Blurring and deblurring digital images using the dihedral group

Husein Hadi Abbas Jassim
Zahir M. Hussain
Edith Cowan University
Hind R.M. Shaaban
Kawther B.R. Al-dbag

Follow this and additional works at: https://ro.ecu.edu.au/ecuworkspost2013

Part of the Applied Mathematics Commons, and the Artificial Intelligence and Robotics Commons

10.14569/IJARAI.2015.041204
This Journal Article is posted at Research Online.
Blurring and Deblurring Digital Images Using the Dihedral Group

Husein Hadi Abbas Jassim
Faculty of Computer Science and Mathematics
University of Kufa
Najaf, Iraq

Zahir M. Hussain
Faculty of Computer Science and Mathematics
University of Kufa
Najaf, Iraq

Adj. Prof., School of Engineering, ECU, Australia

Hind R.M. Shaaban
Faculty of Computer Science and Mathematics
University of Kufa
Najaf, Iraq

Kawther B.R. Al-dbag
Faculty of Computer Science and Mathematics
University of Kufa
Najaf, Iraq

Abstract—A new method of blurring and deblurring digital images is presented. The approach is based on using new filters generating from average filter and H-filters using the action of the dihedral group. These filters are called HB-filters; used to cause a motion blur and then deblurring affected images. Also, enhancing images using HB-filters is presented as compared to other methods like Average, Gaussian, and Motion. Results and analysis show that the HB-filters are better in peak signal to noise ratio (PSNR) and RMSE.

Keywords—Dihedral group; Kronecker Product; motion blur and deblur; digital image

I. INTRODUCTION

This template, There are three main categories of image processing, image enhancement, image compression and restoration and measurement extraction [3,6]. A digital image is divided into pixels. Each pixel has a magnitude that represents intensity. The camera uses the recorded image as a faithful representation of the scene that the user saw, but every image is more or less blurry. Blurring may arise in the recording of image, because it is unavoidable the scene information "spills over" to neighboring pixels. When there is motion between the camera and image objects during photographing, the motion blur the image. In order to recover motion-blurred images, mathematical model of blurring process are used [1]. Many authors studied motion blur. Often, it is not easy or convenient to eliminate the blur technically. Mathematically, motion blur is modeled as a convolution of point spread function (filters) denoted by (PSF) with the image represented by its intensities. The original image must be recovered by using mathematical model of the blurring process which is called image deblurring [7]. Many researchers introduced algorithms to remove blur such as Average filter AF (or Mean filter), Gaussian filter (GF). The Gaussian filter is equivalent to filtering with a mask of radius R, whose weights are given by Gaussian function: \( (x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}} \), where \( \sigma \) is stander deviation of the Gaussian: large \( \sigma \) for more intensive smoothing) [2]. Motion Blur effect filter is a filter that makes the image appear to be moving by adding a blur in a specific direction [10]. The largest subgroup H of dihedral group \( D_3 \) is found in [4].

In this work, Markov basis \( HB \) is used to introduce a new filters from Average filter for adding and removing motion blur of image, denoted by \( HB \)-filters.

II. PRELIMINARY CONCEPTS

This section reviews the preliminaries about H-filters, Dihedral group, Convolution and Deconvolution processes.

A. H-Filters

H-filters are 18 elements as per the following set [5].

\[
\begin{align*}
Z_1 &= \begin{bmatrix} 1 & -1 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} ; \\
Z_2 &= \begin{bmatrix} 0 & 0 & 0 \\ -1 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} ; \\
Z_3 &= \begin{bmatrix} 1 & 0 & -1 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} ; \\
Z_4 &= \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} ; \\
Z_5 &= \begin{bmatrix} -1 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix} ; \\
Z_6 &= \begin{bmatrix} 1 & 0 & -1 \\ 0 & -1 & 1 \\ 0 & 0 & 0 \end{bmatrix} ; \\
Z_7 &= \begin{bmatrix} 0 & 0 & 1 \\ -1 & -1 & 0 \\ -1 & 0 & 0 \end{bmatrix} ; \\
Z_8 &= \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} ; \\
Z_9 &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} ; \\
Z_{10} &= \begin{bmatrix} 1 & 0 & -1 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} ; \\
Z_{11} &= \begin{bmatrix} -1 & 0 & -1 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} ; \\
Z_{12} &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} ; \\
Z_{13} &= \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} ; \\
Z_{14} &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} ; \\
Z_{15} &= \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} ; \\
Z_{16} &= \begin{bmatrix} 0 & 0 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix} ; \\
Z_{17} &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} ; \\
Z_{18} &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.
\end{align*}
\]
Definition 1: Dihedral Group

