IMAGE DEBLURRING

Submitted by
Vishal Nagarale (201451004)
N Santosh (201451063)
Anil Chauhan (201451043)
Prasad Wajekar (201451002)

Under Supervision of
Prof. Ajay Nath

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY,
VADODARA

November 9, 2016

Contents

Introduction

Overview

This report discuss about image deblurring.Image deblurring at its basics is taking any image that is not sharply focused and processing it to make it more clear to the viewer.

Background and Motivation

Restoring an image that is "blurred" is one of the most highly publicized parts of image processing. There are several different kinds of image distortions that occur when you take a picture: noise, incorrect focusing, white balance error, exposure error, lens distortion, motion blur, and more. Most people would like their photos to come out sharp and focused, so that viewers can easily see what the photo is about. In our project we will focus mostly on motion blur. Most of the other distortions are well known and correctly, but the deblurring of images that are focused incorrectly or have camera movement is still very much in development. We will mostly explore incorrect focus and various techniques currently available to deblur images.

Objective

To deblur a given image with mathematical approach.

Project Description

Steps for image blurring

1. Read an image into the MATLAB workspace.

MATLAB has its pre-defined functions to read an image. The image size should be less than 0.5MB for better results.

```
I = imread('filename.tif');
figure,imshow(I)
```

2. Create the PSF.

Point Spread Function describes the response of an imaging system to a point source or point object. This approximates the linear motion of camera with some standard deviation. We have to guess it initially.

```
PSF = fspecial('motion',13,45);
figure,imshow(PSF[],'InitialMagnification','fit');
```

3. Create the blur image.

After getting PSF which describes the degree to which an optical system blurs(spreads) a point of light, we convolve it with original image.

```
Blurred = imfilter(I,PSF,'circ','conv');
figure,imshow(Blurred)
```

4.Deblur the image.

To deblur the image using the **deconvblind** function, We must make an initial guess at the PSF. Examine the blured image and measure the width of a blur around an obviously sharp object. Now we define a new PSF of same size of the original image.

```
INITPSF = ones(size(PSF));
```

```
[ J P ] = deconvblind(Blurred,INITPSF,30);
imshow(J);
```

Now to acquire good result we have to add weight array to exclude areas of high contrast from the original image. This array contains same pixel value as in original image excluding the high contrast pixel values. Refine the guess at the PSF. The reconstructed PSF returned by the first pass at deconvolution,p, shows a clear linearity. For the second pass small amplitude pixels set to zero.

```
WEIGHT = edge(I,'sobel',.28);
se1 = strel('disk',1);
se2 = strel('line',13,45);
WEIGHT = ~imdilate(WEIGHT,[se1 se2]);
WEIGHT = padarray(WEIGHT(2:end-1, 2:end-1),[1 1]);
fighure,imshow(WEIGHT);
P1 = P;
P1(find(P1 < 0.01)) = 0;
```

Run the deconvolution again,this time specifing the weight array and the modified PSF.

```
[J2 P2] = deconvblind(Blurred, P1, 50,[],
double(WEIGHT));
figure,imshow(J2);
```

Blind Deconvolution Algorithm

The algorithm maximizes the likelihood that the resulting image, when convolved with the resulting PSF, is an instance of the blurred image, assuming poisson noise statistics. The blind deconvolution algorithm can be used effectively when no information about the distortion (blurring and noise) is known.

The convolution equation is -

$$g(x,y) = h(x,y) * f(x,y) + n(x,y)$$

Blind deconvolution is a deconvolution algorithm that works without specific knowledge of the impulse response function used in the convolution. Blind deconvolution permits the recovery of a blurred image without the need for a specific point spread function. In its simplest form the blind deconvolution algorithm is an iterative function, that takes an initial guess for the point spread function and the blurred image, and returns a deblurred image and a new guess for the point spread function. It can be converted to an iterative algorithm, where each iteration processes the initial image with the new hopefully improved guess for the point spread function. It is important to note that the blind deconvolution algorithm has an optimal number of iterations for deblurring an image, and it is possible to have too many iterations causing the deblurred image to appear worse than if less iterations would have. Thus, the blind deconvolution algorithm must be tuned by giving a good initial guess for the PSF, as well as the correct number of iterations in order to successfully deblur an image. A snippet from Ayers and Dainty's first paper on blind deconvolution is included below. This shows the state diagram used to iteratively compute the deblurred image.

