International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 6, June - 2013

Iterative Richardson-Lucy Algorithm For The Modeling And Correction Of Blurred Image With A Known Point Spread Function

Miss. Supriya S. Meshram

Priyadarshini college of Engineering,

Nagpur, Maharashtra,India

Prof (Mrs.) P. U. Chati

Priyadarshini College of Engineering

Nagpur, Maharashtra, India

Abstract

This paper addresses how to model and correct the image blur that arises due to defocusing, handshaking, object motion (motion blur) while capturing an image. Non-blind deconvolution is a key component in an image deblurring system . The deconvolution tries to invert the blurring of an image that is modeled by the convolution $g = f * h$. It is the process that obtains a sharp latent image from a blurred image when a point spread function (PSF) is known. However, ringing and noise amplification are inevitable artifacts in image deconvolution since perfect PSF estimation is impossible. The convolutional regularization to reduce these artifacts cannot preserve image details in the deconvolved image when PSF estimation error is large. Hence strong regularization is needed. A simple iterative technique has been developed for non-blind deconvolution using Richardson Lucy Algorithm of two convolved functions. The method is described as follows along with some experimental results. The goal of this paper is to preserve the details of the image, while suppressing ringing and noise artifacts by controlling regularization strength according to the local characteristics of the image.

Key words: PSF, Motion blur, Spatially varying motion blur, Ego motion.

1. INTRODUCTION

Restoration of digital images from their degraded measurement has always been a problem of great interest. A specific solution to the problem of image restoration is generally determined by the nature of degradation phenomena. So it is highly dependent on the nature of the noise present there. Given the noise function, one can use the Richardson-Lucy Algorithm to restore the degraded image. This algorithm was introduced by W.H. Richardson (1972) and L.B. Lucy (1974).

Single image deblurring has gained considerable attention in recent years. Given a blurry input image “b”, existing deblurring approaches model the image degradation as:

$$b = k \otimes l + n; \qquad (1)$$

where “k” is the blur kernel “l” refers to the underlying sharp latent image, “n” is noise, and $\otimes$ is the convolution operator. The task for blind image deconvolution is to infer both “k” and “l” from a single input “b”, which is severely ill-posed. Blind deconvolution approaches intensively use non-blind deconvolution, the process of estimating the latent image “l” given the blurred image “b” and the estimated blur kernel “k”, while inferring “k” and for generating the final output “l”. This makes non-blind deconvolution a key component in the deblurring pipeline. Various non-blind deconvolution approaches have been proposed in the literature, ranging from classic Wiener filter to modern optimization approaches with image priors. However, in practice these methods often produce severe ringing artifacts even when the blur kernel is known or well estimated

1.1 What is an Image?

An image is nothing but a huge collection of numbers known as pixels. In particular a gray image is an image in which the value of each pixel is a single sample i.e. it carries only intensity information. So a pixel in a given image is just the intensity at that particular point. The pixel value is a number between 0 and 1 (both inclusive). 0 denotes the total absence (i.e. black) and 1 denotes the total presence (i.e. white).

1.2Point Spread Function

The Point Spread Function describes the response of an imaging system to a point source or point object. Following is an example of a PSF:

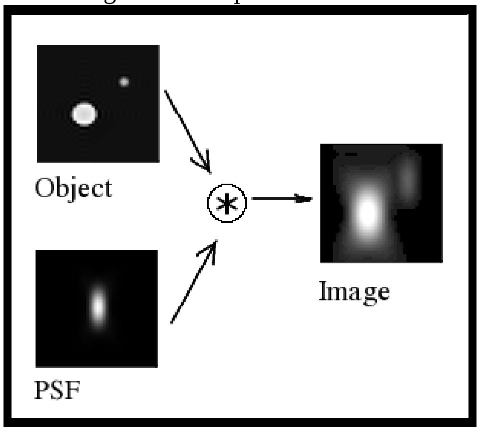


Figure 1: Image formation in a Confocal Microscope: Central longitudinal (XZ) slice, the 3D acquired distribution arises from the convolution of the real light sources with the PSF.

1.3Poisson Distribution

The Poisson Distribution is a discrete probability distribution that expresses the probability of a given number of events occurring in a fixed interval of time and/or space if these events occur with a known average rate and independently of the time since the last event. Poisson Distribution can also be used for the number of events in other specified intervals such as distance, area and volume.

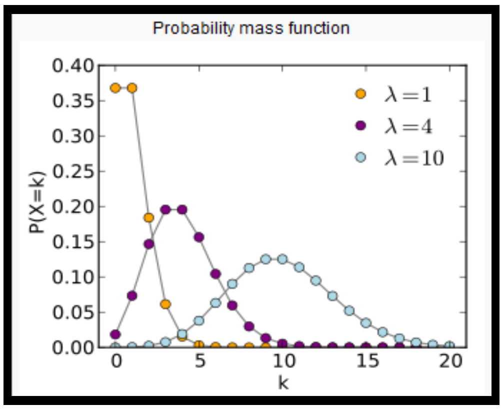


Fig 2. The horizontal axis is the index *k*, the number of occurrences. The function is only defined at integer values of *k*. The connecting lines are only guides for the eye.