Let \( n \) be a positive integer greater than or equal to 3. The group of all symmetries of the regular polygon with \( n \) sides, including both rotations and reflections, is called dihedral group and denoted by \( D_n \). The 2n elements of \( D_n \) can be written as: \( \{ e, r, r^2, \ldots, r^{n-1}, s, sr, sr^2, \ldots, sr^{n-1}\} \), where \( e \) is the identity element in \( D_n \). In general, we can write \( D_n \) as:

\[
ds_n = \{ s^k r^l : 0 \leq k \leq n-1, 0 \leq l \leq 1 \}
\]

which has the following properties:

\[
r^n = 1, \quad s^k r^l = r^{-k} (s^k)^2 = e, \quad \text{for all } 0 \leq k \leq n-1.
\]

The composition of two elements of the \( D_n \) is given by \( r^i r^j = r^{i+j} \), \( s^i r^j = s r^j \), \( s^i s r^j = r s r^j \). 

C. 2D Convolution

Assume two discrete 2-dimensional images \( f(x, y) \) and \( h(x, y) \). Their convolved (or folded) sum is the image \( g(x, y) \), the convolution of these two functions is defined as [12]:

\[
g(x, y) = f(x, y) \otimes h(x, y), \quad \text{so}
\]

\[
f(x, y) \otimes h(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) h(x-m, y-n)
\]

For \( 0 \leq x, m \leq M-1; 0 \leq y, n \leq N-1 \),

where \( M \times N \) is a size of \( h(x, y) \).

III. 2D DISCRETE FOURIER TRANSFORM

The two-dimensional discrete Fourier transform (DFT) of the image function \( f(x, y) \) is defined as:

\[
F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi \frac{ux}{M} \frac{vy}{N}} \tag{2}
\]

where \( f(x, y) \) is a digital image of size \( M \times N \), and the discrete variable \( u \) and \( v \) in the ranges: \( u = 0, 1, 2, \ldots, M-1 \) and \( v = 0, 1, 2, \ldots, N-1 \). 

Given the transform \( F(u, v) \), we can obtain \( f(x, y) \) by using the inverse discrete Fourier transform (IDFT):

\[
f(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi \frac{ux}{M} \frac{vy}{N}} \tag{3}
\]

It can be shown by direct substitution into Eq. 2 and Eq. 3 that the Fourier transform pair satisfies the following translation properties:

\[
f(x-m, y-n) \leftrightarrow F(u, v)e^{-j2\pi \frac{um}{M} \frac{vn}{N}} \tag{4}
\]

Now, interested in finding the Fourier transform of Eq. 1:

\[
\mathcal{F} \{ f(x, y) \} h(x, y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} f(u, v) h(x-m, y-n) e^{-j2\pi \frac{ux}{M} \frac{vy}{N}}, \text{so by Eq. 4 we have,}
\]

\[
\mathcal{F} \{ f(x, y) \} h(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) H(u, v) e^{-j2\pi \frac{um}{M} \frac{vn}{N}} = F(u, v) H(u, v). \tag{5}
\]

This result of the convolution theorem is written as:

\[
f(x, y) \otimes h(x, y) \leftrightarrow F(u, v) H(u, v) \tag{6}
\]

The transform of the original image simply by dividing the transform of the degraded image \( G(u, v) \), by the degradation function \( H(u, v) \) is:

\[
\hat{G}(u, v) = \frac{G(u, v)}{H(u, v)} \tag{7}
\]

That’s called inverse filter [9].

