Applications of Fourier Analysis in Image Recovery
TJHSST Senior Research Project
Computer Systems Lab 2009-2010

Kang Guo

May 25, 2010

Abstract

The goal of this project is to explore and implement image deblurring techniques. Many of these techniques involve some sort of an image transform, and the most commonly used one is the Fourier Transform. A Point Spread Function, which is the Fourier Transform of the blur kernel, can be applied to an image in the frequency domain after it has been transformed to create a blurry image. The opposite of this can be done by applying a transform to a blurry image, and after removing the point spread function from the frequency domain, a deblurred image can be obtained.

Keywords: Discrete Fourier Transform, Fast Fourier Transform, Point Spread Function, Blur Kernel, Blind Deconvolution, Non-Blind Deconvolution, Regularization

1 Introduction

Motion blur in images is a common problem for professionals in various fields. When the image is deblurred, the usefulness of the image increases. Parts of the image that were difficult to identify can be rendered to effective clarity. This project will explore and implement image deblurring techniques. By implementing these techniques, users can efficiently remove blur from an image.

2 Background

Regarding many image processing techniques, transforming the image from a spatial domain to a frequency domain is very helpful. In the area of image deblurring, a Fourier Transform is useful in creating the frequency domain image. The Discrete Fourier Transform only describes the frequencies contained in the spatial domain of the image, as opposed to a Continuous Fourier Transform which will describe a continuous range of frequencies.

The Discrete Fourier Transform is not sufficiently fast for an image of size 256x256 pixels. Instead, speed improvements can be obtained by implementing a Fast Fourier Transform. This transform relies on the fact that the Discrete Fourier Transform is separable. Originally, N, 2-dimensional transforms would have to be done. With the Fast Fourier Transform, 2N 1-dimensional transforms have to be done, resulting in an efficiency on the order of $N \log N$, compared to N^2 for a Discrete Fourier Transform. On a 256x256 pixel image, speed improves from upwards of ten minutes to just five seconds.

After applying the Fourier Transform, a blur kernel can be applied to the frequency domain image, and after applying an inverse transformation, a blurred image will be obtained. This process is called convolution and requires that the Fourier Transform of the blur kernel, known as the Point Spread Function be multiplied to the Fourier Transform of the original image. The same process can be done to remove blur from an image. This is known as deconvolution. Instead of applying the blur kernel to the frequency domain image, it can simply be removed, and then transformed back to spatial domain to produce the deblurred image. This process is simply the inverse, instead of multiplying the transformed images, they can be divided. Note that when working with images that have been Fourier Transformed, image values are stored as complex numbers. Therefore, the identity $e^{i\theta} = \cos\theta + i\sin\theta$ is useful.

Attempting to traditionally deblur an image will result in unwanted noise and ringing artifacts. However, a finite number of Fourier basis functions are able reconstruct the image without much data loss. In determining the blur kernel, iterating between updating the blur kernel and the estimated latent image will ultimately allow the two to converge and produce an acceptable deblurred image. The Richardson-Lucy algorithm is sufficient in blind deconvolution, in which the PSF is not known. In the absence of noise, this algorithm functions well, and by increasing the number of iterations, the

quality of the image increases. The formula is as follows, where h is the PSF, f is the original image, and g is the blurred image. H^* is the adjoint operator of H, where $Hf = g$.

$$f^{n+1} = f^n H^*(\frac{g}{(Hf^n)})$$

Deblurring can also be done with a pair of images: a blurry one and a noisy one. By removing noise from the noisy image, an estimate of what the final image should look like can be obtained. This helps in the estimation of the blur kernel. Again, iterations between estimating the blur kernel, residual deconvolution, and de-ringing will ultimately allow the image to converge and produce an image of acceptable quality.