Aim of our project can be very well explained from the flow diagram of figure 3, Which follows some steps are:

- Step 1: Load a clear image
- Step 2: Blur the loaded clear image
- Step 3: Obtain clear image by using Richardson Lucy Algorithm
- Step 4: Damping (i.e. Compare initially loaded clear image and step 3 clear image)

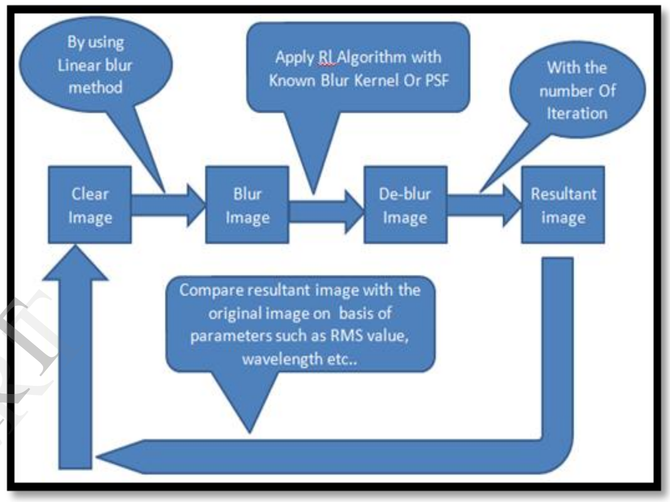

By using Linear blur method
Apply RL Algorithm with Known Blur Kernel Or PSF
With the number Of Iteration
Clear Image
Blur Image
De-blur Image
Resultant image
Compare resultant image with the original image on basis of parameters such as RMS value, wavelength etc..

Fig 3. Flow of Project

2. RELATED WORK

Existing work targeting image blur due to camera ego motion has assumed a global PSF for the entire image. When the blur PSF is known, or can be estimated, well known debluring algorithms, such as Richardson-Lucy and Wiener filter , can be applied to deblur the image. Due to poor kernel estimation or convolution with PSFs that result in unrecoverable frequencies, these conventional deblurring algorithms can introduce undesirable artifacts in the deblurred result such as “ringing” and amplification of image noise.
Consequently, research addressing image deblurring, including camera shake and other types of blur, typically targets either blur kernel estimation or ways to regularize the final result, or both. For example, Dey et al. and Chan and Wong utilized total variation regularization to help ameliorate ringing and noise artifacts. Fergus et al. demonstrated how to use a variational Bayesian approach combined with

gradient-domain statistics to estimate a more accurate PSF. Raskar et al. coded the exposure to make the PSF more suitable for deconvolution. Jia demonstrated how to use an object's alpha matte to better compute the PSF. Levin et al. introduced a gradient sparsity prior to regularize results for images exhibiting defocus blur. This prior is also applicable to motion blurred images. Yuan et al. proposed a multiscale approach to progressively recover blurred details, while Shan et al. introduced regularization based on high-order partial derivatives to reduce image artifacts.
These previous approaches all work under the uniform PSF assumption. Camera ego motion causes a spatially varying motion blur that cannot be accurately modeled with a uniform PSF. Prior work has recognized the need to handle nonuniform motion blur for camera ego motion, moving objects, and defocus blur. For example, early work by Sawchuk addressed motion blur from a moving camera by first using a log-polar transform to transform the image such that the blur could be expressed as a spatially invariant PSF. The range of motion that could be addressed was limited to rotation and translation. When addressing moving objects, the input image can be segmented into multiple regions each with a constant PSF, as demonstrated by Levin , Bardsley et al., Cho et al. , and Li et al. . Such segmented regions, however, should be small, to make the constant PSF assumption valid for the spatially varying motion blur in camera shake motion. For example, Tai et al. extended the hybrid camera framework used by Ben-Ezra and Nayar to estimate a PSF per pixel using an auxiliary video camera. This need for a per-pixel PSF revealed the futility of relying on the conventional kernel-based PSF model for spatially varying blur due to ego motion.

3. RICHARDSON LUCY DECONVOLUTION ALGORITHM (Non-blind)

The reason for the popularity of the Richardson Lucy algorithm is its implementation of maximum likelihood and its apperent ability to produce reconstructed image of good quality in the presence of high noise levels.The Richardson-Lucy algorithm generates a restored image through an iterative method. The essence of the iteration is as follows: the (n+1)th estimate of the restored image is given by the nth estimate of the restored image multiplied by a correction image. That is,

$$\text{image}_{n+1} = \text{image}_n \; \frac{\text{original data}}{\text{image}_n * \text{PSF}} \bullet \text{PSF}$$

In formal way one can express the output of the continuous linear system in the form of probabilities:

$$y(t) = \int h(t|\tau)\, x(\tau)\, d\tau$$

Where $x(\tau)$ is the unblurred image

$h(t|\tau)$ is the PSF (the fraction of light coming from true location τ that get scattered into observed pixel t)

And y(t) is the blurry image.

from this one can write

$$x(\tau) = \int q(\tau|t)\, y(t)\, dt$$

where $q(\tau|t)$ is related to $h(t|\tau)$ via Bayes theorem

$$q(\tau|t) = x(\tau)\frac{h(t|\tau)}{y(t)}$$

$q(\tau|t)$ is calculated using an iteration approach

$$q^{(n)}(\tau|t) = x^{(n)}(\tau)\frac{h(t|\tau)}{y^{(n)}(t)}$$

and

$$x^{(n)}(\tau) = \int q^{(n-1)}(\tau|t)\, y^{(0)}(t)\, dt = \int x^{(n-1)}(\tau)\frac{h(t|\tau)}{y^{(n-1)}(t)}\, y^{(0)}(t)\, dt$$

with $y^{(0)}(t)$ being the measured data $y(t)$ and

$$y^{(n-1)}(t) = \int h(t|\tau)\, x^{(n-1)}(\tau)\, d\tau$$

finally in discrete form we have

$$x^{(n)}(i) = x^{(n-1)}(i)\sum_{j=0}^{N-1} h(j,i)\frac{y(j)}{\sum_{k=0}^{M-1} h(j,k)\, x^{(n-1)}(k)},\quad i \in \langle 0, M-1 \rangle$$

The result of each iteration is used to make a prior for the next one,the Richardson-Lucy iteration converges to the maximum likelihood solution for Poisson statistics in the data.

3.1Advantages Of RL algorithm

a) Iterative Richardson-Lucy algorithm gives an estimation $x^{(k)}$ of the true image x

b) The restored images are robust against small errors in the used point spread function.

c) Easy to implement (no non-linear optimisation is needed)

d) The R-L iteration converges to the maximum likelihood solution for Poisson statistics in the data (Shepp and Vardi 1982), which is appropriate for optical data with noise from counting statistics.

e) The R-L method forces the restored image to be non-negative and conserves flux both globally and locally at each iteration.

