliding orientation window detects the dominant orientation.

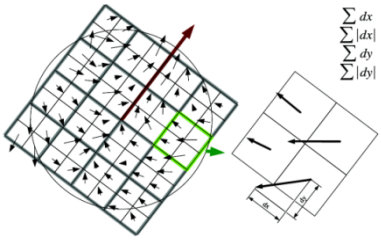


Fig. 3.6 Left: an oriented window formed of 4 x 4 sub-regions. Right: for each sub-region we compute $\sum d_x$, $\sum d_y$, $\sum |d_x|$, $\sum |d_y|$ of it.

B. Descriptor bases on the sum of Haar Wavelet Responses

The first step of making a description is to construct a square region centered at the interest point and oriented along the assigned orientation. The square region is divided into smaller 4 x 4 sub-regions in which we compute the Haar wavelet responses. The Haar wavelet response in horizontal and vertical direction are denoted as d_x and d_y. which are weighted by a Gaussian centered at the interest point for the robustness of geometric deformations and localizations errors. We sum up the wavelet responses d_x and d_y for each sub-regions to get the first part of the feature vector. Also, we sum up the absolute values of the responses $|d_x|$ and $|d_y|$ to get the rest part of a feature vector. Hence, a four-dimension vector $\boldsymbol{v}$, $\boldsymbol{v} = (\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|)$, is generated for every sub-region. Concatenating $\boldsymbol{v}$ for all 4 x 4 sub-regions produces a 64-dimension descriptor vector, i.e., feature vector.

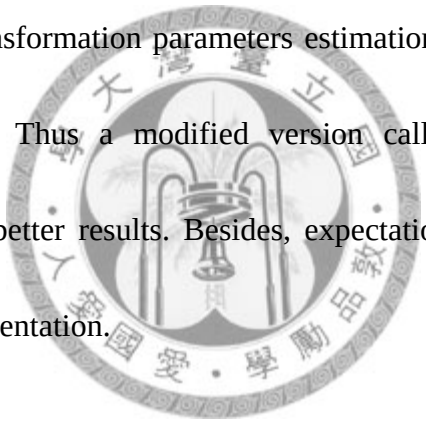
3.1.3 Matching

For any two images, we need to match the extracted feature vectors for the purpose of estimating transformation parameters. The approach is actually quite simple. A nearest neighbor method in feature space matching is adopted. However, not every feature point is correctly detected in the other image or even not detected at all. In such case assigning every feature points a nearest neighbor would fail the following work. So instead of assigning a nearest neighbor to each detected feature point, we assign a

nearest neighbor to a feature point only if the Euclidian distance is 0.7 times closer than the second nearest neighbor.

3.2 Weighted Least Square Estimation of Transformation Parameters Using Expectation Maximization Algorithm

In the previous section we extract the features of images and we also found the matches between two sets of features. In this section, we estimate the transformation between two images using the matches we found by least square estimation. In our experiments, results of transformation parameters estimation using simple least square method are unacceptable. Thus a modified version called weighted least square estimation is adopted for better results. Besides, expectation maximization algorithm (EM) is used in our implementation.



3.2.1 Least Square Estimation of Transformation Parameters

For the stitching of any two different images sharing partial overlapped scene from different angles of view we take perspective transformation as our default transformation model. Perspective transformation contains 8 degrees of freedom which is sufficiently enough to encompass traditional transformations such as translation, rotation and scaling. In 2D planar system, the coordinates of a transformed pixel located at $\mathbf{x} = (x, y)^T$ can be expressed as:

$$X = \frac{ax + by + c}{gx + hy + 1}$$
$$Y = \frac{dx + ey + f}{gx + hy + 1},$$

where X and Y are the transformed x and y, a to h is the 8 degrees of perspective transform which are the parameters we want to estimate. By rearranging the above equations, we obtain a clearer version for the convenience of least square estimation.

$$X = xa + yb + c + 0d + 0e + 0f - Xxg - Xyh$$

$$Y = 0a + 0b + 0c + xd + ye + f - Yxg - Yyh$$

Denote $\mathbf{x}' = (X, Y)^{\mathrm{T}}$ as a vector then we arrive a system of product of a matrix A and a vector p:

$$\begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -Xx & -Xy \\ 0 & 0 & 0 & x & y & 1 & -Yx & -Yy \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{bmatrix} = \begin{bmatrix} X \\ Y \end{bmatrix}$$

As we know the approximated solution of p by least square estimation is derived by

$$A^T Ap = A^T \mathbf{x}'.$$

However, simple least square estimation produces poor results, so a weighted least square estimation is adopted for better results and it is introduced in the following section.

3.2.2 Weighted Least Square Estimation using EM Algorithm

As discussed above, simply least square estimation of parameters gives poor results, therefore, the weighted least square estimation of parameters using EM algorithm performs as follows. We denote every matched points as $(\mathbf{x}_i, \hat{\mathbf{x}}_i)$, where i is the index in the matched points. Also, we extend matrix multiplication of the above transformation to n points.

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -X_1x_1 & -X_1y_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -Y_1x_1 & -Y_1y_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -X_2x_2 & -X_2y_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -Y_1x_1 & -Y_2y_2 \\ & & & & \vdots & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -X_nx_n & -X_ny_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -Y_1x_1 & -Y_ny_n \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{bmatrix} = \begin{bmatrix} X_1 \\ Y_1 \\ X_2 \\ Y_2 \\ \vdots \\ X_n \\ Y_n \end{bmatrix}$$

1. Initial step: least square estimation of parameters p.
2. E-step: transform $\mathbf{x}_i$ to $\mathbf{x}_i'$ using the estimated parameters, calculate the weights w_i for each point by a Gaussian centered at $\hat{\mathbf{x}}_i$ using the distance obtained from the error $|\hat{\mathbf{x}}_i - \mathbf{x}_i'|$.
3. M-step: estimate parameters with weighted least square.

In E-step, we first calculate the weights for weighted least square estimation. The weighting matrix should be a form like:

$$\begin{bmatrix} w_1 & 0 & \cdots & 0 & 0 \\ 0 & w_1 & & 0 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & & w_n & 0 \\ 0 & 0 & \cdots & 0 & w_n \end{bmatrix}_{2n \times 2n}$$

Every point $\mathbf{x}_i$ is transformed using the estimated parameters so that we can get the

weights based on a Gaussian distribution. At M-step, weighted least square estimation is performed to get better and more accurate parameters based on

$$A^T W A p = A^T W \mathbf{x}',$$

where W is a weighting diagonal matrix of w_i, $i = 1 \ldots n$. We iteratively perform E-step and M-step until convergence. Finally, one image is transformed using the final estimated parameters to stitch with the other.

3.3 Kernel Estimation

According Yuan *et al.* [14], a good kernel can be obtained using a simple constrained least-squares optimization if we have two images of the same scene, a blurred one (P_1) and a clear one (P_2). In our system, two overlapped image patches can be attained in the previous chapter when stitching.

3.3.1 Kernel Estimation using Tikhonov Regularization

The goal of kernel estimation is to find the blur kernel K from $B = I \otimes K$ with the initialization $I = P_2$. The above blurring equation can be rewritten in vector-matrix form as $b = Ak$, where b and k are the vector forms of blurred image B and blur kernel K, and A is the matrix form of I. To obtained optimal K, Tikhonov regularization is used with a positive scalar λ by solving $min_k ||Ak - b||^2 + \lambda^2 ||k||^2$.