A. Fourier Spectrum

Because the 2-D DFT is complex in general [8], it can be expressed in polar form:

\[
F(u, v) = |F(u, v)| e^{j\theta(u, v)}
\]

where the magnitude,

\[
|F(u, v)| = \sqrt{R^2(u, v) + I^2(u, v)} \tag{7}
\]

is called the Fourier (or frequency) spectrum. The power spectrum is defined as,

\[
P(u, v) = |F(u, v)|^2 = R^2(u, v) + I^2(u, v).
\]

As before, \( R \) and \( J \) are the real and imaginary parts of \( F(u, v) \) and all computations are carried out for the discrete variables \( u = 0, 1, 2, \ldots, M-1 \) and \( v = 0, 1, 2, \ldots, N-1 \). Therefore,

\[
|F(u, v)|, \Phi(u, v), \text{and } P(u, v) \text{ are arrays of size } M \times N.
\]

B. Image Restoration based on Wiener Deconvolution

The method considers images and noise as random variables, and the objective is to find an estimate \( \hat{f} \) of the uncorrupted image \( f \) such that the mean square error (MSE) between them is minimized. This error measure is given by:

\[
e^2 = \mathbb{E} \left\{ (f - \hat{f})^2 \right\}
\]

Based on these conditions, the minimum of the error function in Eq. 8 is given in the frequency domain by the expression:

\[
\hat{f}(u, v) = \left[ \frac{H^*(u, v)S_r(u, v)}{S_f(u, v)|H(u, v)|^2 + S_n(u, v)} \right] G(u, v)
\]

\[
= \left[ \frac{1}{|H(u, v)|^2 + S_n(u, v) / S_f(u, v)} \right] G(u, v) \tag{9}
\]

The terms in Eq. 9 are as follows:

\[
H(u, v) = \text{degradation function} \quad H^*(u, v) = \text{complex conjugate of } H(u, v) \quad |H(u, v)|^2 = \text{power spectrum of the noise} \quad \text{and} \quad G(u, v) = \text{the transform of the degraded image. Note that if the noise is zero, then the noise power spectrum vanishes and the Wiener filter reduces to the inverse filter.}
\]

IV. THE PROPOSED APPROACH

\( H \)-filters are used to generate \( HB \)-filters by adding each element in \( H \)-filters to the average filter, so we got some \( HB \)-filters with dimensions 3-by-3 and each of which has type of blur different from the other.

Then the \( HB \)-filters can be extended using tensor product (by operation \( \otimes \) ) to larger sizes, in order to get a higher degrees of blur in digital images. Take any one of \( HB \)-filters \( h(x, y) \) of dimension 3-by-3 and extend it by identity matrix \( I_n \), \( n \)-by-\( n \) where \( n \) is an odd number greater than or equals 3, by Tensor Product \( T \):

\[
T(x, y) = h(x, y) \otimes I_n(x, y)
\]

\[
= \begin{bmatrix}
I_n & I_n & I_n \\
I_n & I_n & I_n \\
I_n & I_n & I_n \\
\end{bmatrix}
\]

www.ijarai.thesai.org
This filter will be called **extended HB-filter** generated from HB-filter \( h(x,y) \) and \( I_n \).

**Example 1.**

Choose any one of \( H \)-filters: \( z_2 = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -1 & 0 \\ -1 & 1 & 0 \end{bmatrix} \).

Divide \( z_2 \) by 9, and add it to the average filter \( (A_f) \) as follows:

\[
h_1 = z_2 + A_f = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -1 & 0 \\ -1 & 1 & 0 \end{bmatrix} \div 9 + \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \div 9 = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 0 & 1 \\ 0 & 2 & 1 \end{bmatrix} \div 9 \in \text{HB-filters}.
\]

So, \( h_1 = \begin{bmatrix} 1 \\ 0 \\ 2 \\ 2 \end{bmatrix} \div 9 = h_2 \).

Similarly, one obtains \( \begin{bmatrix} 1 \\ 0 \\ 2 \\ 1 \\ 0 \\ 2 \end{bmatrix} \div 9 \in \text{HB-filters}. \)

Most HB-filters can be obtained using other \( H \)-filters. For example the **extended HB-filters** generated from HB-filter

\[
h(x,y) = \begin{bmatrix} 2 & 0 & 1 \\ 0 & 2 & 1 \end{bmatrix} \] with \( I_3 \) is given by

\[
I_n \times 9 = \begin{bmatrix} 1 \\ 0 \\ 2 \\ 2 \\ 1 \end{bmatrix} \div 9 = h_2.
\]
A. Blurring

This sub-section describes the standard filters algorithm for addition blur of an image by using the convolution theorem.