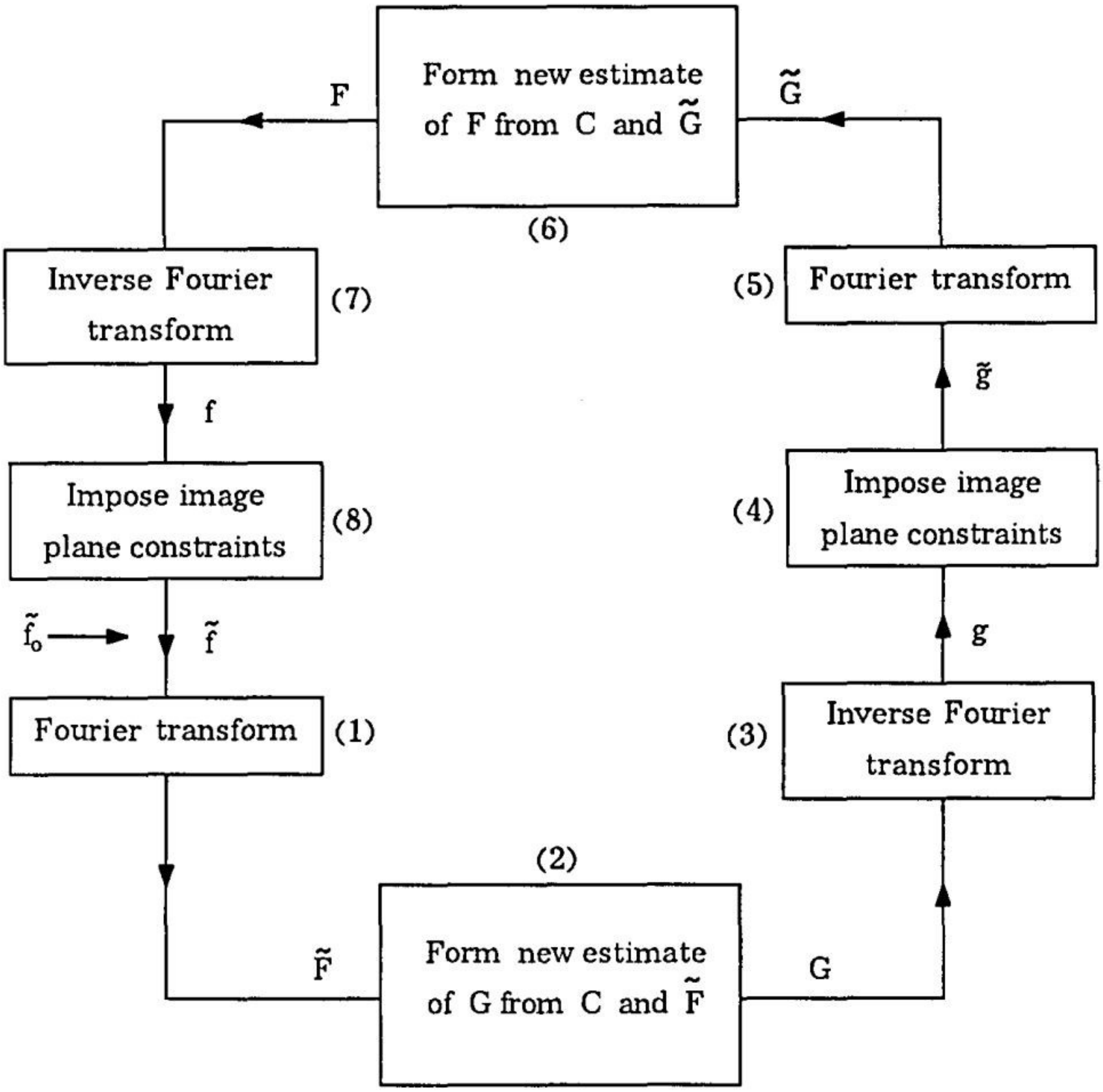


Fig. 1. General deconvolution algorithm.

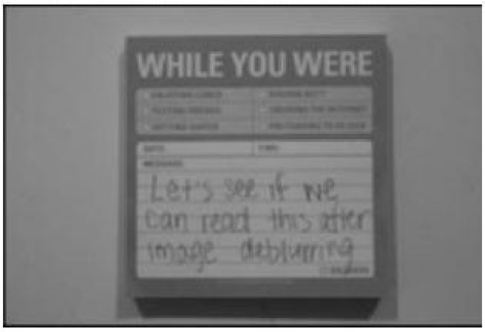
Tools

MATLAB is the self-sufficient tool for it. All required functions for image processing are in-built in it so we studied them and used them in our project.