Here λ is the regularization strength. An explicit solution is given by $(A^T A + \lambda^2 I)k = A^T b$. However, in fact, a blur kernel needs to be non-negative and preserve energy, so some additional constraints is added and we can get optimal K by the following optimization system:

$$min_k ||Ak - b||^2 + \lambda^2 ||k||^2, \text{subject to } \mathrm{k_i} \geq 0, \text{and } \sum_{\mathrm{i}} \mathrm{k_i} = 1.$$

After the kernel is estimated, image deconvolution is applied.

3.4 Image Deconvolution

The deblurring method we adopted here can be traced back to a classical non-blind image deblurring algorithm called Richardson-Lucy deconvolution, which is proposed by Lucy [8] and Richardson [11]. The main and the most common artifact is the appearance of ringings arisen with iterative method. We first review Richardson-Lucy (RL) deconvolution algorithm. Then we describe how progressive inter-scale and intra-scale image deconvolution works [15].

3.4.1 Richardson-Lucy (RL) Algorithm

Richardson-Lucy (RL) is an algorithm for deblurring if the kernel is known a priori, i.e., it is a non-blind image deconvolution method. The image noise is modeled as a Poisson noise distribution and the likelihood probability of the image *I* is expressed as:

$$p(B|I) = \prod_x \frac{(I \otimes K)(x)^{B(x)} e^{-((I \otimes K)(x))}}{B(x)!},$$

where x indicates the position of each pixel, and $p(B|I) = \text{Poisson}((I \otimes K)(x))$ is a Poisson process for each pixel x. For simplicity, x is omitted in the following article. In order to obtain the optimal *I* that satisfied maximum likelihood $p(B|I)$, we take log and negative operations on $p(B|I)$ to get the following energy function:

$$E(I) = \sum \{(I \otimes K) - B \cdot log[(I \otimes K)]\}.$$

So our goal becomes minimizing the above energy function $E(I)$:

$$I^* = \arg\min_I E(I)$$

A common artifact in RL algorithm is ringing artifacts, dark and light ripples around strong edges in the reconstructed image. More iterations produce not only more image details but more ringings.

3.4.2 Bilateral Richardson-Lucy (BRL) Algorithm

The ringing artifacts are usually proportional to the strength of edge jump at discontinuous points and the amplitude of ringing will decrease away from the edge. Human eyes are sensitive to abrupt difference, for example, edges. As a result, the ringing effects are most conspicuous in smooth areas. Besides, human perception can tolerate small scale ringing in high frequency regions. Based on this observation, an energy function $E_B(I)$ is added as a regularization term to reduce ringing while

preserving large-scale edges:

$$I^* = arg\ \min_I [E(I) + \lambda E_B(I)],$$

where λ is the regularization factor. The added term $E_B(I)$ is defined as:

$$E_B(I) = \sum_x \sum_{y \in \Omega} f(|x-y|)\rho(|I(x)-I(y)|),$$

where Ω is a spatial support controls the size of filters, function $f(\cdot)$ is the spatial filter and function $\rho(\cdot)$ is a penalty function. Both filters are defined as follows:

$$f(|x-y|) = e^{-\frac{|x-y|^2}{2\sigma_s}}.$$

$$\rho(|I(x)-I(y)|) = 1 - e^{(-\frac{|I(x)-I(y)|^2}{2\sigma_r})}.$$

Function $\rho(\cdot)$ gives a large but limited penalty on the image difference $I(x)-I(y)$ and function $f(\cdot)$ puts weights on the penalty using a Gaussian distribution centered at *x*. In fact, the penalty occurs when the current processing pixel contains large difference, i.e., edge.

3.4.3 Progressive Inter-scale Scheme

The main idea of progressive deconvolution is to take the output of previous iteration as a guide to help the next deconvolution. In inter-scale scheme, the recovered image in one scale can be used as a guide image for next scale because the recovered image in coarser scale provides more useful edge information. Continue this process from coarse to fine, better result at the finest scale is obtained.

First, a pyramid $\{B^l\}_{l=1}^L$ of a full resolution blurred image B and a pyramid

$\{K^l\}_{l=1}^{L}$ of a blur kernel K are built using bicubic downsampling. For iteration $l = r$, we upsample the recovered image I^{l-1} from the previous scale. Take this upsampled image and B^l as inputs to generate a deblurred image I^l using the Joint Bilateral Richardson-Lucy (JBRL) which is modified based on BRL. We will introduce how JBRL works in the next section.

3.4.4 Joint Bilateral Richardson-Lucy (JBRL) Algorithm

The upsampled recovered image I^{l-1} is denoted as I^g, the guide image. Now the regularization term $E_B(I)$ in BRL is modified by:

$$E_{JB}(I; I^g) = \sum_x \sum_{y \in \Omega} f(|x - y|) g'(|I^g(x) - I^g(y)|) \rho(|I(x) - I(y)|),$$

where $g'(|I^g(x) - I^g(y)|)$ is a range filter applied on the guide image I^g. Here $g'(\cdot)$ is also a Gaussian filter:

$$g'(|I^g(x) - I^g(y)|) = e^{\left(-\frac{|I^g(x) - I^g(y)|^2}{2\sigma_r^g}\right)}$$

The function $g'(\cdot)$ will decrease the regularization if there is a large image gradient in guide image I^g. Since the guide image contains more accurate information, the additional range filter could help obtain better result.

3.4.5 Progressive Intra-scale Scheme

Large regularization (λ) suppresses not only noticeable ringings but also details.

Therefore, a progressive intra-scale deconvolution is needed to recover image details step-by-step with decreasing regularization strength using an iterative residual deconvolution. Assume there is a difference ΔI between I^g and I, we can get a blurring equation as:

$$B = I \otimes K = (I^g + \Delta I) \otimes K = I^g \otimes K + \Delta I \otimes K,$$

or we can rewrite it as:

$$\Delta B = B - I^g \otimes K = \Delta I \otimes K.$$

Simply recovering ΔI from ΔB rather than directly recovering I from B is a better way because magnitudes in ΔB are smaller than B thus ΔI contains less ringing artifact than I. Now the optimal residual ΔI we want can be obtained by minimizing the following energy with the decreasing regularization strength λ during iterations:

$$\Delta I^* = arg\min_{\Delta I}\left[E(\Delta I) + \lambda E_{JB}(\Delta I; I^g)\right], \text{with } \lambda^{t+1} = \gamma\lambda^t,$$

where γ is a decay factor. Below we show a flow chart of progressive inter-scale and intra-scale deconvolution in Fig. 3.7. For more details we refer to [15].