When a Fourier Transform is applied on an image, displaying the magnitude of the Fourier values can be an issue, due to their scale. Some values may be too small or too large and can not be displayed properly as 8-bit pixel values. Applying a logarithmic transform can fix this problem. However, the existence of these small Fourier values becomes a problem again in the inverse process, in which the blurred image is divided by the PSF. Dividing by these small values amplifies noise, and in some cases, may cause unacceptable results, as seen in Fig. 7. I find this to occur when image are blurred with Gaussian blurs and blurs that are small in size. To remove noise that is created in this way, two types of filters can be applied. The inverse filter is a high pass filter that sets all Fourier values deemed to be too small to a common value. This way, division by small Fourier values will be avoided. However, any additive noise will easily corrupt the resulting image. A second approach, by using the Wiener filter, attempts to find a balance between removing noise while preserving the quality of the actual image. The Wiener filter will generally provide better results, but at the cost of higher complexity and loss of efficiency.

The process of regularization involves introducing additional information into a problem to allow a proper solution. This is particularly useful in machine learning and inverse problems. In the case of image deblurring, the inverse part of the process, where we divide the blurred image by the Point Spread Function produces a loss of data. Regularization will usually exist as some sort of restriction, forcing the final solution to fall in a set boundary of answers. The effects of regulariztion can be seen in Figures 8 and 10. In the cases of Tikhonov Regularization and Truncated Singular Value Decomposition, images and blur filters must be expressed as matrices. In most code, images are often stored in matrix arrays, with each cell in the

matrix representing the pixel value at a certain point in the image. However, in most code, operations made with the images do not follow common matrix operations such as matrix multiplication and division. Normally, multiplying a transformed image and a Point Spread Function would take place very similarly to a dot product, in which both images are treated as vectors, and one value is multiplied by one other value. However, in regularization, the Point Spread Function must be treated as a matrix, and original and blurred images must be treated as vectors. In the basic matrix equation $A\text{x}= b$, x will be the original image of length N^2, b will be the blurred image of length N^2, and A will be the Point Spread Function of size $N^2\text{x}N^2$ or N^4 individual values. This is acceptable for small matrices, but in image processing when images are at least 256x256 pixels, the Point Spread Function matrix scales up very quickly.

3 Development

Code for my project was done primarily written in the C programming language. I used imagemagick to convert various images into the pgm format, which I used to directly read color values. Testing revolved around applying forward and inverse Fourier Transforms to test the functionality of the image transforms. Ultimately, after applying blur kernels to the frequency domain image, I was able to visually verify the effect of the deblurring program.

In applying image deblurring techniques, I will be primarily focusing on eliminating ringing artifacts and noise from a transformed image. The creation of noise can be both intrinsic and additive, that is deconvolution process may itself create noise, or additive noise present in the original image may generate even more noise when deconvoluted.

The Fourier image is often displayed with F(0,0) in the center of the image. In the frequency domain with this particular shifting, the further away from the center of the image, the higher the frequency is. Visually, one can detect gradients in the Fourier transform that corresponds to the original image in the spatial domain. For example, this image (Fig. 1) has two dominating directions, one along the x-axis and another along the y-axis. This can be seen in the Fourier image (Fig. 2) shown by the strong lines intersecting at the middle, and in the original image shown by the border of the mirror.

Modeling certain blurs is simple. A simple motion blur can be described

as a white line centered in a square image, with all other pixels black. This image, essentially, the blur filter can then be transformed into the Point Spread Function, and then used in convolution and deconvolution processes. Gaussian blurs can be described as three-dimensional normal curves, much like a "bulge". 8-bit pixel values can be used to model the height of the "bulge" at certain points, and thus, a disk that is white in the center and moving outward, has decreasing pixel values, ultimately converging to black, can describe a Gaussian blur. These blur filters should be centered in the middle of the image and should be rotationally symmetric, so that the Point Spread Function will also be centered in the middle and be rotationally symmetric. Different blur filters will produce different types of blurred images, and using a Fourier Transform approach to blur an image, a large variety of blurs can be modelled.