4. FLOW OF WORK

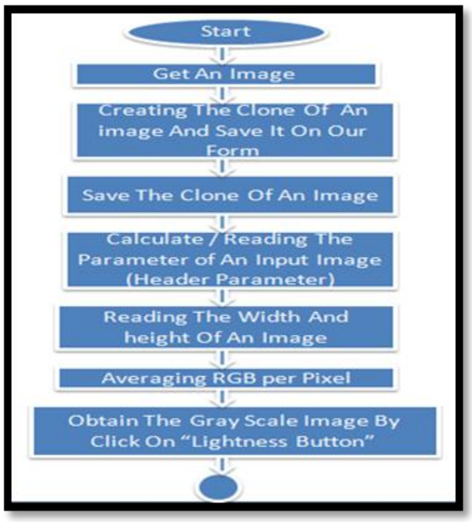

Start
Get An Image
Creating The Clone Of An image And Save It On Our Form
Save The Clone Of An Image
Calculate / Reading The Parameter of An Input Image (Header Parameter)
Reading The Width And height Of An Image
Averaging RGB per Pixel
Obtain The Gray Scale Image By Click On "Lightness Button"

Fig4(a). Flow Of work

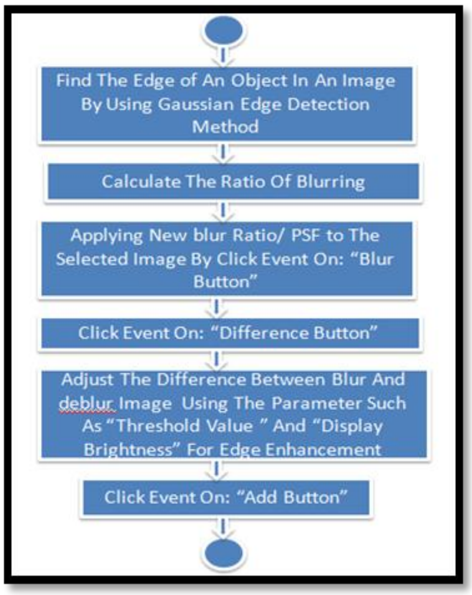

Find The Edge of An Object In An Image By Using Gaussian Edge Detection Method
Calculate The Ratio Of Blurring
Applying New blur Ratio/ PSF to The Selected Image By Click Event On: "Blur Button"
Click Event On: "Difference Button"
Adjust The Difference Between Blur And deblur Image Using The Parameter Such As "Threshold Value " And "Display Brightness" For Edge Enhancement
Click Event On: "Add Button"

Fig4(b). Flow Of Work

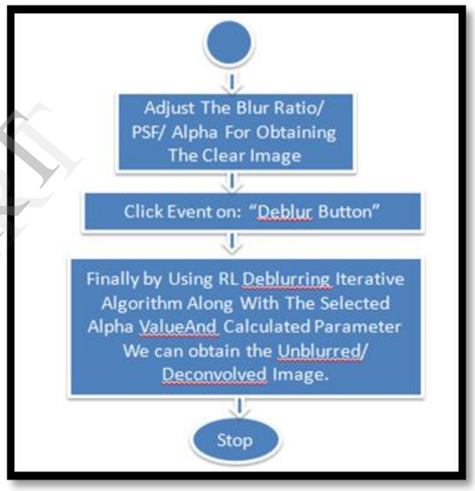

Adjust The Blur Ratio/ PSF/ Alpha For Obtaining The Clear Image
Click Event on: "Deblur Button"
Finally by Using RL Deblurring Iterative Algorithm Along With The Selected Alpha ValueAnd Calculated Parameter We can obtain the Unblurred/ Deconvolved Image.
Stop

Fig4(c). Flow Of Work

5. IMPLEMENTATION WORK

Step 1: Load a clear image

From fig 3 our first step is loading of clear image. The basic objective of step 1 is: An image must come on our "Form" in any case from Hard-disk and while coming it must come with the whole properties of image such as size of image , name of image, pixel size, size of histogram, location of image etc…

Methodology to achieve step 1 follows the following steps:

1) Creation Of Form

2) File open, file save, RGB, lightness, blur, diff, add, deblur button Event Creation.

3) Open Dialog Box
4) Dialog box showing filters with selected images
5) Loading the Clone of selected image on our form

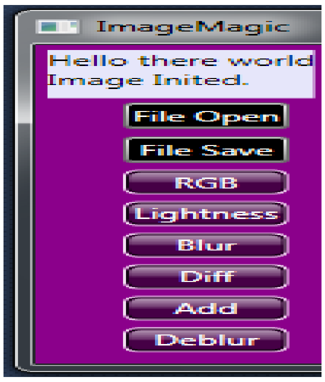


Fig5. Creation of form “Image Magic” and file open, file save, RGB, lightness, blur, diff, add, deblur button event creation.

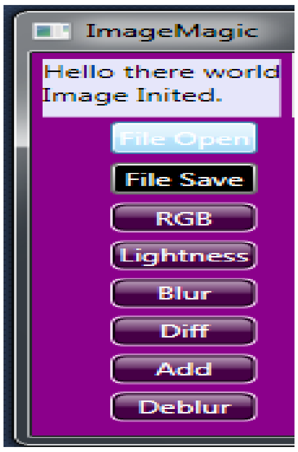


Fig6. Click event on: File Open button

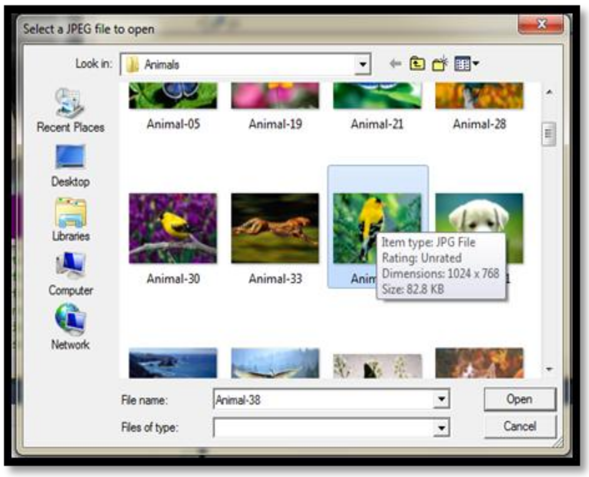


Fig7. Selecting the image from open dialogue box, which is showing filters such as jpg., fpg., bmp., etc (file of type) along with the selected image and its properties.