**Blur algorithm**

Consider an image matrix $f(x, y)$ of dimension $m$-by-$n$, which can be written as follows:

$$f(x, y) = \begin{bmatrix} f_{11} & \cdots & f_{1n} \\ \vdots & \ddots & \vdots \\ f_{m1} & \cdots & f_{mn} \end{bmatrix}_{m \times n}$$

And **HB-filter** $h(x, y)$ $p$-by-$q$ dimension as defined by $h(x, y) = \begin{bmatrix} h_{11} & \cdots & h_{1q} \\ \vdots & \ddots & \vdots \\ h_{p1} & \cdots & h_{pq} \end{bmatrix}_{p \times q}$

**Step1:** In the beginning add $f(x, y)$ by $(p-1)$ rows with zeros from up and down, and $(p-1)$ columns with zeros from left and right, such that the result is $(m+2(p-1))$-by-$(n+2(q-1))$ dimensions, as follows:

$$f(x, y) = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ f_{11} & f_{12} & \cdots & f_{1n} & \vdots & \vdots & \vdots \\ f_{m1} & f_{m2} & \cdots & f_{mn} & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 \end{bmatrix}_{(m+2(p-1)) \times (n+2(q-1))}$$

where $i= m+2(p-1)$ and $j= n+2(q-1)$.

**Step2:** Reverse $h(x, y)$ (that used in blurring) for two directions,

$$h(x, y) = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1q} \\ h_{21} & h_{22} & \cdots & h_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ h_{p1} & h_{p2} & \cdots & h_{pq} \end{bmatrix}_{p \times q}$$

**Step3:** Make the two arrays as follows:

$$h(x, y) = \begin{bmatrix} h_{pq} & h_{12} & \cdots & h_{pq} \\ h_{2q} & h_{22} & \cdots & h_{22} \\ \vdots & \vdots & \ddots & \vdots \\ h_{1q} & \cdots & \cdots & h_{1q} \end{bmatrix}_{(p+1) \times (q+1)}$$

**Step4:** Calculate the convolution equation for all pixels of blurred matrix $g(x, y)$:

$$g(x, y) = f(x, y) \ast h(x, y) = \sum_{i=1}^{p} \sum_{j=1}^{q} f(i, j) h(i, j)$$

So,

$$g(1,1) = (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + \cdots + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq})$$

$$= (f_{11} \times h_{11})$$

After that shift the filter $h(x, y)$ as much as one column as follows:

$$g(x, y) = \begin{bmatrix} h_{11} & h_{21} & \cdots & h_{pq} \\ h_{21} & h_{22} & \cdots & h_{pq} \\ \vdots & \vdots & \ddots & \vdots \\ h_{pq} & h_{pq} & \cdots & h_{pq} \end{bmatrix}_{p \times q}$$

**Step5:** Calculate the convolution equation for all pixels of blurred matrix $g(x, y)$:

$$g(x, y) = f(x, y) \ast h(x, y) = \sum_{i=1}^{p} \sum_{j=1}^{q} f(i, j) h(i, j)$$

So,

$$g(1,1) = (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + \cdots + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq}) + (0 \times h_{pq})$$

$$= (f_{11} \times h_{11})$$

Now repeat step 4 to obtain digital image convolution $g(x, y)$ at all times that the two arrays overlap. We continue until we find $g(r, c)$, where $r \& c=m+(p-1)$, then the final form of the blurred matrix $g(x, y)$ is:

$$g(x, y) = \begin{bmatrix} g_{11} & \cdots & g_{1c} \\ \vdots & \ddots & \vdots \\ g_{r1} & \cdots & g_{rc} \end{bmatrix}_{m \times c}$$

**Step6:** Delete from $g(x, y)$ as much as $(p-1)$ rows from up and down, and $(p-1)$ columns from left and right, such that the blurred matrix $g(x, y)$ becomes $m$-by-$n$ in dimension:

$$g(x, y) = \begin{bmatrix} g_{11} & \cdots & g_{1n} \\ \vdots & \ddots & \vdots \\ g_{m1} & \cdots & g_{mn} \end{bmatrix}_{m \times n}$$

**Example 2.**

Suppose the image matrix $f(x, y)$ is:

$$T(x, y) = h(x, y) \ast I_z(x, y) = \begin{bmatrix} 2 & 0 & 1 \\ 0 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix}_{3 \times 3} \ast \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}_{3 \times 3}$$

$$= \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 2 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}_{9 \times 9}$$

Then,

$$h(x, y) = \begin{bmatrix} 0 & \cdots & 0 & q & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \end{bmatrix}_{\infty \times j}$$
We blur this matrix with one of the HB-filters: $h(x, y) = \begin{bmatrix} 2 & 0 & 1 \\ 0 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} / 9$.