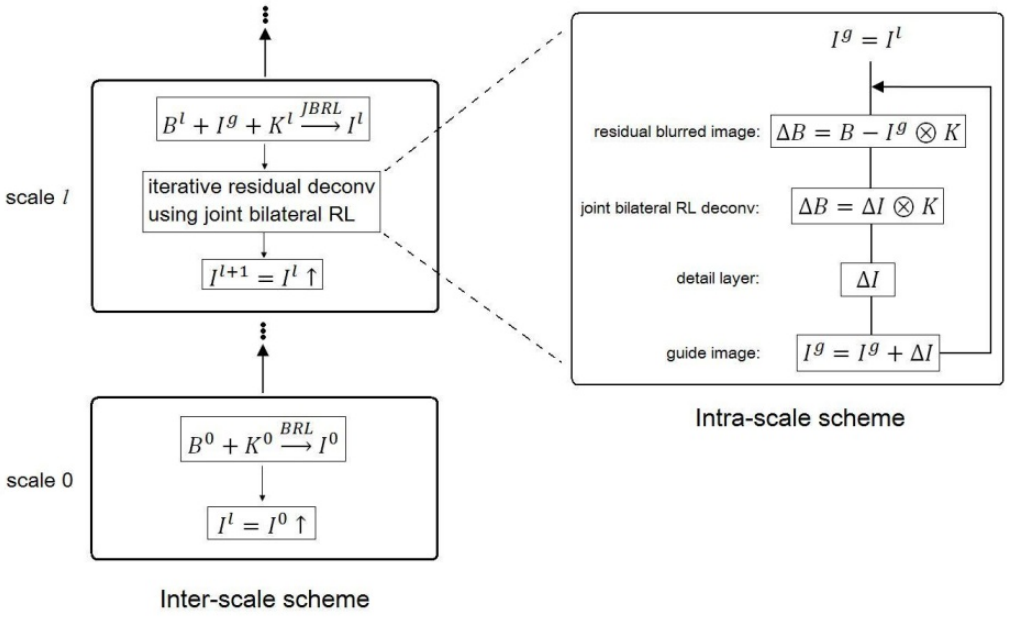


Fig. 3.7 Flow chart of the progressive deconvolution.

Chapter 4 Proposed System

In 4.1 we take a look of our work for two cases as an overview. More details and comparisons of both cases are discussed in 4.2 and 4.3.

4.1 System Overview

To get a panorama, images are stitched to generate a larger view. Usually, a series of pictures are taken to generate a panorama, but somehow camera shake inevitably happened. A direct method is to simply stitch the blurred image with the non-blurred image, extract overlapped regions, estimate the blur kernel and recover a non-blurred image using the estimated kernel. This is the most naïve and straightest approach yet in our experiments we found that this approach is not stable. As a result, another approach has to be derived to improve the naïve approach and make our system more robust.

Fig. 4.2 is an overview of our system and Fig. 4.1 is the naïve approach for comparison. As introduced previously, the inputs of the naïve approach are a pair of blurred/non-blurred images (B_1 and I_2) relative to a geometric transform. We first stitch the inputs to get the overlapped patches. The blur kernel is estimated once these two aligned blurred/non-blurred patches are obtained. Finally we perform deblurring using the estimated kernel.

The naïve approach leads to poor result for the reason of misaligned patches. We

decide to deblur the input B_1 first to get a better stitching result. As shown in Fig. 4.2, we take the transformation parameters from dB_1 and I_2 as the inputs of stitching of B_1 and I_2 to get better-aligned overlapped patches. Based on these patches a better estimation of blur kernel is reached. Satisfying result is attained in the final deblurring step using the estimated blur kernel.

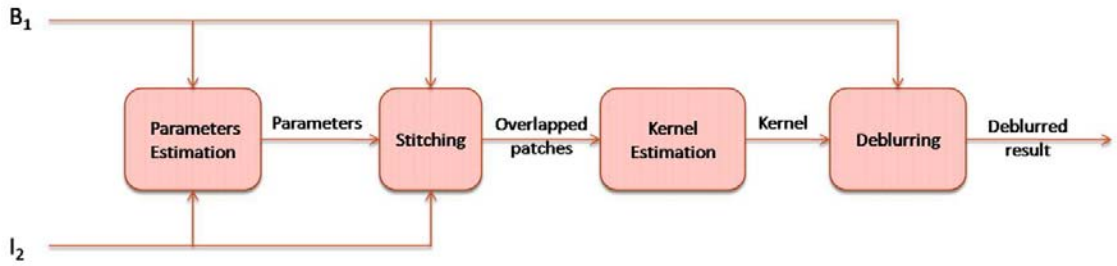


Fig. 4.1 Flow chart of the naïve approach

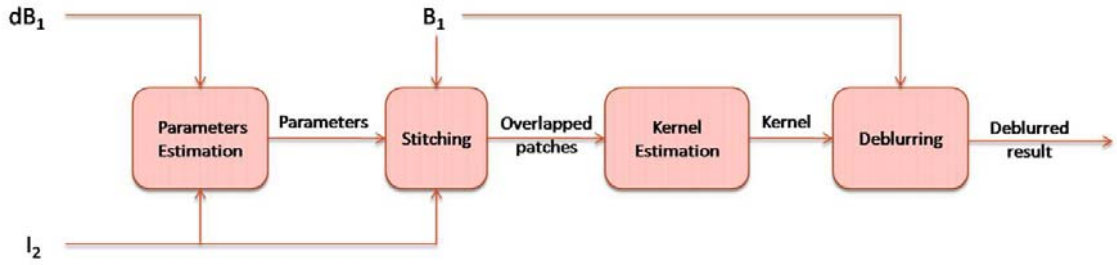


Fig. 4.2 Flow char of our system

4.2 Naïve approach (case1)

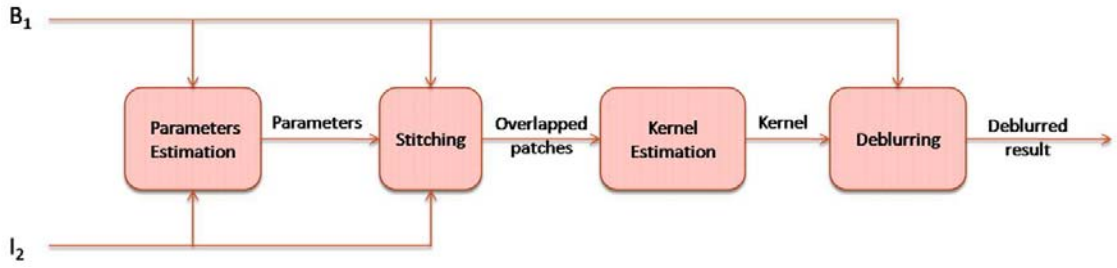


Fig. 4.3 Flow chart of naïve approach (case1).

In case1 we try to stitch a blurred image with a non-blurred image directly while recovering the blurred image in the meantime. In image stitching, we observed that two overlapped patches, a blurred one and a non-blurred one, of the same scene can be obtained. Based on these blurred/non-blurred overlapped regions, a good kernel can be estimated using the method we adopted which is described in Chapter 3. In such situation, both the latent image and observed blurred images are available. The kernel estimation therefore becomes more constrained to be solved. Once the blur kernel is attained, we adopted a progressive inter-scale and intra-scale image deconvolution to recover blurred image using the estimated kernel from coarse to fine. We denote the blurred image as B_1 and the non-blurred image as I_2. Here 5 steps are taken in the naïve approach (case1):

1. Calculate SURF feature points of B_1 and I_2 and then match them.

2. Estimate the transformation parameters between B_1 and I_2 by weighted lease square estimation using EM algorithm.
3. Stitch B_1 and I_2 using the estimated parameters and extract the overlapped regions P_1 and P_2.
4. Estimate the blur kernel using P_1 and P_2 with the method proposed by Yuan *et al.* [14]
5. Recover the blurred image with the estimated kernel using the non-blind image deconvolution method proposed Yuan *et al.* [15].