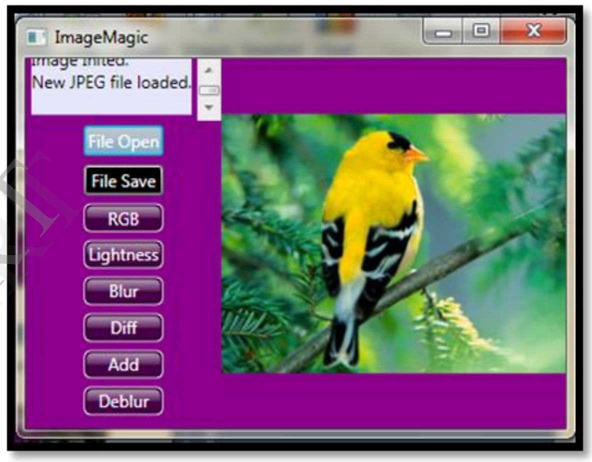


Fig8. Loaded clone of selected image on our form.

Step2: Blur the loaded clear image.

From fig3 our next step is to obtain the blurred image from the loaded clone of image, which we have obtained from step1. Step 2 follows the following steps for obtaining the blur image:

1) Save the clone of image.
2) Click event on “File save button”
3) Click event on “Lightness button”

By clicking on lightness button we can obtain grays scale image. Here we require the gray scale image because in image processing whenever there is a need of prominent edge detection the RGB image has to be converted into gray scale image so that the edge can be detected with accuracy and the it is converted into editable mode using BMS engine.

4) Click event on “Blur button”

For obtaining the blurry image or the synthetic image there are so many techniques some of them are: linear blurring technique, in which horizontal and vertical fixed number of pixel. Block blurring technique where averaging a small block of pixels by propagating a fixed sized window through the entire image. Gaussian blurring technique where the convolution of image with two dimensional Gaussian functions has taken place. But in our project we are using linear blurring technique because of its advantages as follows:

- A linear blur helps us to add motion to our images. It blurs all of our image or a portion of our image we select.
- A linear blur is a blur that goes in a straight line. It can originate in any direction, running vertically, horizontally or at an angle.
- Linear blur shouldn't be confused with linear sampling. Linear sampling is a type of Gaussian blur, not a true linear blur. A Gaussian blur creates soft edges in an image, making it appear blurry.

5) Obtaining the "blur port".
6) Obtaining the blurred image or synthetic image on our form.

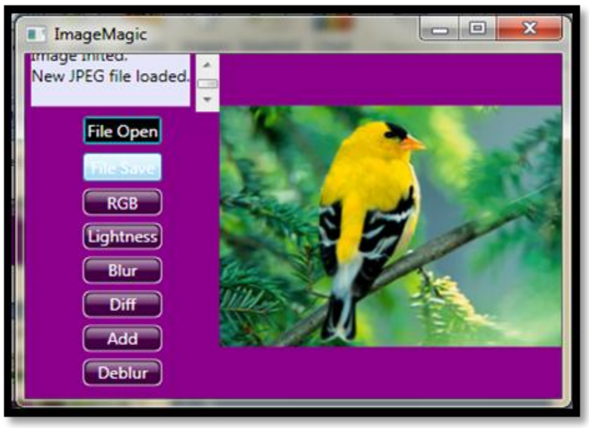


Fig9.Click event on "File Save button"

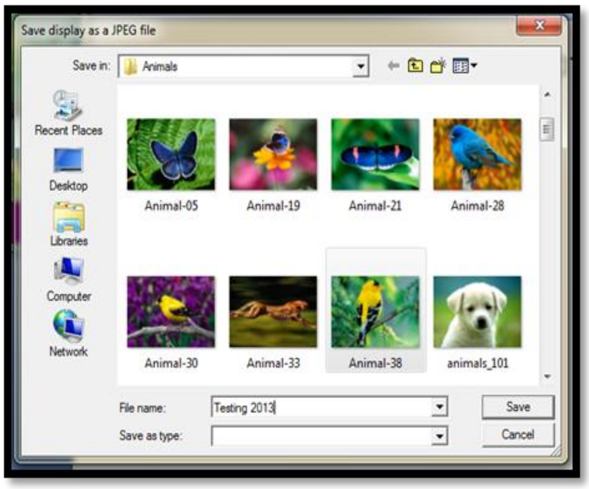


Fig10. Save the selected file with another name (here e.g. Testing 2013)

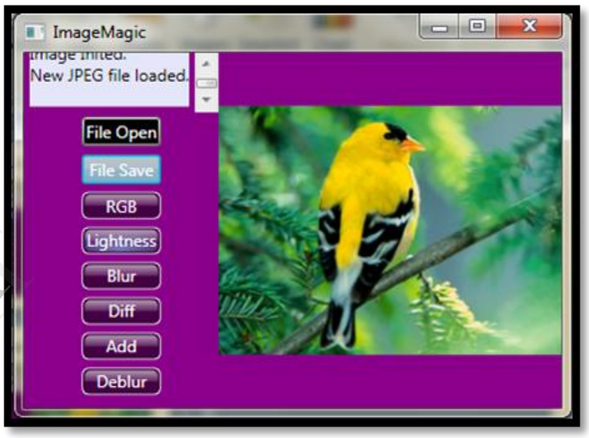


Fig11 . Click event on "lightness button"