**Step 1:** Add two rows from up and down, and two columns from left and right of zeros for the matrix $f(x, y)$, such that becomes 7-by-7 dimension, as follows:

$$f(x, y) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 209 & 90 & 60 & 0 & 0 \\ 0 & 0 & 0 & 77 & 30 & 0 & 0 \\ 0 & 0 & 100 & 46 & 20 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}_{7 \times 7}.$$

**Step 2:** Reverse the filter $h(x, y)$ for two directions:

$$h(x, y) = \begin{bmatrix} 2 & 0 & 1 \\ 0 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} / 9 \implies \begin{bmatrix} 1 & 1 & 1 \\ 2 & 0 & 0 \\ 1 & 0 & 2 \end{bmatrix} / 9.$$

**Step 3:** Make the two arrays, as the following form:

$$h(x, y) = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 0 \\ 1 & 0 & 2 \end{bmatrix} / 9.$$

**Step 4:** Calculate the convolution equation for all pixels of blurred matrix $g(x, y)$:

$$g(x, y) = f(x, y) * h(x, y) = \sum_{m_1=1}^{3} \sum_{n_1=1}^{3} f(m_1, n_1) h(m_1, n_1).$$

Now, $g(1,1) = (209 \times 0.2222) = 26.4444$.

After that, shift the filter $h(x, y)$ as much as one column, then repeat the same step.

So, $g(1,2) = (90 \times 0.2222) = 20$, $g(1,3) = (209 \times 0.1111) + (60 \times 0.2222) = 36.5556$, $g(5,5) = (20 \times 0.2222) = 2.2222$.

The final form of the blurred matrix $g(x, y)$ is:

$$g(x, y) = \begin{bmatrix} 46.4444 & 20 & 36.5556 & 10 & 6.6667 \\ 0 & 63.5556 & 94.8889 & 31.8889 & 10 \\ 45.4444 & 43.4444 & 72.5556 & 37 & 12.2222 \\ 0 & 30.7778 & 33.2222 & 21.4444 & 5.5556 \\ 11.1111 & 16.2222 & 18.4444 & 7.3333 & 2.2222 \end{bmatrix}_{5 \times 5}.$$

**Step 5:** Delete from $g(x, y)$ as much as one row from up and down, and one column from left and right, such that the result is the blurred matrix $g_1(x, y)$ 3-by-3 dimension.

$$g_1(x, y) = \begin{bmatrix} 63.5556 & 49.39 & 31.57 \\ 43.4444 & 72.5556 & 37 \\ 30.7778 & 33.2222 & 21.4444 \end{bmatrix}_{3 \times 3}.$$

### B. Deblurring

Here we express the proposed deblurring method.

**Deblur Algorithm**

Weiner deconvolution for the matrix $g(x, y)$ and $h(x, y)$ is given by:

$$\hat{f}(u, v) = \frac{\frac{1}{H(u,v)} G(u, v)}{\frac{1}{H(u,v)} [\frac{1}{H(u,v)} G(u, v)]}.$$

Suppose there is no noise (i.e. $S_g(u,v) = 0$), then the noise of power spectrum vanishes and the Weiner reduces to the invers filter, so one has: $\hat{f}(u, v) = \frac{G(u,v)}{H(u,v)}$.