Unfortunately, the recovered image is far away from what we expect. The image is usually poorly reconstructed as the kernel is incorrectly estimated. The incorrectly estimated kernel can be imputed to the misaligned overlapped regions. In short, all these errors come from a fundamental problem; the images are ill-stitched, i.e., transformation parameters are inaccurately estimated since we directly stitch B_1 and I_2 together. To solve this issue, better-aligned patches are essential for kernel estimation which means that we have to figure out a way to stitch images more accurately. We decided to deblur the blurred image first as a pre-processing step for better stitching. Our experiments show that stitching pre-deblurred image with the non-blurred image yields better-aligned overlapped patches, thus, leads to better kernel estimation. Finally, satisfying deblurring result is obtained. An example of blurred/non-blurred images

related to a scaling change using the naïve approach (case1) is shown in Fig. 4.4.

(a) blurred image

(b) non-blurred image

(c) blurred/non-blurred stitching result

(d) blurred overlapped patch

(e) non-blurred overlapped patch

(f) true kernel

(g) estimated kernel

(h) deblurred result

Fig. 4.4 An example of scaling case using naïve approach.

4.3 Pre-deblur approach (case2)

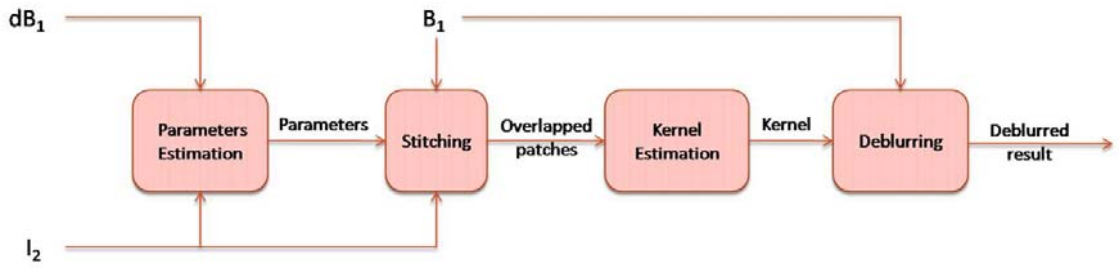


Fig. 4.5 Flow chart of our system (case2).

As described above, in order to pre-deblur the blurred image before stitching, the blur kernel has to be estimated without the help of non-blurred overlapped patch. In such condition, we adopted a single image kernel estimation method proposed by Fergus *et al.* [4] which utilizes the statistics of nature images. Once the blur kernel is estimated using Fergus' method, we apply progressive inter-scale and intra-scale image deconvolution [15] to recover the image. Even though the deblurred result is not good enough using single image kernel estimation, our stitching result is much improved than the naïve approach (case1). Hence, the overlapped regions are better-aligned. Now we can get a more correct kernel so the recovering process gives a more satisfying recovered image. Here we denote the pre-deblurred image as dB_1 and our pre-deblurred approach (case2) is composed of 10 steps as follows:

1. Single image kernel estimation using the method proposed by Fergus *et al.*

[4].

2. Pre-deblur the blurred image using the progressive inter-scale and intra-scale image deconvolution method proposed by Yuan *et al.* [15].

6. Calculate SURF feature points of dB_1 and I_2 and match them.

7. Estimate the transformation parameters between dB_1 and I_2 by weighted lease square estimation using EM algorithm.

8. Stitch B_1 and I_2 using the recorded parameters and extract the overlapped regions P_1 and P_2.

9. Estimate the blur kernel using P_1 and P_2 with the method proposed by Yuan *et al.* [14]

10. Deblur the blurred image using the estimated kernel.

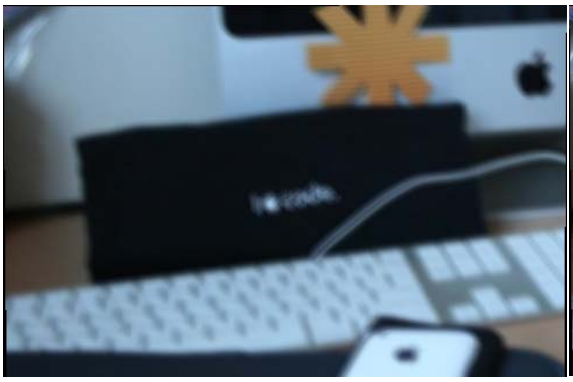
Different from case1, this time we stitch pre-deblurred/non-blurred images to get better stitching result. One thing should be noticed that the purpose of pre-deblurring is to get better transformation parameters. After pre-blurred/non-blurred images are stitched, the parameters are recorded for the stitching of blurred/non-blurred images. An example is presented using pre-deblur approach (case2) in Fig. 4.6. In the next chapter we will show more experiments results both using naïve approach (case1) and pre-deblur approach (case2) for comparison.

(a) Fergus' kernel

(b) pre-blurred image

(c) non-blurred image

(d) pre-blurred/non-blurred stitching result



(e) blurred overlapped patch

(f) non-blurred overlapped patch

(g) true kernel

(h) estimated kernel

(i) deblurred result

Fig. 4.6 An example of scaling transformation using pre-deblur approach.

Chapter 5 Experiment and Results

In this section we will show some experiments results. We first show the results of directly stitch blurry/non-blurred together in our system (the naïve approach). Better stitching results after a roughly pre-deblurring step are presented later. Four geometric transformations are considered in our experiments: Translation, Rotation, Scaling and Perspective. We test on two kernels of size 15 x 15 in our experiments.

(a) kernel1

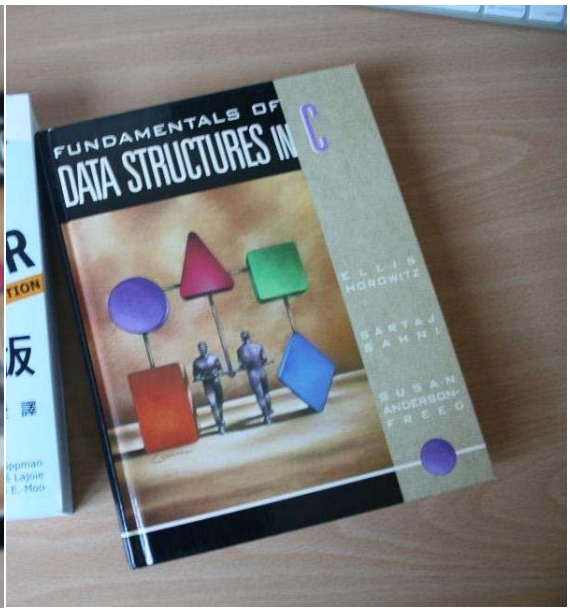
(b) kernel2

Fig. 5.1 Two kernels we test on.