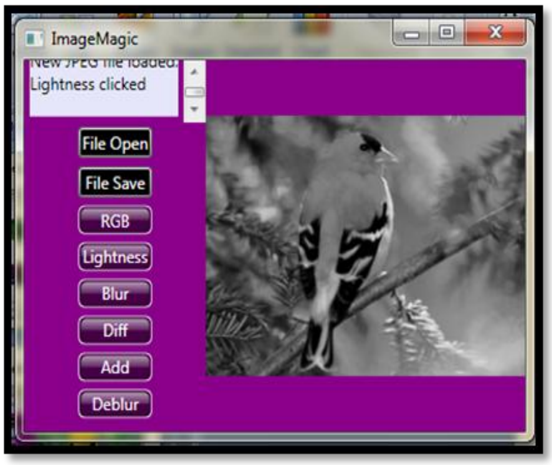


Fig12. Obtained gray scale image from loaded clear image.

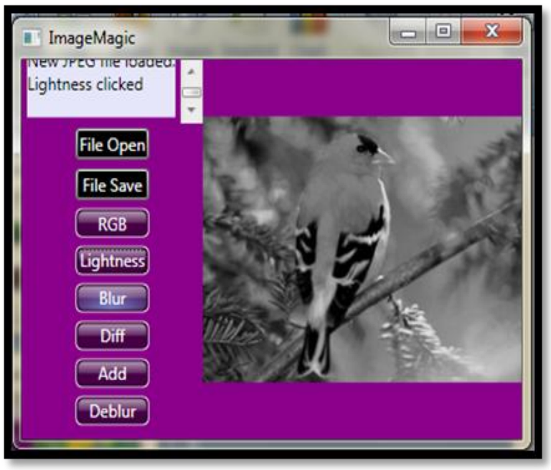


Fig13. Click event on "blur button"

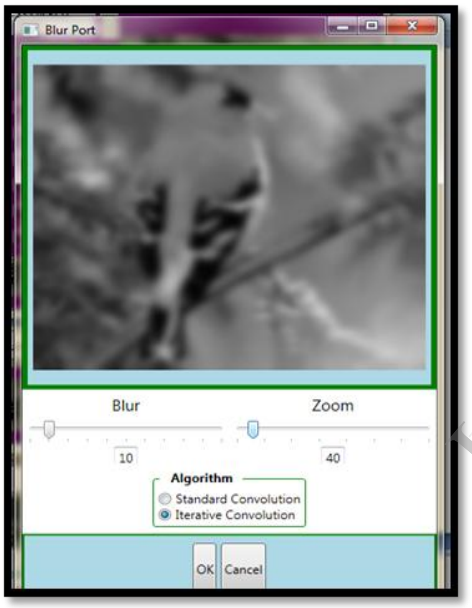


Fig 14. Obtaining the bur port where we can adjust the blurring ratio (here e.g. blur ratio=10) also we can see the image by zooming it.

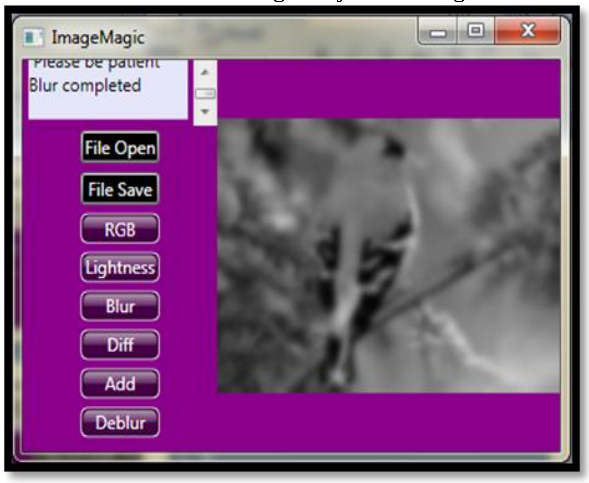


Fig15. Obtained blurred image or synthetic image.

Step 3: Obtaining the clear image by using Richardson Lucy Deblurring algorithm.

From fig3. Our next step is to obtained a deblurred image from the blurry image, which governs the following steps:

1) Click event on "diff button"
2) Obtaining the difference port
3) Obtaining the resultant image of differnce port on our form.
4) Click event on "add button"
5) Obtaining the "add port"
6) Obtaing the resultant image of add port on our form.
7) Click event on "deblurr button" (here we are using Richardson Lucy Deblurring algorithm (non blind) as we have mentioned in this paper)
8) Obtaining the resultant deblurred/ deconvolved image.

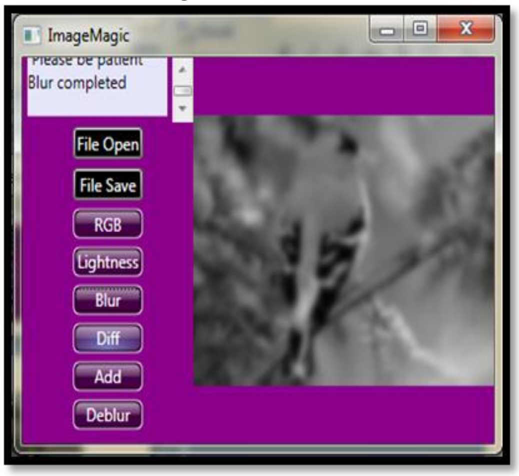
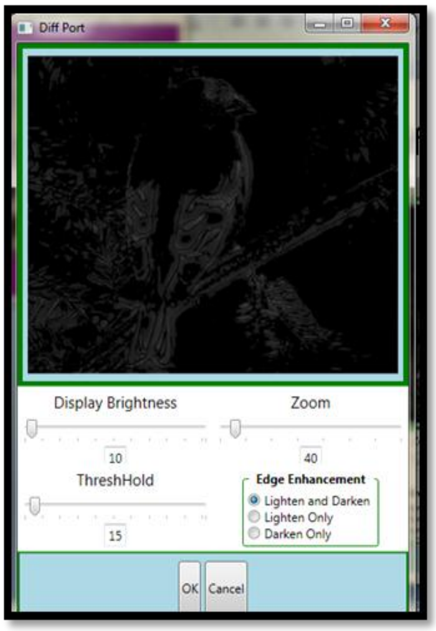
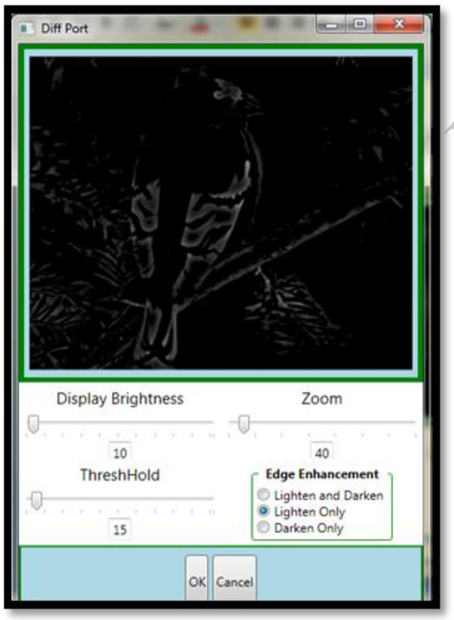


