EENG 860 Special Topics: Digital Image Processing
Project 2

Instructor: Dr. Ahmadreza Baghaie

Due Date: May 11, 2020

Image Restoration Techniques in Fourier Domain

Image restoration is a widely popular area of image processing with applications in various branches of science. In the lectures, image degradation and restoration were discussed in details and several methods, both in spatial and frequency domain, were introduced for reducing image degradation. In this project, you are asked to write MATLAB codes for restoration of images degraded by blurring and additive Gaussian noise. More specifically, you will implement three methods: 1) Filtered Inverse Filtering (FIF), 2) Wiener Filtering (WF) and 3) Constrained Least Squares Filtering (CLSF).

Assuming $f(x, y)$, $h(x, y)$ and $n(x, y)$ as the original undegraded image, the impulse response or the point spread function (PSF) of the linear position-invariant (LPI) degradation operator and the additive noise, respectively, the degraded image $g(x, y)$ in spatial domain is generated as:

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

In the Fourier domain, this can be represented as:

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

where each term is the Fourier transform of its corresponding spatial form.

In inverse filtering, the estimate of the original image in the Fourier domain, $\hat{F}(x, y)$, is generated by:

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

This formulation is heavily dependent on the level of noise in the image. However, it is possible to reduce the effects of noise in the reconstruction by limiting the frequency components only to ones close to the Fourier domain's center by applying a low-pass filter as follows:

$$\hat{F}(u, v) = \frac{G(u, v)}{H(u, v)} H_{D_0}(u, v) \tag{4}$$

where $H_{D_0}(u, v)$ represents the frequency transfer function of the the low-pass filter with cutoff frequency D_0.

In Wiener filtering the restoration is done by:

$$\hat{F}(u, v) = \left[\frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + |N(u, v)|^2/|F(u, v)|^2}\right] G(u, v) \tag{5}$$

where $|H(u, v)|^2$, $|N(u, v)|^2$ and $|F(u, v)|^2$ are the power spectra of the degradation operator, noise and undegraded image, respectively. This formulation can be simplified by assuming white noise with constant spectrum and also having a constant ratio between the power spectrum of the noise and the power spectrum of the undegraded image, as follows:

$$\hat{F}(u, v) = \left[\frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + K}\right] G(u, v) \tag{6}$$

In Constrained Least Squares Filtering, the Laplacian operator is used as a measure of smoothness of the image. In this formulation the estimate of the undegraded image in the Fourier domain is generated by:

$$\hat{F}(u,v) = \left[\frac{H(u,v)^*}{|H(u,v)|^2 + \gamma|P(u,v)|^2}\right] G(u,v) \tag{7}$$

where $H(u,v)^*$ is the complex conjugate of $H(u,v)$, and $P(u,v)$ is the Fourier transform of the Laplacian kernel:

$$p(x,y) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{8}$$

It should be noted that for the formulation to be valid, the two terms in the denominator should have exactly the same size. So the Laplacian kernel needs to be zero-padded accordingly.

Project Description

In this project you are asked to write implementations for FIF, WF and CLSF for restoration of images degraded by motion blurring and additive noise. A set of 512×512 images is provided with the project description. In addition to that, a '`.mat`' file is provided which contains three blurring patterns namely, Gaussian blur, linear motion blur, and non-linear motion blur as shown in Figure 1.

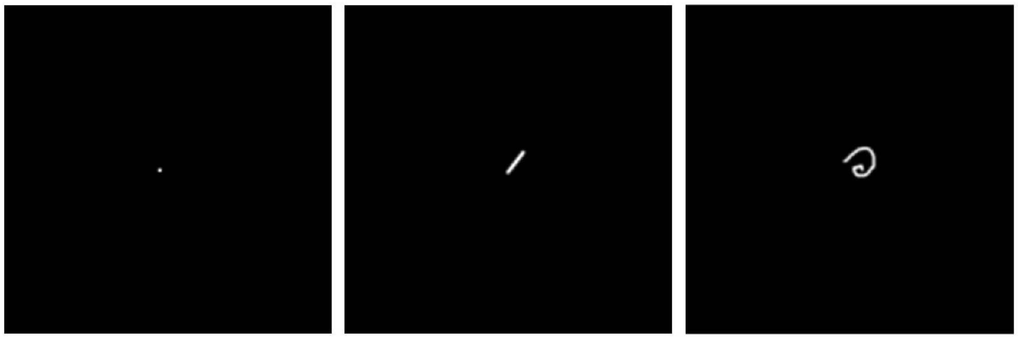
Figure 1: Blur kernels

The code should read the input image and load the '`.mat`' file, apply the degradation to the image followed by applying the additive noise. It is recommended that each of the three restoration methods is implemented as a separate MATLAB function, with appropriate set of inputs, that generates the estimate of the undegraded image as output. There should also be a MAIN script that reads the input images, applies the degradation and noise, calls on the three restoration functions and stores the results in appropriately named variables, and then displays them side-by-side in a figure. Of course the final figure can also contain the original image, the degraded image and the degraded and noisy image for better comparison. A sample is shown in Figure 2.