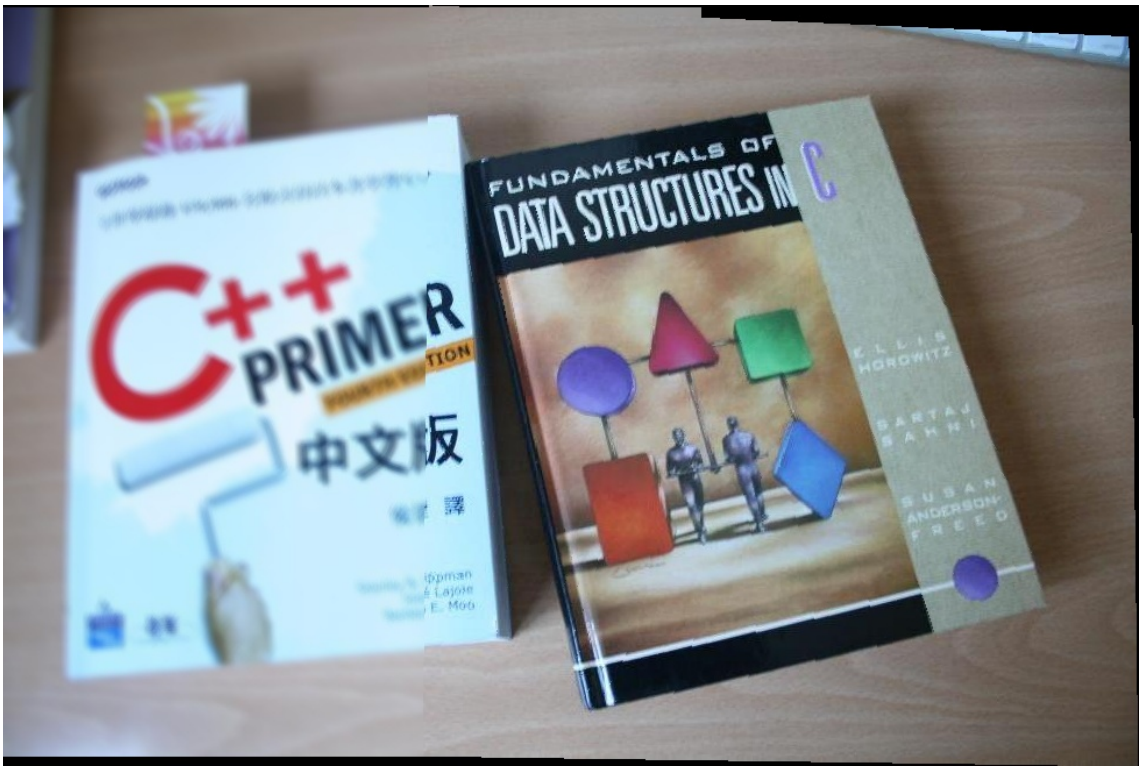
5.1 Deblur with Directly Stitching Blurred/Non-blurred images

Here we show outputs of translation and rotation case for kernel1, scaling and perspective case for kernel2.