Fig16. Click event on difference button



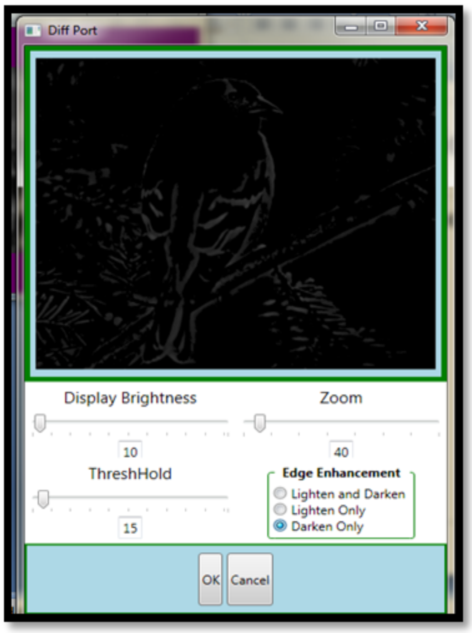
Display Brightness: 10
Zoom: 40
ThreshHold: 15
Edge Enhancement: [✓ Lighten and Darken] Lighten Only, Darken Only

Fig17(a). Obtaining the "diff port" (here Display Brightness = 10 Threshold = 15 Edge Enhancement= Lighten and Darken)



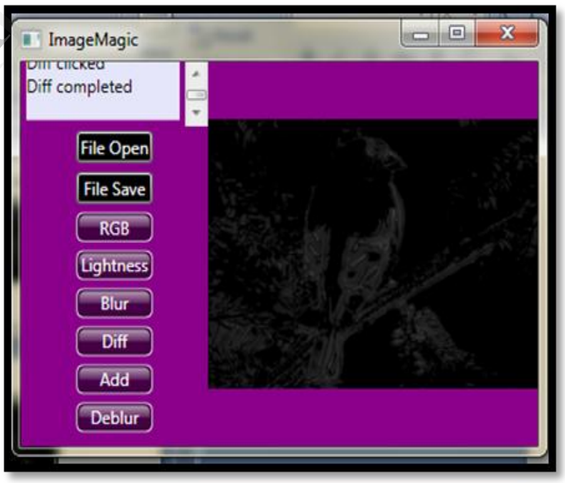
Display Brightness: 10
Zoom: 40
ThreshHold: 15
Edge Enhancement: Lighten and Darken, [✓ Lighten Only] Darken Only

Fig17(b).Obtaining the "diff port"(here Display Brightness = 10 Threshold = 15 Edge Enhancement= Lighten only)



Display Brightness: 10
Zoom: 40
ThreshHold: 15
Edge Enhancement: Lighten and Darken, Lighten Only, [✓ Darken Only]

Fig17(c). Obtaining the "blur port"(here Display Brightness = 10 Threshold = 15 Edge Enhancement= Darken only)



Diff clicked
Diff completed

Fig18. Obtaining the resultant image of "diff port" (here the resultant image is having edge enhancement

of lighten and darken both).

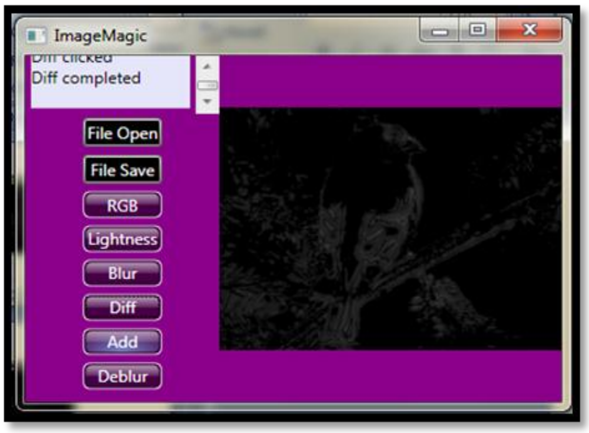


Fig19. Click event on "Add button"

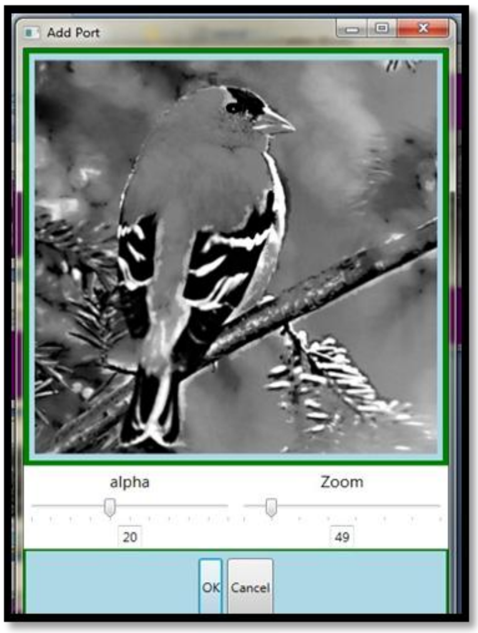


Fig20. Obtaining the add port where we can adjust the value of alpha (e.g. non blind deconvolution where the value of PSF is known here PSF is nothing but alpha) (here value of alpha =20)

