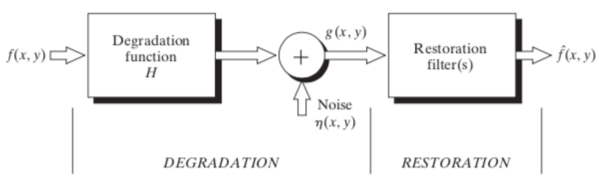
EE 610 : Assignment 2 - Image Restoration

Saqib Azim - 150070031, saqib_azim@iitb.ac.in
Indian Institute of Technology, Bombay

***Abstract*— This report deals with the problem of restoring an image which has been blurred by a known kernel function. Various image restoration techniques such as full-inverse filtering, truncated inverse filtering, Weiner filtering and Constrained Least Square Filtering have been implemented and their performances have been compared. The metrics used for performance evaluation of the above-mentioned approaches are Peak Signal-to-Noise Ratio (PSNR) and Structural Simlarity Index Metric (SSIM) between the undegraded and restored image.**

I. Introduction

The goal of image restoration technique is to improve the quality of an image either in perceived sense or better quality compared to original image. Image Restoration tries to recover the original image degraded by some degradation phenomenon. The restoration techniques are oriented towards modeling the degradation process and applying its inverse in order to recover the original or desired image. The degradation and restoration model used is shown in Fig.1



(a)

Figure 1: Model of image degradation and restoration process

The image restoration problem is a difficult task in itself because in most real life scenarios, the exact degradation function is unknown. For example, an image taken from a camera blurred due to motion of camera or background or object itself. In this situation, the degradation is due to some environmental phenomenon and hence the degradation function can only be approximated. Another example can be image transmission (when we upload an image on Facebook, it compresses it and hence some data is lost) in which some quality of the image can be lost. On top of all these, how can one forget the noise due to various factors (measuring device, environment, etc,) which can only be statistically modeled and hence exact reconstruction is nearly impossible in most scenarios.

Though in some cases, image degradation is not perceived to human eyes or are good enough for a particular task and in these cases, no or little amount of image restoration will do the job. But situations such as scientific computing, satellite communication, biological images require or demand high quality images or as close as possible to original image. This makes image restoration a very important problem to be taken care of.

As shown in Fig.1, the degradation process is modeled as a degradation function H acting on the original image $f(x, y)$ which is further destroyed by additive noise $\eta(x, y)$ to produce degraded image $g(x, y)$. The objective of the restoration task is to obtain an estimate of the original image $\hat{f}(x, y)$ as close as possible to the $f(x, y)$. H is generally assumed to be linear, positive-invariant function. The degraded image in the spatial domain is given by

$$g(x, y) = f(x, y) * h(x, y) + \eta(x, y) \tag{1}$$

In frequency domain the above equation is given by

$$G(u, v) = F(u, v)H(u, v) + N(u, v) \tag{2}$$

where the terms represent the fourier transform of their corresponding terms in spatial domain.

II. Background and related work

A. *Full Inverse Filtering*

When the kernel is known, either by observation or by some estimation process, the most crude way to estimate the original signal or restore the blurred image is to ignore the noise and using equation. 2, simply calculate the $\hat{F}(u, v)$ by dividing the fourier transform of degraded image by fourier transform of the kernel function and then take the inverse fourier transform of the resultant to estimate $\hat{f}(x, y)$.

$$\hat{F}(u, v) = \frac{G(u, v)}{H(u, v)} \tag{3}$$

$$\hat{f}(x, y) = \mathcal{F}^{-1}(\hat{F}(u, v)) \tag{4}$$

B. *Truncated Inverse Filtering*

The result of the full-inverse is bad because firstly, ignoring noise completely and secondly dividing by $H(u, v)$, which can have zero or near zero elements.

$$\hat{F}(u, v) = F(u, v) + \frac{N(u, v)}{H(u, v)} \tag{5}$$

As shown in eq. 5, simple division can lead to blowing up of the value of the fraction term $\frac{N(u,v)}{H(u,v)}$ which means that the noise term will dominate the original image for those values where $H(u, v)$ is nearly or exactly zero.
A simple hack to mitigate the above mentioned problem is to zero-out the values of the fraction $\frac{G(u,v)}{H(u,v)}$ after a certain

threshold. But using simple thresholding introduces ripple effects,and hence to overcome the ripples, one can use low-ass filter like Butterworth filter with a cut-off frequency D_0. The transfer function of butterworth filter is defined below.

$$H(u,v) = \frac{1}{\sqrt{1 + \frac{D(u,v)}{D_0}^{2n}}} \tag{6}$$

where $D(u,v) = \sqrt{(u - M/2)^2 + (v - N/2)^2}$ and M, N are the rows and columns of the fourier transform of image.