(a) blurred image

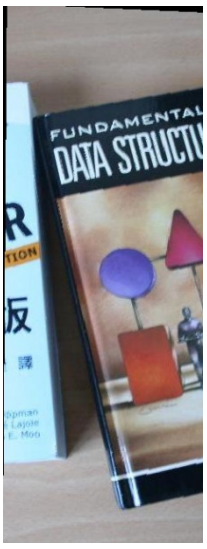


(b) non-blurred image



(c) blurred/non-blurred stitching result

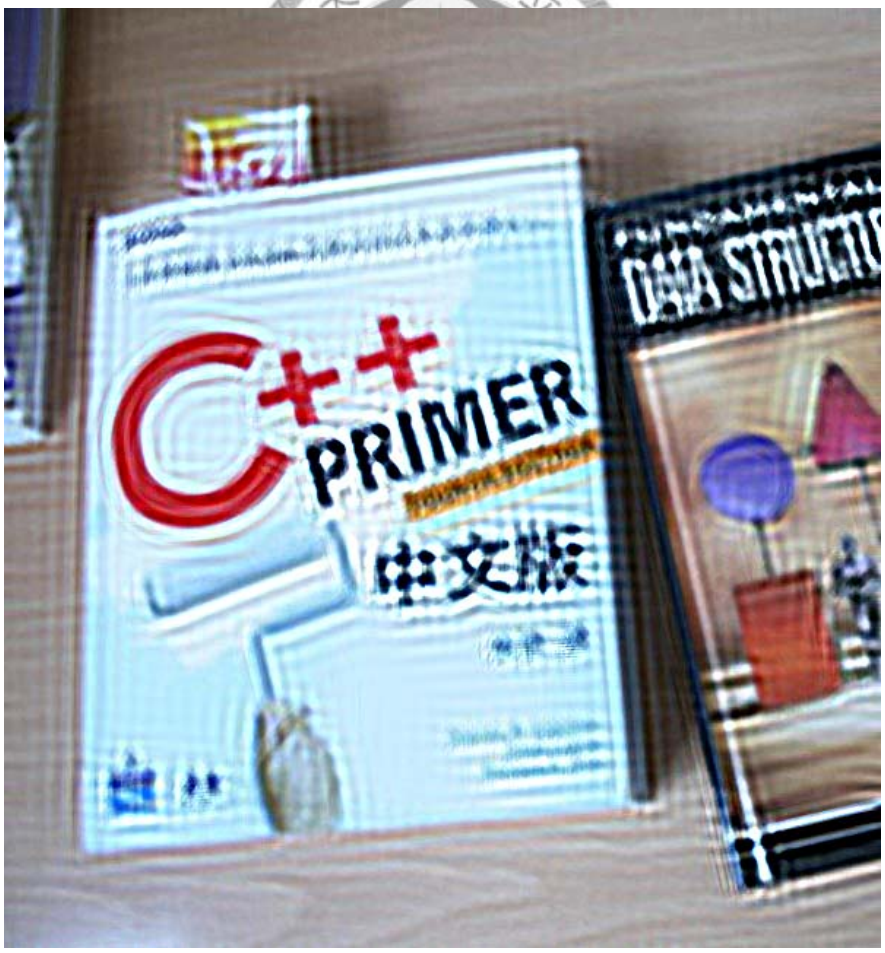
(d) blurred overlapped patch



(e) non-blurred overlapped patch

(f) true kernel

(g) estimated kernel



(h) deblurred result

Fig. 5.2 Case1: translation/kernel1

(a) blurred image

(b) non-blurred image

(c) blurred/non-blurred stitching result

(d) blurred overlapped patch

(e) non-blurred overlapped patch

(f) true kernel

(g) estimated kernel

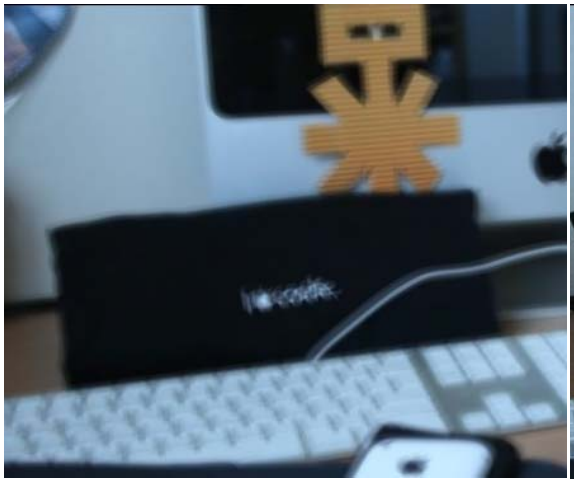
(h) deblurred result

Fig. 5.3 Case1: rotation/kernel1

(a) blurred image

(b) non-blurred image

(c) blurred/non-blurred stitching result



(d) blurred overlapped patch

(e) non-blurred overlapped patch

(f) true kernel

(g) estimated kernel

(h) deblurred result

Fig. 5.4 Case1: scaling/kernel2

(a) blurred image

(b) non-blurred image

(c) blurred/non-blurred stitching result

(d) blurred overlapped patch

(e) non-blurred overlapped patch

(f) true kernel

(g) estimated kernel

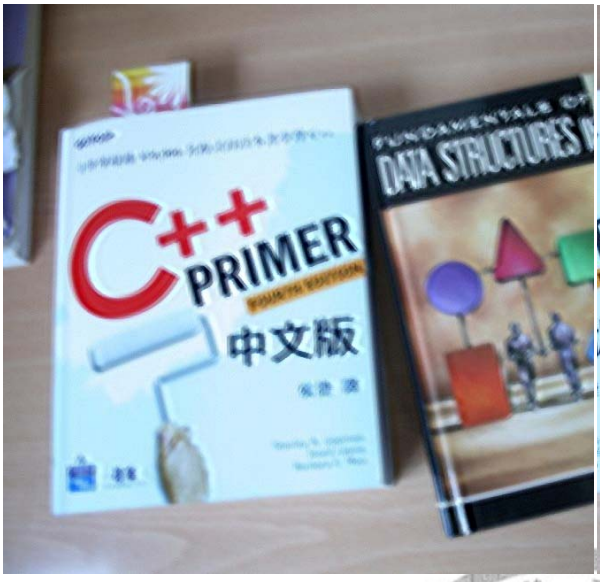
(h) deblurred result

Fig. 5.5 Case1: perspective/kernel2

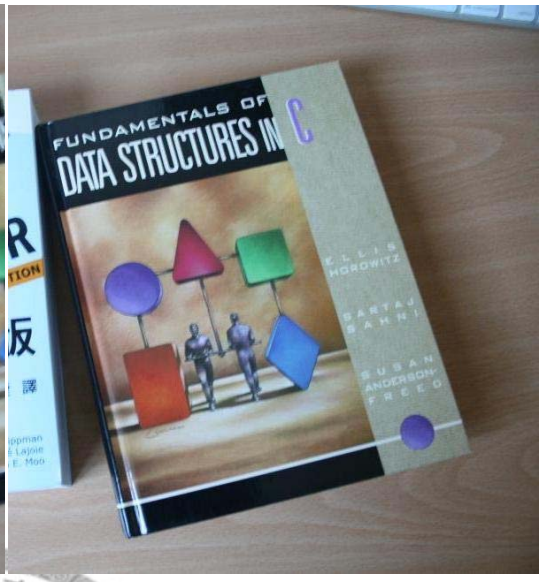
5.2 Deblur with Stitching Pre-deblurred/Non-blurred images

As we know, the deblurred results are not satisfying due to the poor estimated kernel. The main reason of incorrect kernel estimation is caused by a fundamental problem; we directly stitch blurred/non-blurred images together. To solve this issue, we decide to roughly deblur the blurry image before stitching. This roughly deblurring step is based on Fergus' work [4], which takes only one single image as input to estimate the blur kernel.