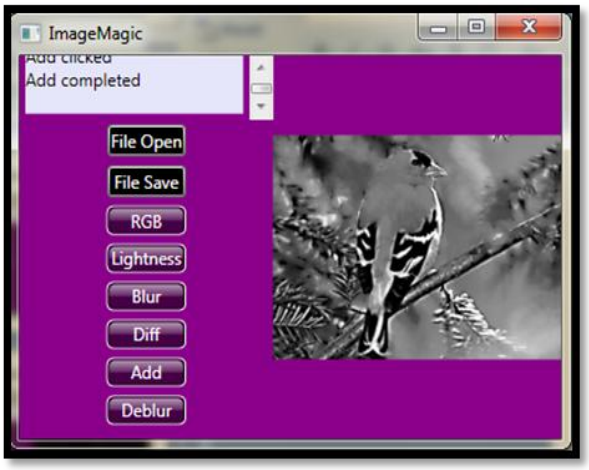


Fig21. Obtaining the resultant image of add port.

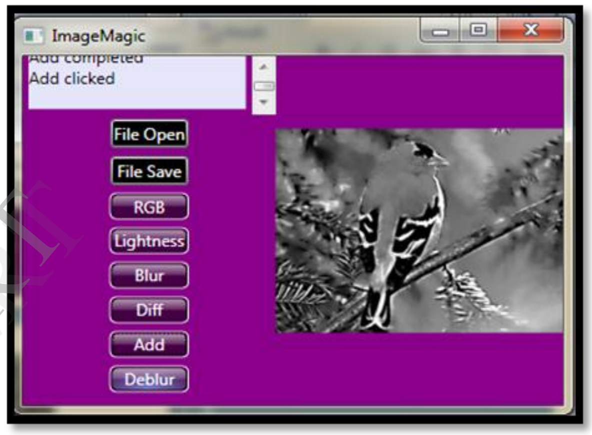


Fig22. Click event on "deblurr button"

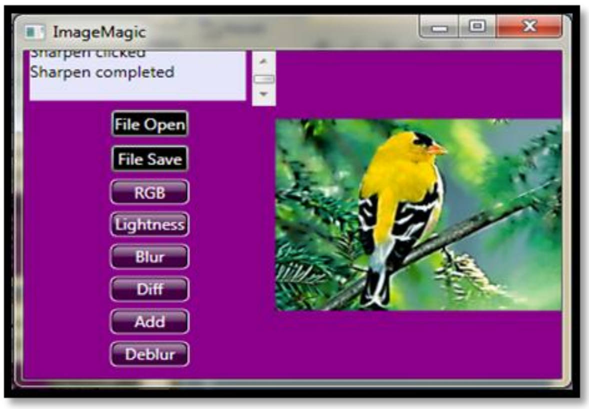


Fig23. Obtaining the resultant deblurred/ deconvolved image.

Step 4: Damping

So from fig3 our last step is damping that is comparison initially loaded image on our form and the finally obtained resultant image. In fig24 we can see the damping.

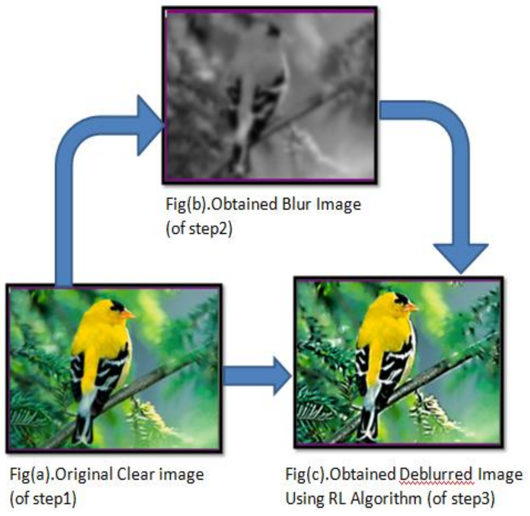

Fig(b).Obtained Blur Image
(of step2)
Fig(a).Original Clear image
(of step1)
Fig(c).Obtained Deblurred Image
Using RL Algorithm (of step3)

Fig24. Damping process

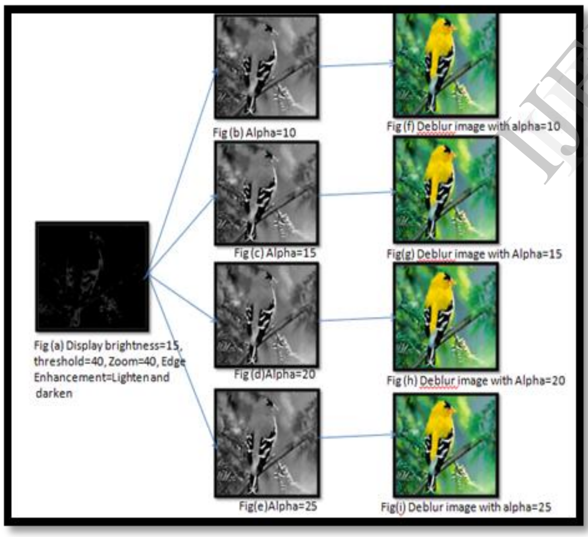


Fig (b) Alpha=10
Fig (f) Deblur image with alpha=10
Fig (c) Alpha=15
Fig(g) Deblur image with Alpha=15
Fig (a) Display brightness=15, threshold=40, Zoom=40, Edge Enhancement=Lighten and darken
Fig (d)Alpha=20
Fig (h) Deblur image with Alpha=20
Fig(e)Alpha=25
Fig(i) Deblur image with alpha=25

Fig25. Experiment with different alpha values with edge enhancement as lighten and darken only.