C. Wiener or Minimum Mean Square Error filtering

As the name suggests, this method minimizes the mean square error between the original image $f(x,y)$ and the restored image $\hat{f}(x,y)$ given by

$$e^2 = E[(f - \hat{f})^2] \tag{7}$$

The solution to the above equation in frequency domain is given by,

$$\hat{F}(u,v) = [\frac{H^*(u,v)S_f(u,v)}{S_f(u,v)|H(u,v)|^2 + S_\eta(u,v)}]G(u,v) \tag{8}$$

where $H(u,v)$ is the degradation function, $|H(u,v)|^2 = H(u,v)H^*(u,v)$, $S_\eta(u,v)$ is the power spectrum of the noise, $S_f(u,v)$ is the power spectrum of the undegraded image. In most scenarios, $S_\eta(u,v)$ is known or can be guessed or estimated but power spectrum of the undegraded image is rarely known and hence to make wiener filtering usable in practical scenarios, the fraction term $\frac{S_\eta(u,v)}{S_f(u,v)}$ is taken to be a constant K. The modified equation is

$$\hat{F}(u,v) = [\frac{1}{H(u,v)} \frac{|H(u,v)|^2}{|H(u,v)|^2 + K}]G(u,v) \tag{9}$$

D. Constrained Least Square Filtering

: In wiener filtering, the power spectrum of undegraded image is not always known and the so called "jugaad" solution of putting the fraction as a constant is sometimes not suitable. Hence this method requires only the mean and variance The frequency domain solution of this filtering is given by,

$$\hat{F}(u,v) = [\frac{|H(u,v)|^2}{|H(u,v)|^2 + \gamma|P(u,v)|^2}]G(u,v) \tag{10}$$

where $P(u,v)$ is the fourier transform of $p(x,y)$

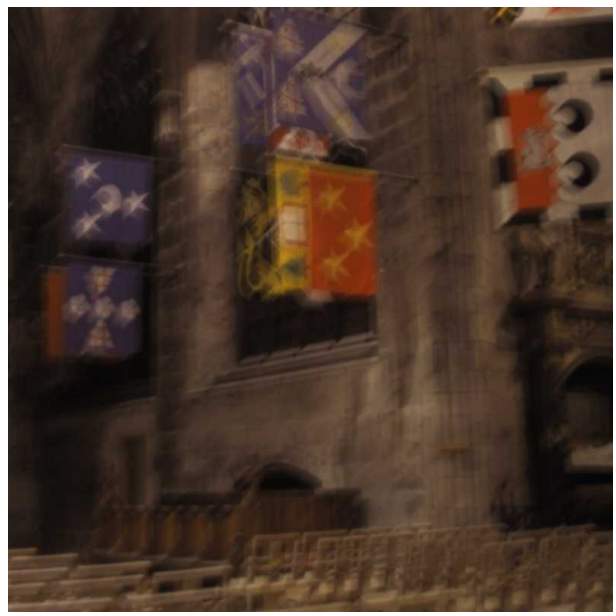
$$p(x,y) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{11}$$

III. Experiments and Results

Since the experiments were conducted assuming spatially invariant kernel which is different from the spatially variant kernel which was used to degrade the original image in Fig. 2. the resulting deblurred images are decent when it comes to perception while some results are extremely bad. The blurred image in Fig. 2 is generated using the original image in Fig. 1 and kernel function in Fig. 4 using the eq. 1.

(a)

Figure 2: Original Undegraded Image



(a)

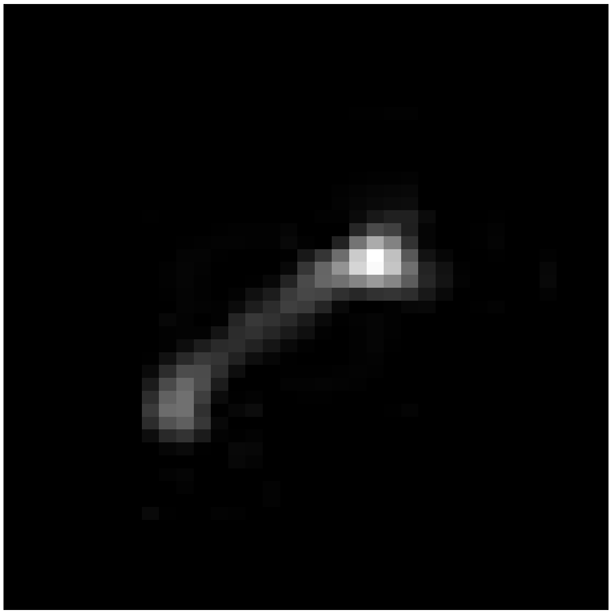
Figure 3: Image Blurred using a spatially variant kernel

A. Full Inverse Filtering Results

The result of deblurring using full inverse filtering is shown in fig. 5. The Peak Signal-to-Noise Ratio between the original image in Fig. 2 and filtered image in Fig. 5 is 12.74. The structural similarity index between the same pair of images is 0.3148.