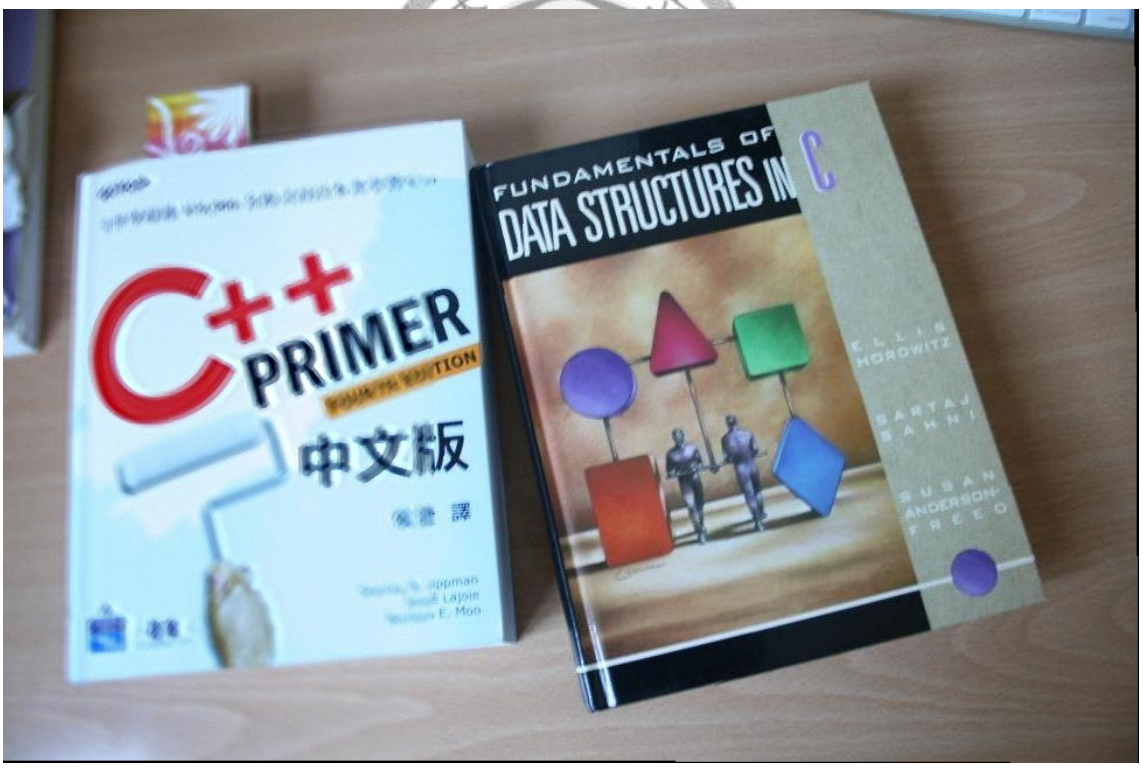
(a) Fergus' kernel



(b) pre-blurred image

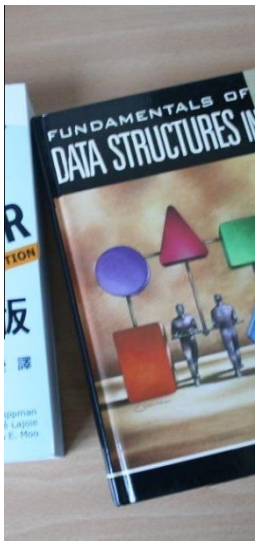


(c) non-blurred image



(d) pre-blurred/non-blurred stitching result

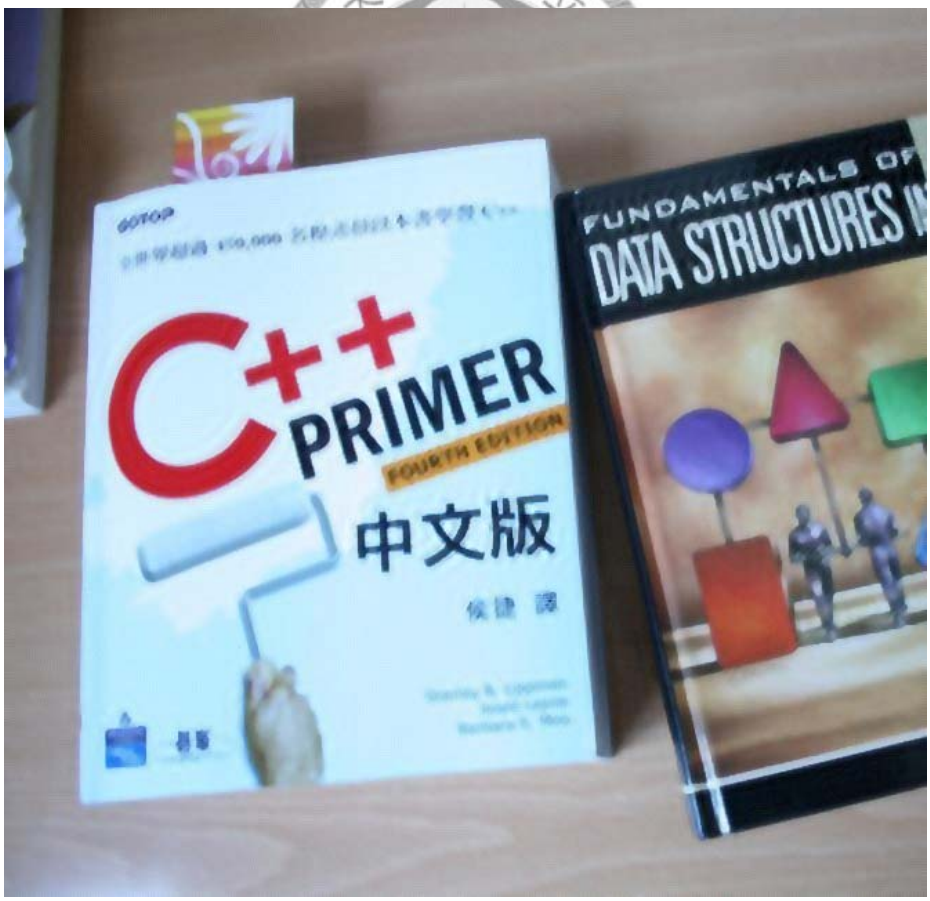
(e) blurred overlapped patch



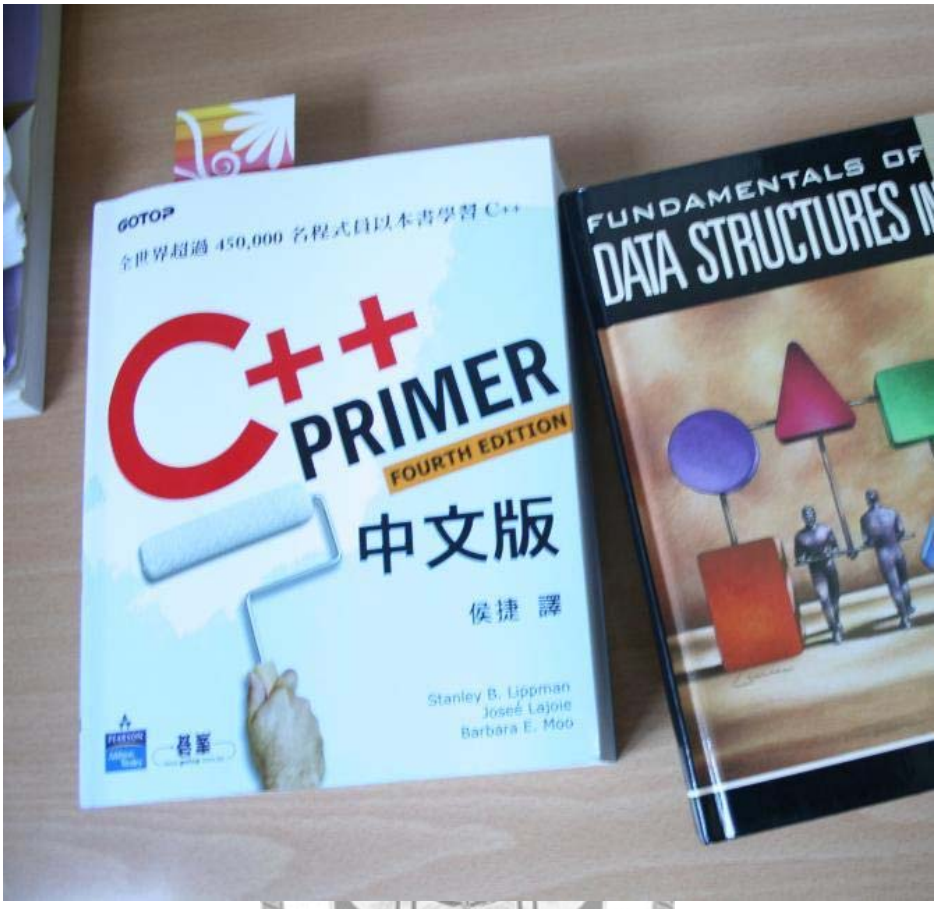
(f) non-blurred overlapped patch

(g) true kernel

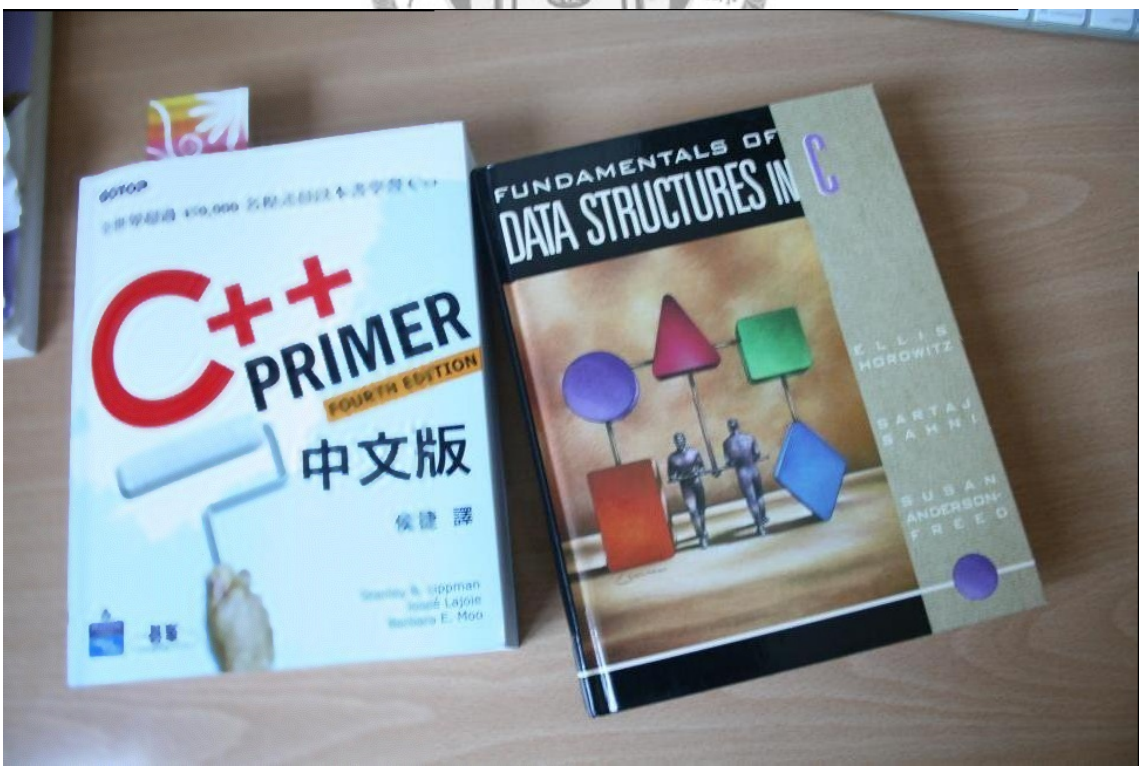
(h) estimated kernel



(i) deblurred result



(j) original image



(k) re-stitching result

Fig. 5.6 Case2: translation/kernel1

(a) Fergus' kernel

(b) pre-blurred image

(c) non-blurred image

(d) pre-blurred/non-blurred stitching result

(e) blurred overlapped patch

(f) non-blurred overlapped patch

(g) true kernel

(h) estimated kernel

(i) deblurred result

(j) original image

(k) re-stitching result

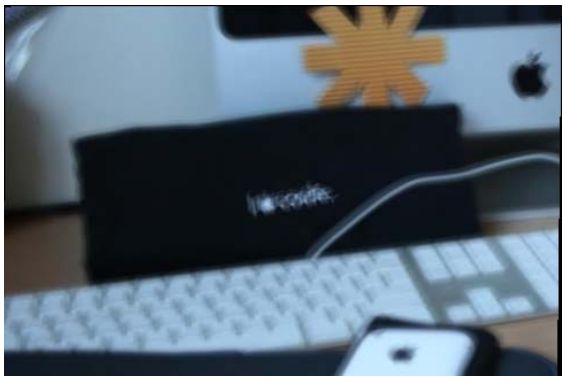
Fig. 5.7 Case2: rotation/kernel1

(a) Fergus' kernel

(b) pre-blurred image

(c) non-blurred image

(d) pre-blurred/non-blurred stitching result



(e) blurred overlapped patch

(f) non-blurred overlapped patch

(g) true kernel

(h) estimated kernel

(i) deblurred result

(j) original image

(k) re-stitching result

Fig. 5.8 Case2: scaling/kernel2

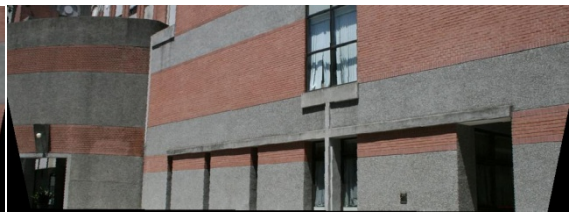
(a) Fergus' kernel

(b) pre-blurred image

(c) non-blurred image

(d) pre-blurred/non-blurred stitching result

(e) blurred overlapped patch



(f) non-blurred overlapped patch

(g) true kernel

(h) estimated kernel

(i) deblurred result

(j) original image

(k) re-stitching result

Fig. 5.9 Case2: perspective/kernel2

Chapter 6 Conclusion and Discussion

6.1 Conclusion

In this thesis we try to stitch two images under the occurrence of camera shake at the same time to deblur the blurry image. However, our experiments show that directly stitching of blurred/non-blurred images yields unsatisfying stitching result so as the kernel estimation. With no surprise, the final deblurring result fails. We found that the major problem causing this failure of deblurring is our misaligned patches. Due to these bad output patches, deblurring using an incorrect kernel results in an incorrect deblurred image.

To solve this problem, we decide to pre-deblur the blurred image before stitching, so we adopt a single image kernel estimation method as a pre-processing step. In this pre-processing step, we deblur the image using the progressive deconvolution approach with the kernel estimated from Fergus' approach. After the image is pre-deblurred, stitching of pre-deblurred/non-blurred images leads to well aligned patches. Since overlapped patches are well aligned, the transformation parameters between two images can be accurately calculated. Now we can stitch blurred/non-blurred images using the estimated parameters to get another pair of better-aligned patches, which are used in kernel estimation step. At last, satisfying deblurring result is achieved and re-stitching

of deblurred/non-blurred is performed.

6.2 Discussion

Some ringings more or less occurred in the deblurred image, for example, the up right corner in Fig. 5.9 (i). We impute this artifact to that our kernel is not perfect as true kernel. Another observation is that in some cases the naïve approach (case1) does work well. Nevertheless, we decided to take pre-deblur approach (case2) as our system flow for the concern of system robustness. An example of such case both using naïve approach (case1) and pre-deblur approach (case2) are shown in Fig. 6.1 and Fig. 6.2.