Fig. 3 outlines the image blurring process, in which the Point Spread Function and the transformed image are multiplied. Different types of blur can be modeled with the Point Spread Function, as seen in Fig. 5 and Fig. 6. Linear blurs are similar to camera shake which results in a shift of the image. Gaussian blurs are similar to pictures that are taken out of focus which results in loss of pixel clarity. Fig. 4 outlines the image deblurring process, which is simply the inverse of the blurring process. The values are divided instead of multiplied, and the image is returned to its normal state. In this process, some very small Fourier values are divided, resulting in amplification of noise.

Matlab is a good tool for general work with mathematics, but it also has an Image Processing Toolbox that is very helpful for image deblurring. Many functions necessary in the code that I have already written are implemented in Matlab. Due to the complexity of the deconvolution techniques presented thus far and the scope of this project, I will be implementing filters and regularization in Matlab. Both of the Matlab functions *deconvreg()* (Fig. 9) and *deconvlucy()* (Fig. 10) implement a type of non-blind deconvolution. *deconvreg()* applies regularization to a normal non-blind deconvolution algorithm, while *deconvlucy()* estimates and update latent deblurred images iteratively. *deconvreg()* produces almost an ideal result, with the exception of small amounts of noise. Otherwise, the generated image is almost identical to the original. *deconvlucy()* produces an image that contains ringing artifacts as a result of the Fourier transforms involved. The generated image still appears to be blurred, but to a lesser degree. However, both of these deconvolution algorithms deal with non-blind deconvolution, and a blind de-

convolution problem would not only be much more difficult, better results would also be more difficult to obtain.

4 Conclusion

This paper demonstrates only the most basic of image deblurring techniques, using Fourier Transforms and other general processes. The focus was primarily on non-blind deconvolution, in which the user knows how the image was blurred in the first place. Reversing the blurring process to create a deblurring algorithm is relatively simple, and even still, it is not perfect. Perfecting the process of deconvolution and generating high-quality images from degraded ones is a difficult science, but it is one that, when completed, many people can benefit from.

Images serve various purposes in many different fields, and the clarity of an image is almost universally preferred. Thus, a program to effectively remove blur in images would be useful in any subject area. Such functionality would allow photographers and image editors to be able to remove blur and increase clarity of images. Casual image enhancement would allow photographers to take more presentable pictures with less blur. Deblurring photographs taken by roadside cameras would allow law enforcement to clearly read license plate numbers. Photographs of astronomical objects are often blurred and can be restored so that they can be more useful in studies. Images provide perfect vision to an observer and can often give one large amounts of knowledge from just one photograph. When the quality of the image is degraded, so much information is lost and professionals are working with many unknown variables. Recovering these images has many applications, and the sky is the limit for image deblurring.

References

[1] Q. Shan, J. Jia, and A. Agarwala. "High-quality Motion Deblurring from a Single Image" *ACM Trans. Graph. 27*, 3, Article 73 (August 2008).

[2] L. Yuan, J. Sun, L. Quan, and H. Shum. "Image Deblurring With Blurry/Noisy Image Pairs" *ACM Trans. Graph. 26*, 3, Article 1 (July 2007).

[3] R. Fisher, S. Perkins, A. Walker, and E. Wolfart. "Image Transforms - Fourier Transform", 2003. http://homepages.inf.ed.ac.uk/rbf/HIPR2/fourier.htm

[4] T. Chan and J. Shen. "Theory and Computation of Variational Image Deblurring" *WSPC/Lecture Notes Series* (November 2005).

[5] S. Allon, M. Debertrand, and B. Sleutjes. "Fast Deblurring Algorithms", 2004. http://www.bmi2.bmttue.nl/image-analysis/Education/OGO/0504-3.2bDeblur/OGO3.2b_2004_Deblur.pdf

[6] Y. Fan. "Deblurring Images: Python, CGI, and Web", 2006. www.mathcs.emory.edu/~yfan/SSProject.pdf

[7] G. Kempen and L. Vliet. (2000). "The influence of the regularization parameter and the first estimate on the performance of Tikhonov regularized non-linear image restoration algorithms", *Journal of Microscopy.* 198. pp. 63-75.