**Step 1:** Find Fourier transform of the blurred matrix $g(x, y)$ $r$-by-$c$ dimensions,

$$G(u, v) = \sum_{x=1}^{m} \sum_{y=1}^{n} g(x,y) e^{-j2\pi \left(\frac{ux}{M} + \frac{vy}{N}\right)}.$$

**Step 2:** Find Fourier transform of HB-filter $h(x, y)$.

$$H(u, v) = \sum_{x=1}^{m} \sum_{y=1}^{n} h(x,y) e^{-j2\pi \left(\frac{ux}{M} + \frac{vy}{N}\right)}.$$

If the dimension of $h(x, y)$ is less than dimension of $g(x, y)$, we will add zeros for $h(x, y)$ to create as same as the dimension of the image matrix $g(x, y)$ before doing the transform, such that the result is $m$-by-$n$ in dimension.

**Step 3:** Calculate the transform of estimated image $\hat{f}(u,v)$.

**Step 4:** Find estimated image $\hat{f}(x, y)$ by taking inverse Fourier transform of $\hat{f}(u,v)$, by follows:

$$\hat{f}(x, y) = \frac{1}{MN} \sum_{u=1}^{m} \sum_{v=1}^{n} \hat{f}(u, v) e^{j2\pi \left(\frac{ux}{M} + \frac{vy}{N}\right)}.$$

**Step 5:** Remove zeros from $\hat{f}(x,y)$ as much as $(p-1)/2$ of last rows and columns, where resulted dimensions equal to dimensions original image matrix $f(x,y)$.

**Example 2.** We will take blurred matrix $g(x,y)$ from ex.2,

$$g(x, y) = \begin{bmatrix} 46.4444 & 20 & 36.5556 & 10 & 6.6667 \\ 0 & 63.5556 & 94.8889 & 31.8889 & 10 \\ 45.4444 & 43.4444 & 72.5556 & 37 & 12.2222 \\ 0 & 30.7778 & 33.2222 & 21.4444 & 5.5556 \\ 11.1111 & 16.2222 & 18.4444 & 7.3333 & 2.2222 \end{bmatrix}_{5 \times 5}.$$

with HB-filter $h(x, y) = \begin{bmatrix} 2 & 0 & 1 \\ 0 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix}_{3 \times 3} / 9$.

Now, from the Weiner equation, suppose that $S_g(u,v)/S_f(u,v) = 0$, then the Weiner reduces to the invers filter as following, $\hat{f}(u,v) = \frac{G(u,v)}{H(u,v)}$.