In our system we have a limitation in generating a panorama. We should notice that the deblurring model we discussed so far is assumed shift-invariant. This shift-invariant property means that the observed blurred image is a product of convolution with a latent image and a blur kernel. In other words, the image is blurred by the same kernel for all image pixels. However in perspective case, if we transform the blurred image to stitch with the non-blurred image, the blurred image will no longer fit the shift-invariant blurring model even though translation, rotation or scaling transformation holds this property. Therefore, in our system we can only transform the non-blurred image to stitch with the blurred image which limits our application. In the future, we will need a deblurring method derived from the assumption of shift-variant property. Also, blur

kernel estimation should be modified.

(a) blurred image

(b) non-blurred image

(c) blurred/non-blurred stitching result

(d) blurred overlapped patch

(e) non-blurred overlapped patch

(f) true kernel

(g) estimated kernel

(h) deblurred result

Fig. 6.1 A successful example of the naïve approach (case1).

(a) Fergus' kernel

(b) pre-blurred image

(c) non-blurred image

(d) pre-blurred/non-blurred stitching result

(e) blurred overlapped patch

(f) non-blurred overlapped patch

(g) true kernel

(h) estimated kernel

(i) deblurred result

(j) original image

(k) re-stitching result

Fig. 6.2 Taking pre-deblur approach as our system flow is more robust.

REFERENCE

[1] H. Bay, A. Ess, T. Tuytelaars, L. V. Gool, “SURF: Speeded Up Robust Features,” *CVIU 110*, 3, 346-359, 2008.

[2] N. Dey, L. Blanc-Fraud, C. Zimmer, Z. Kam, P. Roux, J. Olivo-Marin, and J. Zerubia, “Richardson-lucy algorithm with total variation regularization for 3d confocal microscope deconvolution,” *Microscopy Research Technique 26*, 69, 260-266, 2006.

[3] F. Durand, and J. Dorsey, “Fast bilateral filtering for the display of high-dynamic-range images,” *ACM Trans. Graph* (*SIGGRAPH)*, 257-266, 2002.

[4] R. Fergus, B.Singh, A.Hertzmann, S. T. Roweis, and W. T. Freeman, “Removing camera shake from a single photograph,” *ACM Trans. Graph* (*SIGGRAPH)25(3)*:787-794, 2006.

[5] C. Harris, and M. Stephens, “A combined corner and edge detector,” In *Proceedings of Fourth Alvey Vision Conference*, 147-157, 1988.

[6] J. Kopf, M. Cohen, D. Lischinski, and M. Uyttendaele, “Joint bilateral upsampling,” *ACM Trans. Graph* (*SIGGRAPH) 26*, 3, 96-99, 2007.

[7] D. Lowe, “Distinctive image features from scale-invariant keypoints,” *IJCV*,

60(2):91-110, January 2004.

[8] L. B. Lucy, “An iterative technique for the rectification of observed distributions,” *Astronomical Journal*, Vol. 79, 745, 1974.

[9] J. W. Miskin, “Ensemble learning for independent component analysis,” *Cambridge Thesis*, 2000.

[10] A. Rav-Acha, and S. Peleg, “Two motion-blurred images are better than one,” *Pattern Recognition Letter*, 26, 3 ,311-317, 2005.

[11] W. H. Richardson, “Bayesian-based iterative method of image restoration,” *JOSA*, A 62, 1, 55-59, 1972.

[12] A. Tikhonov, “On the stability of inverse problems,” *Dokl. Akad. Nauk SSSR 39*, 5, 195-198, 1943.

[13] C. Tomasi, and R. Manduchi, “Bilateral filtering for gray and color images,” *ICCV*, 839-847, 1998.

[14] L. Yuan, J. Sun, L. Quan, and H.-Y. Shum, “Image deblurring with blurred/noisy image pairs,” *ACM Trans. Graph* (*SIGGRAPH) 26*, 3 ,1-10, 2007.

[15] L. Yuan, J. Sun, L. Quan, and H.-Y. Shum, “Progressive inter-scale and intra-scale non-blind image deconvolution,” *ACM Trans. Graph* (*SIGGRAPH) 27*, 3, 2008.