[8] Z. Zhao and R. Blahut. "The Richardson-Lucy Algorithm Based Demodulation Algorithms for the Two-dimensional Intersymbol Interference Channel", 2005. www.ifp.illinois.edu/~zzhao/publications/slides_wustl23Mar05.pdf

Figure 1: Original Image

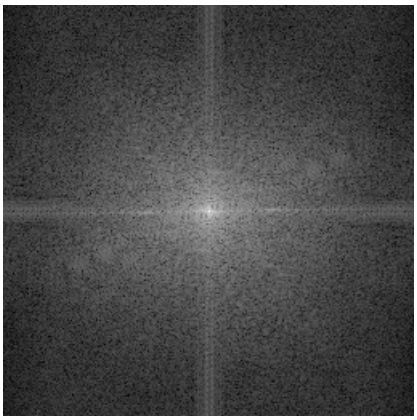

Figure 2: Fourier Transformed Image

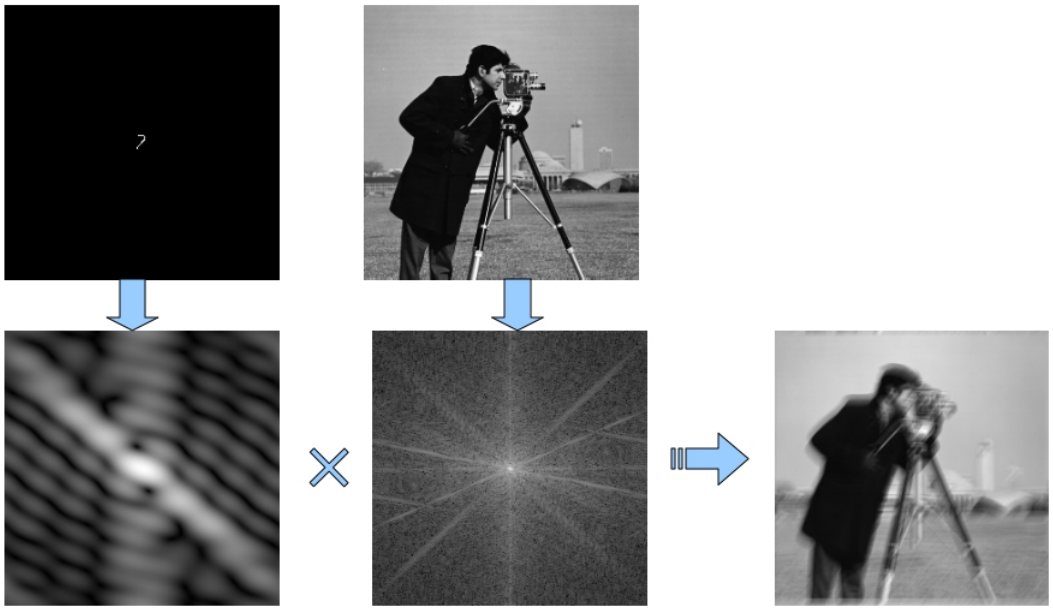

