Image Restoration

Assignment-2

Sarthak Nijhawan
150070037, EE Dept,
IIT Bombay

***Abstract*—This paper primarily focuses on the performance of various image restoration techniques such as inverse filtering, wiener filtering etc, prevalent in the domain of Image Processing along with their workings in detail. In this assignment, the main aim is to restore the underlying ground truth image, degraded as a part of some process during its capturing, as efficiently as possible using some of the standard techniques employed on a regular basis. The quality of restoration is evaluated using metrics such as PSNR and SSIM, which in turn try to emulate and model the human visual perception system mathematically.**

***Index Terms*—Image Restoration, PSNR, SSIM, Fast Fourier Transform**

.

I. Introduction

The purpose of image restoration is to "compensate for" or "undo" defects which degrade an image. Degradation comes in many forms such as motion blur, noise, and camera misfocus. In cases like motion blur, it is possible to come up with an very good estimate of the actual blurring function and "undo" the blur to restore the original image. While there also exist some cases where mathematical modeling of degradation is not possible. In this paper, we will introduce and implement several of the methods used in the image processing world to restore images and try to compare their performances on a diversed set of images degraded under different surroundings and conditions.

The block diagram for our general degradation model is given by

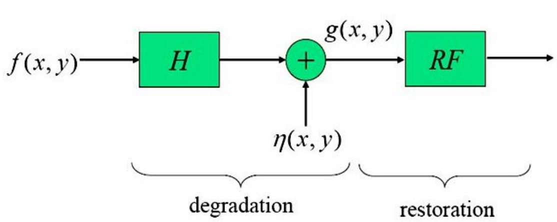

Fig. 1: Degradation model

where **g** is the corrupted image obtained by passing the original image **f** through a low pass filter (blurring/degrading function) **h** and adding noise to it. The corresponding equations in spatial and frequency domain are as follows:

$$\begin{aligned} g(x,y) &= f(x,y) * h(x,y) + \eta(x,y) \\ G(u,v) &= F(u,v)H(u,v) + N(u,v) \end{aligned} \tag{1}$$

With aforementioned model assumed to be true, we employ the following four techniques for image restoration, with the assumption that we're provided with the degraded image and its corresponding estimated degradation kernel.

1) Inverse Filtering
2) Truncated Inverse Filtering
3) Wiener Filtering
4) Constrained Least Square Filtering

The importance of this problem can be owed to various commonly occurring phenomena such as incorporation of unwanted noise while capturing, motion blurring due to high speed moving objects, mis-focusing of camera lenses, etc. A number of real-world problems from astronomy to consumer imaging find applications for image restoration algorithms. Plus, image restoration is an easily visualized example of a larger class of inverse problems that arise in all kinds of scientific, medical, industrial and theoretical problems.

The difficulty of the restoration problem premises on various factors :

- The degradation model doesn't generalize all the scenarios, owing to its assumptions which may fail in certain situations
- Even if the model holds, estimating the degradation function, accurately, is a huge problem in itself
- Moreover, not always is the case that ground truth will be available to evaluate the fidelity of the reconstruction

In this paper, as we'll see in the following sections, there are various situations where a particular algorithm can either excel in reconstructing the image or even fail miserably. I've tried to incorporate various such conditions in the paper along with the reasoning of such behaviours shown by the filters. Apart from testing each filter, I've also used a diversified set of sample images, where each and every image is degraded in a completely different environment, to compare the performances of all the filters based on the PSNR and SSIM values achieved post restoration.

The paper is divided into 6 sections. Following up, Section 2 briefs out some background theories and working of each restoration method, separately. Following up in Section 3 consists of some of the experiments I performed and the results that were achieved leading to some significant inferences. Section 4 concludes the discussion on the *Image Restoration*. Section 5 and Section 6 provides the reader with all the relevant links pertaining to the assignment that includes some resources/references and the code, pertaining to the whole discussion.

II. Background Read

This section explains each restoration technique, mathematically and its implementation in detail along with some background on the metrics used for evaluation of performance of a filter, which will play a significant role in making inferences in the next section.

A. Inverse Filtering

The quickest and easiest way to restore an image is by inverse filtering. This method generally assumes a high Signal-to-Noise ratio which lead to the estimation procedure, as shown below :

$$\begin{aligned}\hat{F}(u,v) &= G(u,v)/H(u,v)\\ &= F(u,v) + N(u,v)/H(u,v)\end{aligned} \tag{2}$$

where, $\hat{F}(u,v)$ is the fourier transform of the estimated image which is calculated by dividing the degraded image fourier transform, $G(u,v)$, by the given (or estimated) degradation function, $H(u,v)$.