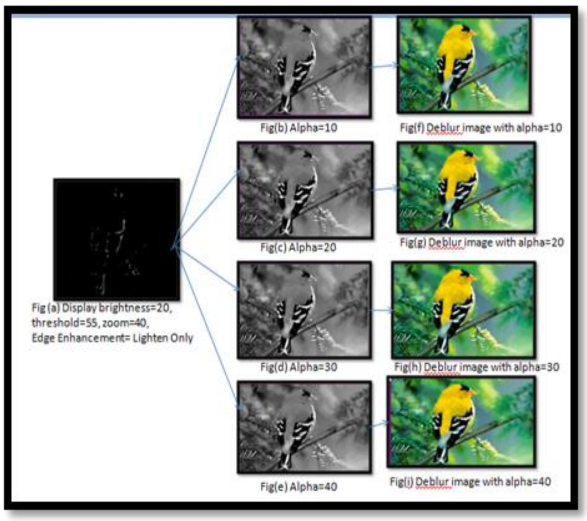

Fig(b) Alpha=10
Fig(f) Deblur image with alpha=10
Fig(c) Alpha=20
Fig(g) Deblur image with alpha=20
Fig (a) Display brightness=20, threshold=55, zoom=40, Edge Enhancement= Lighten Only
Fig(d) Alpha=30
Fig(h) Deblur image with alpha=30
Fig(e) Alpha=40
Fig(i) Deblur image with alpha=40

Fig26. Experiment with different alpha values with edge enhancement as lighten only.

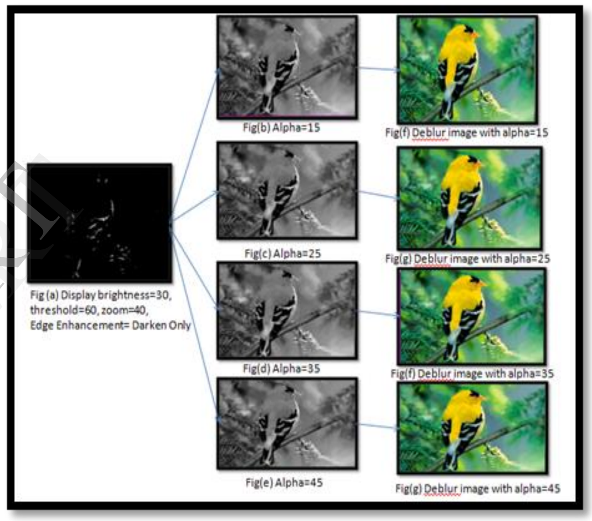

Fig(b) Alpha=15
Fig(f) Deblur image with alpha=15
Fig(c) Alpha=25
Fig(g) Deblur image with alpha=25
Fig (a) Display brightness=30, threshold=60, zoom=40, Edge Enhancement= Darken Only
Fig(d) Alpha=35
Fig(f) Deblur image with alpha=35
Fig(e) Alpha=45
Fig(g) Deblur image with alpha=45

Fig27. Experiment with different alpha values with edge enhancement as darken only.

6. CONCLUSION

We introduced a powerful framework for space-variant non blind-deconvolution technique. Our current implementation takes less than 1minute to obtain the clear image from the input blurry image. The running time of our algorithm depends on several factors, including image size , number of discrete sampling N, and the number of iterations. We have used dotnet framework 4.0 (version 2010) to obtain these results.

7. FUTURE DIRECTIONS

There are several future directions of this work. One is to explore other existing algorithms and hardware (e.g., coded exposure and coded aperture) that can use projective motion path blur formulation. Another important direction is to consider how our framework can be used to perform blind-deconvolution, where the camera's motion path is unknown. One can also increase the size of loaded image, since in our experiment we can take input image which is not more than 85 kb.

8. REFERENCES

1) T.F. Chan and C.K. Wong, "Total Variation Blind Deconvolutiontion," IEEE Trans. Image Processing, vol. 7, no. 3, pp. 370-375, Mar1998

2) Yu-Wing Tai, Member, IEEE, Ping Tan, Member, IEEE, and Michael S. Brown, Member, IEEE, "Richardson-Lucy Deblurring for Scenes under a Projective Motion Path" IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 33, NO. 8, AUGUST 2011.

3) G. R. Ayers and J. C. Dainty, ªIterative blind deconvolution method and its applications,º Opt. Lett. 13, 547±549 (1988).

4) J. R. Feinup, ªPhase retrieval algorithms: a comparison,º Appl. Opt. 21, 2758±2769 (1982).

5) L.A. Shepp and Y. Vardi, "Maximum Likelihood Reconstruction for Emission Tomography," IEEE Trans. Medical Imaging, vol. 1, no. 2, pp. 113-122, Oct. 1982.

6) Y. Vardi, Nonlinear Programming. Prentice-Hall, 1969.

7) A.A. Sawchuk, "Space-Variant Image Restoration by Coordinate Transformations," J. Optical Soc. Am., vol. 64, no. 2, pp. 138-144, 1974.

8) W. H. Richardson, ªBayesian-based iterative method of image restoration,º J. Opt. Soc. Am. 62, 55±59 (1972).

9) L. B. Lucy, ªAn iterative technique for the rectification of observed distributions,º Astron. J. 79, 745±754 (1974).

10) Lucy L.B., A.J. 79 (1974) 745.

11) Richardson W.H., J. Opt. Soc. Am. 62 (1972) 55.

12) T. J. Holmes, ªBlind deconvolution of quantum-limited incoherent imagery: maximum likelihood approach,º J. Opt. Soc. Am. A 9, 1052±1061 (1992).

Books

- Dot Net Programming Anetshel By Aslibilgin and A. Sybex
- Mastering Dot Net By Evangelos
- Digital Image Processing By Rafael C. Gonzalez And Richard E. Wood