Figure 3: Blurring an Image

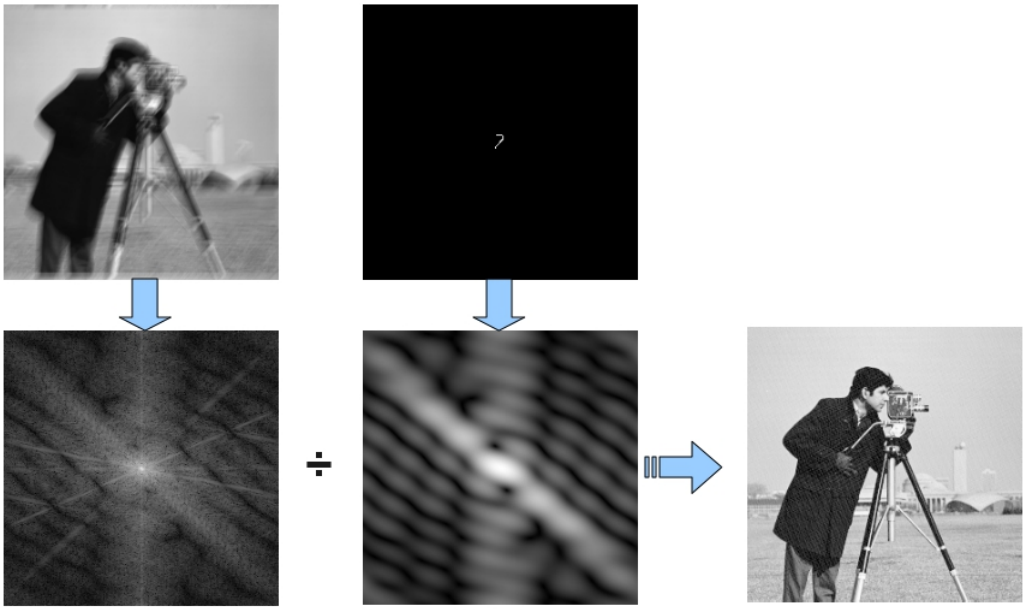

Figure 4: Deblurring an Image

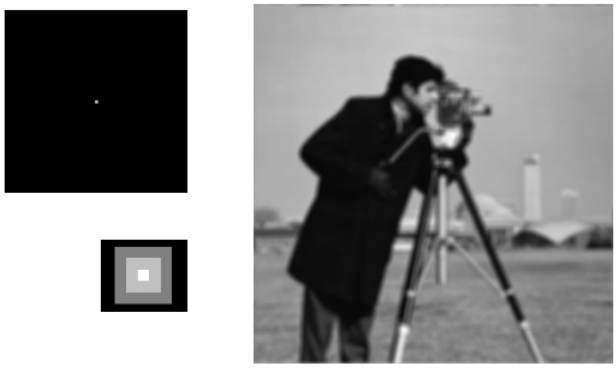

Figure 5: Gaussian Blur

Figure 6: Linear Blur

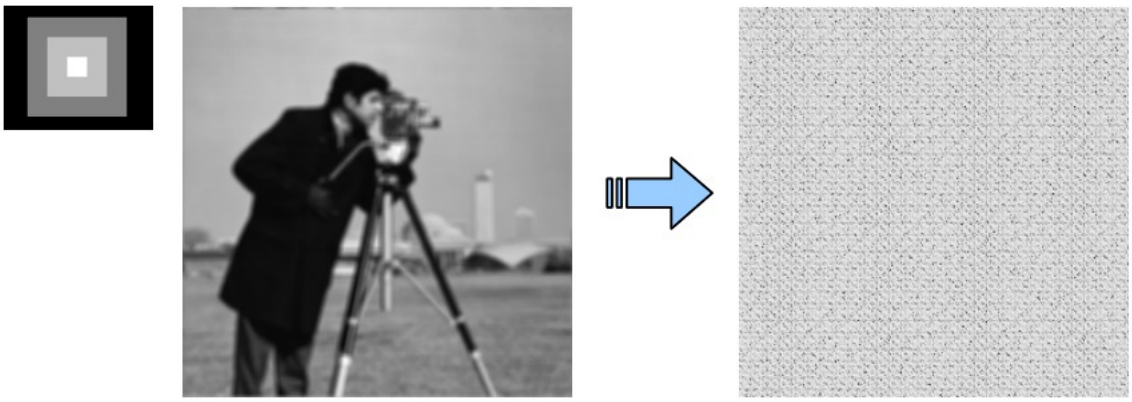

Figure 7: Noise

Figure 8: Without Regularization

Figure 9: deconvlucy(), motion blur

Figure 10: deconvreg(), motion blur