Unfortunately, since the inverse filter is a form of high pass filer (since, H(u,v) is generally a low pass degradation), inverse filtering responds very badly to any noise that is present in the image because noise tends to be high frequency. This accounts for a major drawback of the filtering process and hence, many a times restored image looks full of noise with no trace of underlying structure/image.

B. Truncated Inverse Filtering

This method is an improvement over the naive inverse filtering and modifies the filtering process by suppressing all the high frequency components of the estimated image, as shown below:

$$\begin{aligned}\hat{F}_{inv}(u,v) &= G(u,v)/H(u,v)\\ &= F(u,v) + N(u,v)/H(u,v)\end{aligned} \tag{3}$$

where, $\hat{F}_{inv}(u,v)$ is the transform of the image obtained by applying inverse filtering. This method, then further applies a low pass filter $B(u,v)$ that approximately, cancels out the noise term from the equation as shown below

$$\begin{aligned}\hat{F}_{trunc}(u,v) &= B(u,v)\hat{F}_{inv}(u,v)\\ &= B(u,v)F(u,v) + B(u,v)N(u,v)/H(u,v)\\ &\approx B(u,v)F(u,v)\end{aligned} \tag{4}$$

One of the major drawback of this method is that it also rejects the high frequencies which are responsible for edges, regardless which leads to unnecessary smoothness along with ringing effects in the restored image due to Gibb's phenomenon.

C. Wiener Filtering

Wiener filtering is an improvement over both the aforementioned filters in a sense that it optimally tries to balance between both inverse-filtering and noise-smoothing, which is more like a trade-off and a mix of both techniques, where each comes into picture whenever necessary. It removes the additive noise and inverts the blurring simultaneously.

The Wiener filtering is optimal in terms of the mean square error. In other words, it minimizes the overall mean square error in the process of inverse filtering and noise smoothing. The Wiener filtering is a linear estimation of the original image.

$$MSE = E[(f(x,y) - \hat{f}(x,y))^2] \tag{5}$$

where, MSE is the expectation of the error difference between the original and reconstructed image, which is minimized to get the following equation, in frequency domain:

$$\hat{F}(u,v) = B(u,v)G(u,v) \tag{6}$$

where,

$$B(u,v) = \frac{H^*(u,v)}{|H(u,v)|^2 + S_f(u,v)/S_\eta(u,v)} \tag{7}$$

It is easy to see that the Wiener filter has two separate parts, an inverse filtering part and a noise smoothing part. It not only performs the deconvolution by inverse filtering (highpass filtering) but also removes the noise with a compression operation (lowpass filtering).

This filter too suffers from a drawback, which is nothing but the estimation issue of $S_f(x,y)$ and $S_\eta(x,y)$, where the latter can still be estimated based on certain assumptions whereas estimating/approximately the former is a huge problem in itself. Hence, the following approximation of Wiener filtering is deployed, generally, where the ratio is replaced by a constant, K, as shown below

$$B(u,v) = \frac{H^*(u,v)}{|H(u,v)|^2 + K} \tag{8}$$

D. Constrained Least Squares Filtering

The name of the filtering techniques owes from the regularized cost function it optimizes. Regularization adds another term to the minimization criterion to force the image to be somewhat smooth, as shown below:

$$J(\alpha) = \sum_{x,y} |g(x,y) - h(x,y) * f(x,y)|^2 + \alpha[l(x,y) * f(x,y)]^2 \tag{9}$$

where, the first term in the summation corresponds to the estimation error term while the second term is a filtered version of the estimated image that we expect to be small. In image processing, we usually expect a high level of local correlation or smoothness in the image. So the filter is chosen as a highpass filter so that we minimize the high-frequency content or "roughness" in the solution. The parameter α controls the degree of the roughness penalty. The larger it is, the smoother the restored image will be. The same equation in frequency domain can be equivalently represented as follows:

$$J(\alpha) = \sum_{x,y} |G(u,v) - H(u,v)F(u,v)|^2 + \alpha|P(u,v)F(u,v)|^2 \tag{10}$$

After optimizing the cost function, the fourier transform of the estimated image is calculated as

$$\hat{F}(u,v) = B(u,v)G(u,v) \tag{11}$$

where,

$$B(u,v) = \frac{H^*(u,v)}{|H(u,v)|^2 + \alpha * |P(u,v)|^2} \tag{12}$$

where, $P(u,v)$ represents the laplacian operator/filter which fulfills the highpass requirement in the cost function to ensure a certain amount of smoothness in the image.