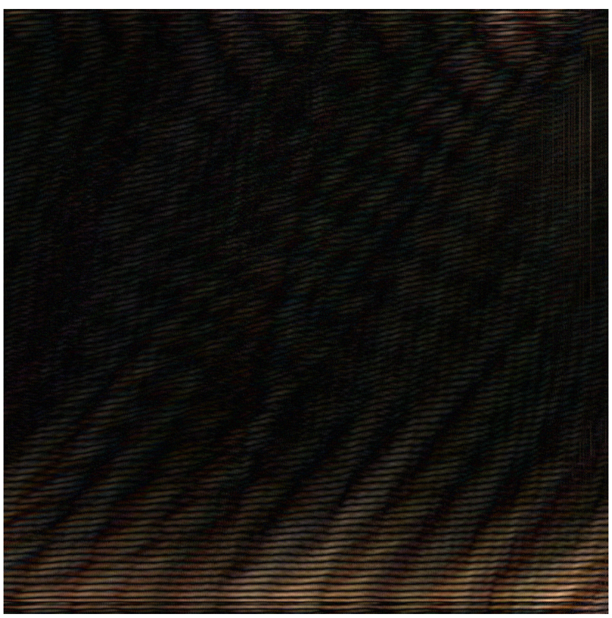
B. Truncated Inverse Filtering Results

The results of using a low-pass butterworth filter is pretty excellent compared to full inverse filtering and at the same times it also shows the full inverse filtering actually contains



(a)

Figure 4: Spatially Invariant Kernel used for deblurring Fig. 3 which has been cropped out from a spatially variant kernel function



(a)

Figure 5: Result of deblurring of Fig. 3 using Full Inverse Filtering. PSNR = 12.74, SSIM = 0.3148

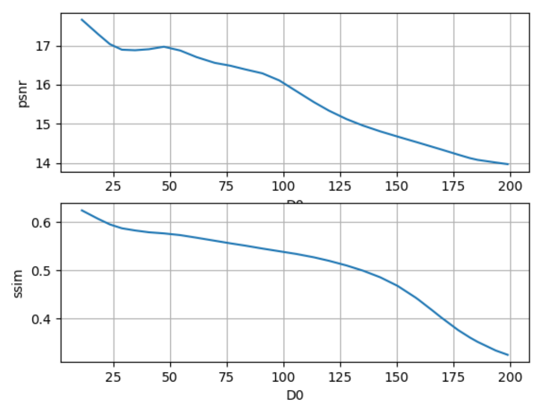
information about the undegraded image which only had to be removed using some technique like low-pass filtering. The variation of psnr and ssim with the cut-off radius(D_0) is given in Fig. 7.

C. Minimum Mean Square Error or Wiener Filtering Results

The wiener deblurring, shown in Fig. 8, produces a significantly comparable output to the best case of truncated inverse deblurring. The variation in the perceived quality of deblurred image in this case does not change much with the

(a)

Figure 6: Result of deblurring of Fig. 3 using Truncated Inverse Filtering with Butterworth Filter cut-off frequency(D0) = 105, order = 5



(a)

Figure 7: Variation of PSNR and SSIM with the cut-off radius of butterworth filter of order = 5

value of constant K. The variation of psnr and ssim with the constant K is shown in Fig. 9.

D. Constrained Least Square Filtering

The change in the deblurred images with γ is not much perceptible to human eyes in this case. Its result, shown in Fig. 10, is also comparable to that of the Truncated inverse and Weiner filter results. The variation of psnr and ssim with the constant γ is shown in Fig. 11.