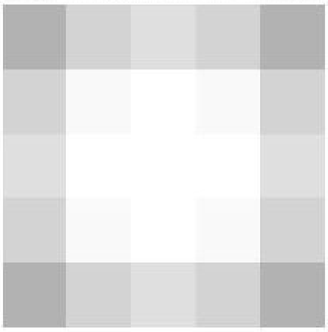
Result

original image

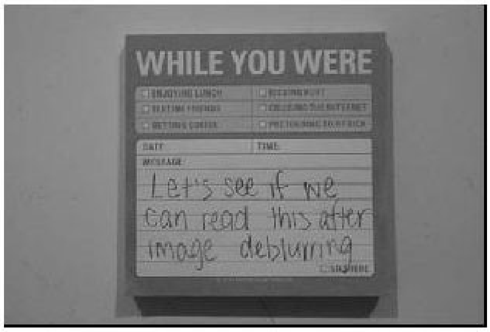
Gaussian blurred image with variance = 40



Blur kernel for gaussian with variance=3

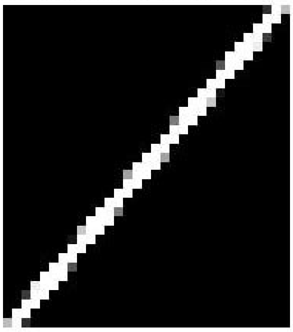


Inverse ifft2 of the gausian blurred image divided by its PSF

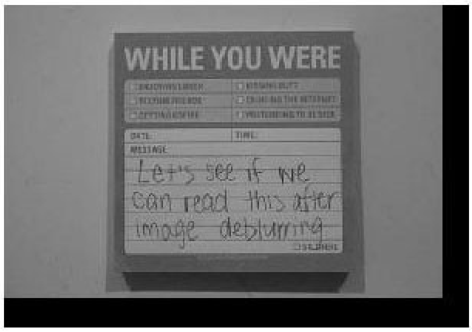


Motion blurred image with LEN = 50 and THETA = 45

Blur kernel of motion blur of length=50 and theta=45

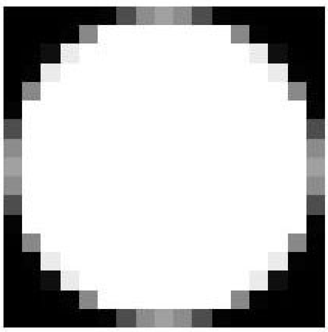


Inverse ifft2 of the motion blurred image divided by its PSF

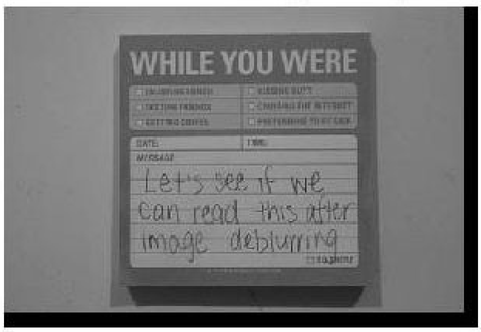


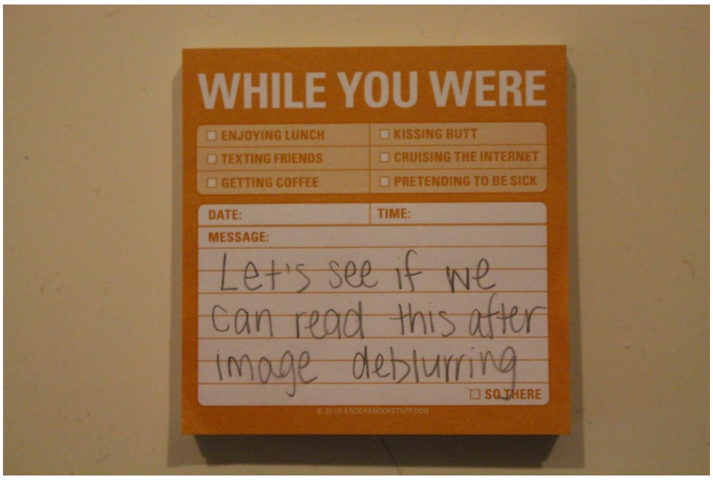
Disk blurred image with Radius=8

Blur kernel for a disk blur of radius=8



Inverse ifft2 of the disk blurred image divided by its PSF





From the above result we successfully able to deblur the images under given circumstances.

Conclusion

Estimation of PSF, noise levels, and number of iterations was a process of repitition, guessing, and checking. One long standing issue in image deblurring with these advanced techniques is what the optimal inputs are for an image and how to quantify that at the output

References

- https://en.wikipedia.org/wiki/Blind_deconvolution
- Deblurring Images Matrices, Spectra and Filtering by Hansen. Nagy, O'Leary
- Mathworks Documentation.