A smaller alpha results in a noisier but sharper image while larger alpha results in a cleaner but blurrier image. While the choice of the regularization filter does have an influence on the results, even a non-optimal choice can yield good results. This method enjoys an extra advantage over the others as it can prove out be a useful tool when statistical information of any kind is unavailable about the original image or the imaging setup is unavailable.

E. PSNR

It is the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. PSNR is most easily defined via the mean squared error (MSE). Given a noise-free MxN monochrome image I and its noisy approximation K, MSE is defined as:

$$MSE = \frac{1}{m\,n}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2$$

The PSNR (in dB) is defined as:

$$\begin{aligned} PSNR &= 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right) \\ &= 20 \cdot \log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right) \\ &= 20 \cdot \log_{10}(MAX_I) - 10 \cdot \log_{10}(MSE) \end{aligned}$$

For color images with three RGB values per pixel, the definition of PSNR is the same except the MSE is the sum over all squared value differences divided by image size and by three. Also note that, PSNR attempts to approximate the human perception of reconstruction quality.

F. SSIM

SSIM is used for measuring the similarity between two images. The measurement or prediction of image quality is based on an initial distortion-free image as reference. SSIM is designed to improve on traditional methods such as peak signal-to-noise ratio (PSNR) and mean squared error (MSE).

The difference with respect to other techniques mentioned previously such as MSE or PSNR is that these approaches estimate absolute errors; on the other hand, SSIM is a perception-based model that considers image degradation as perceived change in structural information, while also incorporating important perceptual phenomena, including both luminance masking and contrast masking terms. Structural information is the idea that the pixels have strong inter-dependencies especially when they are spatially close. These dependencies carry important information about the structure of the objects in the visual scene.

The SSIM formula is based on three comparison measurements between the samples of x and y: luminance (l), contrast (c) and structure (s). The individual comparison functions are

$$l(x,y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2}$$

$$s(x,y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3}$$

SSIM is then a weighted combination of those comparative measures

$$SSIM(x,y) = \left[l(x,y)^\alpha \cdot c(x,y)^\beta \cdot s(x,y)^\gamma\right]$$

Setting the weights α, β, γ to 1, the formula can be reduced to the form :

$$\text{SSIM}(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

G. FFT Radix-2 DIT Algorithm

Naive 2D DFT algorithm takes a lot of time to run. It has a worst case running time complexity of $\mathcal{O}(M^2N + MN^2)$. This takes a huge amount of time to compute in the forward direction. For instance, in our case, it took approximately 36 seconds to compute the DFT of a 256x256 sized image as opposed to the FFT Radix-2 algorithm which took not more than 1(0.8) second to compute. Following this up we turned our attention to the *Radix-2 Decimation In Time Algorithm* which worked out really well in practice due to its impressive running time complexity of $\mathcal{O}(MN\log N + NM\log M)$; where [M,N] is the size of the image. It uses the *Divide and Conquer* approach in its implementation. The algorithm to the same follows up in the approach section.

III. Experiments and Results

This section contains the summary of all the experiments performed and the reasoning behind the results. This section discusses not only each and every filter in detail, practically but also compares them over a diversified set of sample images.

A. *Inverse Filtering*

As mentioned in previous section that inverse filtering suffers from a severe drawback of noise, which is in turn amplified due the low-pass nature of the degradation function. This very effect can be directly observed from the results, as shown below in Fig-2.

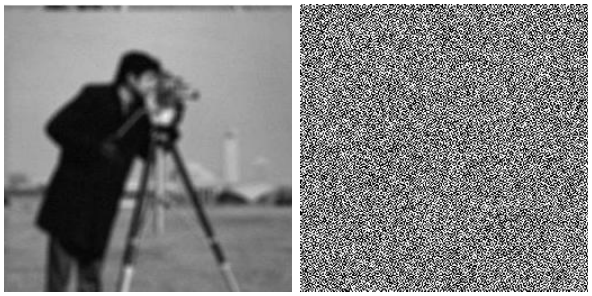

Fig. 2: Degraded Image (left); Restored Image(right)

B. *Truncated Inverse Filtering*

This method, as explained previously overcomes the drawback of inverse filtering by rejecting out all the higher frequencies. This is turn induces a variety of problems ranging from the problem of removal/smoothing of edges in the image to the problem of introduction of gibb's phenomenon, as shown below.