E. NOTE

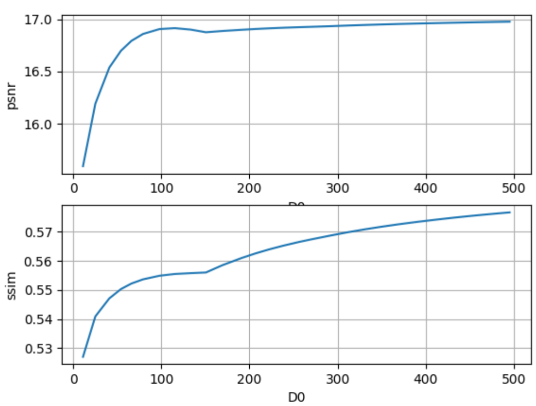
The results of Truncated Inverse, Weiner and Constrained LS Filters have been shown for their best case performance in

(a)

Figure 8: Result of deblurring of Fig. 3 using Wiener Filter with the constant K = 40

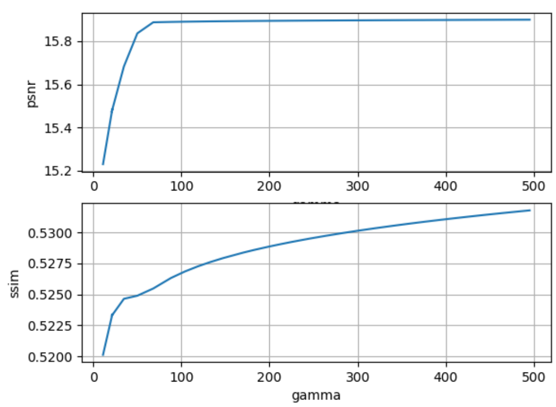
(a)

Figure 10: Result of deblurring of Fig. 3 using Constrained Least Square Filtering with constant gamma=135



(a)

Figure 9: Variation of PSNR and SSIM with the constant K in the wiener formula in eq. 9



(a)

Figure 11: Variation of PSNR and SSIM with the constant γ in the constrained LS solution in eq. 10

terms of my own perception and not based on any statistical measurements, PSNR, MSE or SSIM.

IV. Discrete Fourier Transform Implementation using Vectorized Approach

The fourier transform of an MxN image is given by

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \exp(-j2\pi(\frac{ux}{M} + \frac{vy}{N})) \tag{12}$$

which can be written as

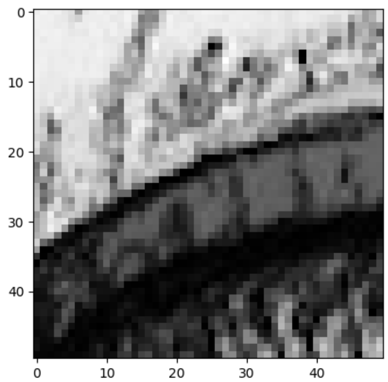
$$F(u,v) = \sum_{x=0}^{M-1} \exp(-j2\pi\frac{ux}{M}) \sum_{y=0}^{N-1} f(x,y) \exp(-j2\pi\frac{vy}{N}) \tag{13}$$

Without the vectorized approach, and implementing DFT using 4 ”for” loops takes around 2 hrs on an 800X800 image because the algorithm is $O(M^2N^2)$. The vectorized implementation does the same task in less than 1 second and algorithm follows $O(M^2 + N^2)$.

The results of slow DFT approach compared to fast vectorized DFT approach is shown in Fig. 13 which is nearly the same. The time difference is huge.For a 800x800 image, the slow approach takes nearly 2 hrs while the fast approach takes less than 1 second.

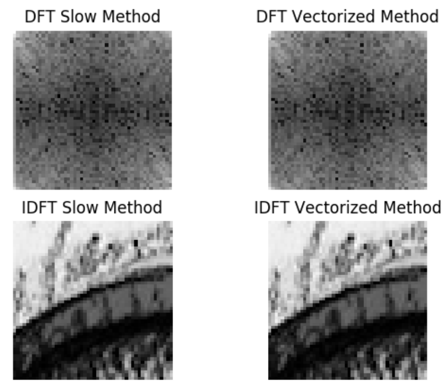
V. Links

Github Link of the Assignment
Drive folder of the Repository



(a)

Figure 12: Testing 50X50 image



(a)

Figure 13: Comparison of the slow and fast dft and idft approaches implemented on a 50x50 image. The slow DFT as well as IDFT on a 50x50 images takes around 30 s while the vectorized version of the same takes less than 10 ms

References

[1] R. C. Gonzalez, R. E. Woods, Digital Image Processing, 3rd ed. Prentice Hall

[2] Recording and playback of camera shake: benchmarking blind deconvolution with a real-world database, http://webdav.is.mpg.de/pixel/benchmark4camerashake/

[3] Matlab PSNR

[4] Matlab SSIM

[5] Stack Overflow

[6] Cooley-Tuckey FFT Algorithm

[7] https://jakevdp.github.io/blog/2013/08/28/understanding-the-fft/

[8] FFT Python Github Repository

[9] Insert images in Latex