**Step 1:** Find Fourier transform of the matrix $g(x,y)$,

$$G(u, v) = \sum_{x=1}^{m} \sum_{y=1}^{n} g(x,y) e^{-j2\pi \left(\frac{ux}{M} + \frac{vy}{N}\right)}.$$
Now, \(G(1,1) = \sum_{x=1}^{5} \sum_{y=1}^{5} g(x, y) e^{-j2\pi \frac{x}{5} \frac{y}{5}}\)
\[= (g(1,1)e^{-j2\pi \frac{1}{5} \frac{1}{5}}) + (g(1,2)e^{-j2\pi \frac{2}{5} \frac{1}{5}}) + (g(1,3)e^{-j2\pi \frac{3}{5} \frac{1}{5}}) + \ldots + (g(5,5)e^{-j2\pi \frac{5}{5} \frac{5}{5}})\]
\[= 46.4444e^{-j(\frac{2}{5})\pi} + 20e^{-j(\frac{4}{5})\pi} + 36.5556e^{-j(\frac{3}{5})\pi} + 10e^{-j2\pi} + \ldots + 2.22222e^{-j4\pi} = 632 + 0j\]
\(G(1,2) = \sum_{x=1}^{5} \sum_{y=1}^{5} g(x, y) e^{-j2\pi \frac{x}{5} \frac{2y}{5}}\)
\[= -89.44 - 191.15j\]
\(G(1,3) = \sum_{x=1}^{5} \sum_{y=1}^{5} g(x, y) e^{-j2\pi \frac{x}{5} \frac{3y}{5}}\)
\[= 30.94 + 17.24j\]
\(G(5,5) = \sum_{x=1}^{5} \sum_{y=1}^{5} g(x, y) e^{-j2\pi \frac{5x}{5} \frac{5y}{5}}\)
\[= -1.13 - 45.84j\]
So, the final form of \(G(u,v)\) be
\[H(u,v) = \sum_{x=1}^{5} \sum_{y=1}^{5} h(x, y) e^{-j2\pi \frac{x}{5} \frac{y}{5}}\]
\(H(5,5) = \sum_{x=1}^{5} \sum_{y=1}^{5} h(x, y) e^{-j2\pi \frac{5x}{5} \frac{5y}{5}}\)
\[= -0.2828 + 0.0249j\]
So, the final form of \(H(u,v)\) is:
\[H(u,v) = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]
\(f(x,y) = \frac{1}{MN} \sum_{u=1}^{M} \sum_{v=1}^{N} f(u,v) e^{j2\pi \frac{ux}{M} \frac{vy}{N}}\)
\(f(x,y) = \frac{1}{5} \sum_{u=1}^{5} \sum_{v=1}^{5} f(u,v) e^{j2\pi \frac{ux}{5} \frac{vy}{5}}\)
\[= \frac{1}{5} \left( f(1,1) e^{j2\pi \frac{1}{5} \frac{1}{5}} + f(1,2) e^{j2\pi \frac{1}{5} \frac{2}{5}} + f(1,3) e^{j2\pi \frac{1}{5} \frac{3}{5}} + \ldots + f(5,5) e^{j2\pi \frac{5}{5} \frac{5}{5}} \right)\]
\[= \frac{1}{25} \left( (632 + 0j) e^{j(\frac{2}{5})\pi} + (285.83) e^{j(\frac{4}{5})\pi} + (170.67) e^{j(\frac{3}{5})\pi} + (-10.23) e^{j(\frac{1}{5})\pi} \right)\]
\(f(1,2) = 90\)
\(f(1,3) = 60\)
\(f(5,5) = 0\)
Now, the final of estimated image \(f(x,y)\) is
\[f(x,y) = \begin{bmatrix}
209 & 90 & 60 \\
0 & 77 & 30 \\
100 & 46 & 20 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
\end{bmatrix}\]
\(f(x,y)\) is:
\[g(x,y) = \begin{bmatrix}
209 & 90 & 60 \\
0 & 77 & 30 \\
100 & 46 & 20 \\
\end{bmatrix}\]
Now, we give the (original, blurred, estimated) block image to explain the image enhancement in ex.2 and ex.3 as shown in Fig.1.
TABLE I. THE COMPARISON OF BETWEEN DIFFERENT FILTERS

<table>
<thead>
<tr>
<th>Degree of blur</th>
<th>Image blur</th>
<th>Aver. filter</th>
<th>Gauss. filter</th>
<th>Motion filter</th>
<th>Proposed filter</th>
</tr>
</thead>
<tbody>
<tr>
<td>9x9</td>
<td>21.44</td>
<td>7.25</td>
<td>21.45</td>
<td>13.78</td>
<td>45.53</td>
</tr>
<tr>
<td>21x21</td>
<td>18.03</td>
<td>7.01</td>
<td>18.04</td>
<td>12.7</td>
<td>49.9</td>
</tr>
<tr>
<td>27x27</td>
<td>17.02</td>
<td>7.03</td>
<td>17.02</td>
<td>11.79</td>
<td>46.23</td>
</tr>
<tr>
<td>RMSE</td>
<td>9x9</td>
<td>21.61</td>
<td>110.66</td>
<td>21.58</td>
<td>52.18</td>
</tr>
<tr>
<td>21x21</td>
<td>31.98</td>
<td>113.72</td>
<td>31.96</td>
<td>59.1</td>
<td>0.81</td>
</tr>
<tr>
<td>27x27</td>
<td>35.95</td>
<td>113.45</td>
<td>35.94</td>
<td>65.65</td>
<td>1.24</td>
</tr>
</tbody>
</table>

C. Comparison with other filters

HB-filters are compared in PSNR (in dB) and RMSE with the (AF, GF, and MF) filters. The proposed method and the other methods are applied on (256x256) Pepper RGB image by using (jpg, format) as in Table I. The application of proposed method and some other methods on the color images (in jpg, format) of different blur is shown in Fig.2.

V. CONCLUSION

Blur has been added and removed from digital images using HB-filters. The HB-filters perform well for grayscale, binary and color (jpg, png) images with different blur degrees. Results show that the HB method has higher PSNR and less RMSE than Average, Gaussian and Motion methods.

ACKNOWLEDGMENT

We would like to thank the University of Kufa for financial support.

REFERENCES