Fig. 3: Ground Truth Image (left); Degraded Image (right)

Fig. 4: Restored Images (Image 2)
Radius=30 (top left); Radius=40 (top right)
Radius=50 (bottom left); Radius=60 (bottom right)

Clearly, as shown above in Fig-3, increasing the radius of acceptance of frequencies leads to an increase in the noise content of the image, while choosing a rather low radius, will lead to the introduction of the so called *Ringing Effects* in the image.

C. *Wiener Filtering*

The following points are to be noted:

- Low values of K (parameter) leads to a more noisy reconstruction owing to the fact that at low K (¡¡1) wiener filter approximates nothing but the naive inverse filter
- High values of K can again be harmful as it causes more than required smoothing of the image which again leads to an imbalance in the trade-off of *Noise-smoothing* vs *Inverse Filtering*

Fig. 5: Ground Truth Image (left); Degraded Image (right)

Fig. 6: Restored Images (Image 2)
K=0.01 (top left); K=0.04 (top right)
K=0.08 (bottom left); K=0.15 (bottom right)

D. Constrained Least Squares Filtering

The constrained least squares method, in a way mimics or even emulates the methodology of Wiener filtering, expect for a minor change in the cost function and hence in the denominator of the fourier transform of the restored image. Therefore, similar to the wiener filtering, with the increase in the parameter α, image starts to get a bit more smooth while for lower values noise starts to appear in the image which leads to the distortion of all the fine details of the underlying image, as shown below, in Fig-7

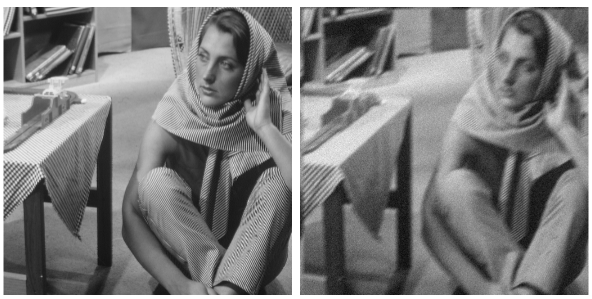

Fig. 7: Ground Truth Image (left); Degraded Image (right)

Fig. 8: Restored Images (Image 2)
α=0.01 (top left); α=0.05 (top right)
α=0.1 (bottom left); α=0.15 (bottom right)

E. Comparison

Here, we have considered 4 different images degraded with 4 different kernels. Each image in the set is restored using each of the 4 filters with parameters set in such a way that maximize the similarity metrics such as SSIM, PSNR in the best possible way along with visual appeasement. The table is shown as follows:

Filters	Image 1	Image 2	Image 3	Image 4
Inverse	3.376 dB	5.06 dB	4.62 dB	5.11 dB
	0.008	0.003	0.02	0.009
Truncated	17.9 dB	12.53 dB	16.24 dB	12.93 dB
	0.628	0.38	0.65	0.529
Wiener	18.277 dB	13.1 dB	16.78 dB	13.3 dB
	0.64	0.429	0.702	0.55
CLS	18.25 dB	12.8 dB	16.47 dB	13.1 dB
	0.657	0.41	0.6873	0.534

TABLE I: Comparison of filters using PSNR (in dB) and SSIM

Fig. 9: Ground Truth Image 1(left); Degraded Image 1(right)

Fig. 10: Restored Images (Image 1)
Inverse Filter (top left); Truncated Inverse (top right)
Wiener Filter (bottom left); CLSF (bottom right)

Fig. 11: Ground Truth Image 2(left); Degraded Image 2(right)

Fig. 12: Restored Images (Image 2)
Inverse Filter (top left); Truncated Inverse (top right)
Wiener Filter (bottom left); CLSF (bottom right)

Fig. 13: Ground Truth Image 3(left); Degraded Image 3(right)

Fig. 14: Restored Images (Image 3)
Inverse Filter (top left); Truncated Inverse (top right)
Wiener Filter (bottom left); CLSF (bottom right)

Fig. 15: Ground Truth Image 4(left); Degraded Image 4(right)