The associated parameter for each method, D_0 for FIF, K for WF and γ for CLSF, should be tuned to achieve the best result. To assess the performance of the methods, you can use Peak-Signal-to-Noise-Ratio (PSNR), `psnr` function in MATLAB, with the original and restored images as the inputs to obtain a quantitative measure of accuracy of the restoration. Keep in mind that because of the random nature of noise, every time you run your code the noise component changes and consequently the value of PSNR changes slightly.

The results should be presented in a short report, with proper explanations on the effects of changing the parameters. The report should be accompanied by the MATLAB codes as separate M-files that produce the final results without any need for manipulation of the parameters. The codes should be well-commented so I can assess your knowledge of the material. You can use the sample images that I have uploaded for your tests, or use any grayscale image with size 512×512 that you want. In case the image that you want

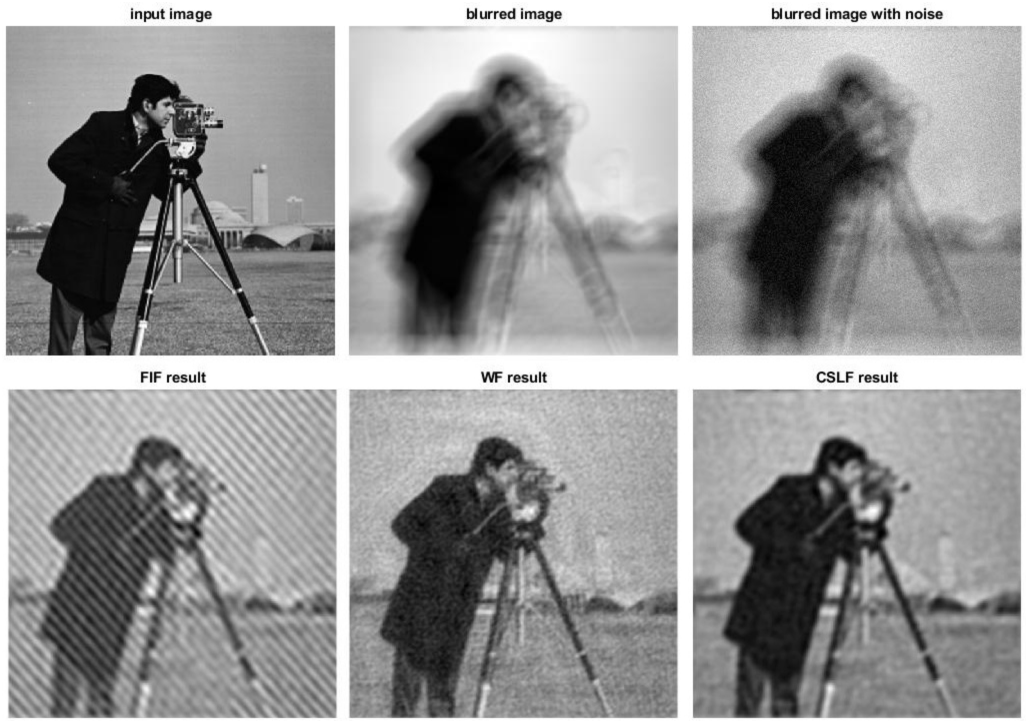

Figure 2: Sample results

to use is large, you can use `imresize` to change its size. Also, you can use `tic` and `toc` functions to record and compare the execution times of the methods. All the materials should be uploaded as a single ZIP file to Blackboard by the deadline.

Implementation Notes

- To load the '`.mat`' file containing the blur kernels, use the `load` function.
- Re-scale the input image to the range [0,1] using `mat2gray` function at the beginning of your code.
- To apply the degradation operator to the input image, you can use `conv2` function. However, it is recommended to use `imfilter` with `circular` option to account for the periodicity assumptions discussed in the class.
- Noise can be added using `imnoise` function. Make sure the noise is zero-mean, with variance in the range 0.1-5%.

Useful MATLAB Functions

```
imread         imwrite    mat2gray     rgb2gray    imshow
imagesc        subplot    imhist       histeq      imadjust
adaphisteq     padarray   conv2        imboxfilt   imgaussfilt
medfilt2       stdfilt    imbilatfilt  imsharpen   imnoise
psnr           ssim       fft2         fftshift    imcomplement
imreducehaze   fspecial   imfilter     imresize    imwarp
```