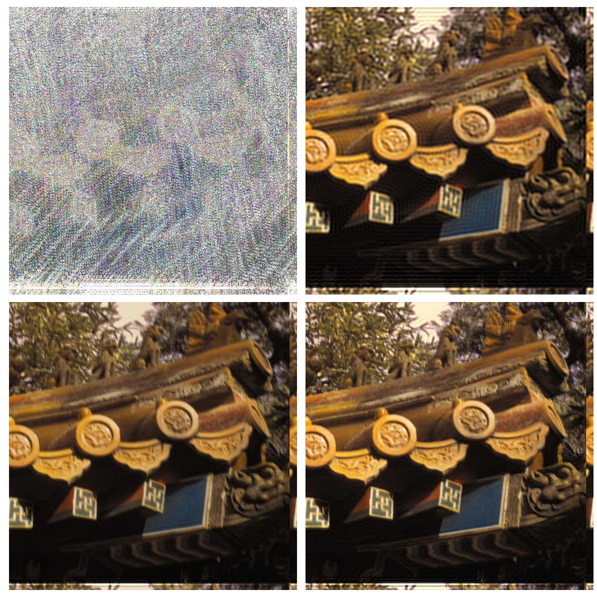

Fig. 16: Restored Images (Image 4)
Inverse Filter (top left); Truncated Inverse (top right)
Wiener Filter (bottom left); CLSF (bottom right)

Following inferences can be made:

- Wiener filtering and CLSF almost always outperform the both inverse and truncated-inverse filtering methodologies
- Inverse filtering can't be used as a stand-alone method due to its poor performance, in all the cases, which easily seem to outweigh the ease of computation that the method provides
- PSNR doesn't always hold good, as is evident from the column for Image 1 where cls outperforms wiener in visual perceptions and SSIM, both, while PSNR seems to predict the opposite sequence.
- SSIM is not a good metric to be always used to evaluate the restoration, as is evident in examples 2,3,4 where CLSF clearly outperforms Wiener filtering visually but not in terms of the metric.

F. Custom Image

Here, I chose one of my own (degraded due to motion blur) pics from my childhood. To restore the image, following steps are followed, in order:

1) A patch is cut out of the degraded image such that the former has a high signal-to-noise ratio along with some simple structures/geometrical shapes inside it, such as constant regions etc so that they can be reproduced easily
2) An approximate patch is considered that represents the same patch region of the underlying ground truth (or initial un-degraded) image
3) Henceforth, $H(u, v)$ is estimated by observation and with the assumption of space invariance of the impulse response, $h(x, y) = IFFT(H(u, v))$ is padded with zeros and used for the overall image for image restoration
4) Post estimation of the degradation function, all the aforementioned methods are applied and compared for best performance

Following up are the results at each step, obtained in order (Note: Best results were achieved in the case of wiener filtering):

Fig. 17: Degraded Image

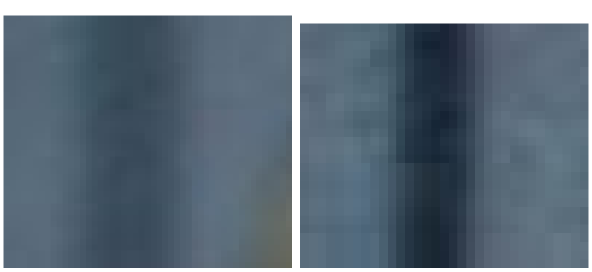

Fig. 18: Noisy Patch considered(left). Restored Image(right)

Fig. 19: Reconstructed Image

IV. Discussion and Conclusions

It is important to note that not always do the similarity metrics such as PSNR, SSIM, MSE etc, hold true for comparisons. The main reasons being the underlying assumptions for each method/metric may or may not even be valid in certain scenarios, which makes the problem of image restoration even harder.

It is evident from the previous discussions that Image restoration in indeed a tough nut to crack, especially in cases where their is no prior information available about the degradation function. Hence, owing to the un-predictability, difficulty and importance of the task of image restoration and its widespread applications in almost every area of research and recreation, I believe that this paradigm can very well be tackled in the domain of Machine Learning/Deep Learning/AI.

V. Relevant Links

- Code - Github Repository

VI. Reference

- Class Notes and Lecture Slides
- Inverse Filtering
- Wiener Filtering
- CLS Filtering
- Image Restoration techniques
- FFT Implementation
- General Resource - Wikipedia