RESTORATION OF FAST MOVING OBJECTS

A Project report submitted in partial fulfilment of the requirements for

The award of the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

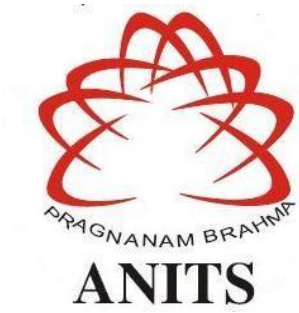
B.MADHUBABU (317126512129) P.SONIA (317126512157)

CH.SANDEEPTHI (317126512130) N.NARAYANA SWAMY (318126512L26)

Under the guidance of

Ms.P.DEVI **M.Tech.,(Ph.D.)**

Assistant Professor



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES
(UGC AUTONOMOUS)

(Permanently Affiliated to AU, Approved by AICTE and Accredited by NBA & NAAC with 'A' Grade)

Sangivalasa, bheemilimandal, visakhapatnam dist.(A.P)2020-2021

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES
(UGC AUTONOMOUS)
(Permanently Affiliated to AU, Approved by AICTE and Accredited by NBA & NAACwith 'A' Grade)
Sangivalasa, bheemilimandal, visakhapatnamdist.(A.P)

PRAGNANAM BRAHMA

ANITS

CERTIFICATE

This is to certify that the project report entitled **"RESTORATION OF FAST MOVING OBJECTS**submittedby**B.Madhubabu(317126512129),CH.Sandeepthi(317126512130),P.Sonia(317126512157) and N.Narayana Swamy (318126512L26)** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Electronics & Communication Engineering** of Andhra University, Visakhapatnam is a record of bonafide work carried out under my guidance and supervision.

Project Guide

Ms.P.DEVI
M.Tech(Ph.D.)
Department of E.C.E
ANITS

Assistant Professor
Department of E.C.E.
Anil Neerukonda
Institute of Technology & Sciences
Sangivalasa Visakhapatnam-531 162

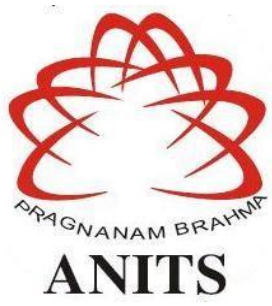
Head of the Department

Dr. V.Rajyalakshmi
M.E., Ph.D., MISTE. MIEEE, FIETE
Department of E.C.E
ANITS

Head of the Department
Department of E C E
Anil Neerukonda Institute of Technology & Sciences
Sangivalasa - 531 162

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES
(UGC AUTONOMOUS)
(Permanently Affiliated to AU, Approved by AICTE and Accredited by NBA & NAACwith 'A' Grade)
Sangivalasa, bheemilimandal, visakhapatnamdist.(A.P)



CERTIFICATE

This is to certify that the project report entitled **"RESTORATION OF FAST MOVING OBJECTS**submittedby**B.Madhubabu(317126512129),CH.Sandeepthi(317126512130),P.Sonia(317126512157) and N.Narayana Swamy (318126512L26)** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Electronics & Communication Engineering** of Andhra University, Visakhapatnam is a record of bonafide work carried out under my guidance and supervision.

Project Guide

Ms.P.DEVI
M.Tech(Ph.D.)
Department of E.C.E
ANITS

Head of the Department

Dr. V.Rajyalakshmi
M.E., Ph.D., MISTE. MIEEE, FIETE
Department of E.C.E
ANITS

CONTENTS

ABSTRACT

The object that moves over a distance exceeding its size within the exposure of time is Fast Moving objects (FMO). The main aim of the project is to restore the moving object. The main motivation behind choosing this project is, we all experience blur and the problem of blur is very serious in some applications. For instance, if we take CC TV footage and if we want to extract images from video recorded in it, it may not be clear, similarly if we click pictures in sports stadiums, medical scanning and astronomical applications blur is serious problem, so we would like to work on conversion of clear image from degraded or blur image.

For moving objects, Point Spread Function is not known. For this type of applications Blind deconvolution method gives a better result. In this project, 2D object restoration is performed by using Blind Deconvolution Method. In this method, PSF function of blurred object is determined and Weiner Filter gives the accurate results comparing to the other filters. For CLS filter the values of PSNR =12.6027 dB, RMS=31.8594 .The accuracy is measured in terms of RMS(Root Mean Square), PSNR(Peak Signal to Noise Ratio) and SSIM(Structural Similarity Index of metrics) values of different filter. Unlike 2D image restoration, in 3D image restoration the accuracy is measured in terms of RMS value. The restoration of 3D object is performed by using 3d Rotation method.

LIST OF FIGURES

CHAPTER 1
IMAGE RESTORATION

1.1 FAST MOVING OBJECTS:

The notion of a Fast Moving Object (FMO), i.e. an object that moves over a distance exceeding its size within the exposure time, is introduced. FMOs may, and typically do, rotate with high angular speed. FMOs are very common in sports videos, but are not rare elsewhere. In a single frame, such objects are often barely visible and appear as semi-transparent streaks. Fast Moving Objects (FMO) means the object that is moving over a certain distance with certain speed with respect to time exceeding its size .FMOs are invisible to human eye. While capturing an image of an FMO it will undergo motion blur. That blur may be due to many reasons like motion of the object, blur, noise, mis-focus etc.,

Fast moving object is considered as the one which could not easily be captured by conventional cameras in real time. The typical examples encompass fast moving cars, flying rockets, bouncing Ping-Pong balls, tennis balls, balancing pencils etc. It is impossible to recognize such moving objects without using a suitable algorithm and effective software system which are capable to learn and recognize patterns from complex Spatio- and Spectro-Temporal Data (SSTD). Deep learning has improved machine learning in computer vision from end to end. In this paper, we propose a new methodology for deep learning of video data and for accurate classification of moving objects captured in the data using e SNN (evolving Spike Neural Network).

Taking video footage encapsulating moving objects as input data, we conduct convolution operations for each video frame by using a Gaussian filter as the first step of deep learning, then adaptive down sampling is used to shrink the video frame both in width and height, after that spike encoding of these features is applied over time to identify the changes of each image block. Finally, we use the spikes of each 10 ×10 block of video frames as features and import them into Neu Cube for training and testing using dynamic evolving spiking neural network as a classifier to classify movement of the objects. Compared to other deep neural networks and other machine learning techniques, our NeuCube model has outperformed in various scenes for fast moving object recognition using spatio- and spectro- temporal video data, achieving accuracy of about 90%. It can be used for both high and low-resolution videos. Moreover, it allows to be further trained on new data in an on-line and incremental mode.

1.2 IMAGE RESTORATION

Image Restoration is the process of reconstructing or recovering an image that has been degraded by some degradation phenomenon. Restoration techniques are primarily modeling of the degradation and applying the inverse process in order to recover the original image. Image restoration techniques exist both in spatial and frequency domain. Image restoration is the method of conversion of a degraded/noisy image into original image. Degradation may occur due to many reasons like blur , noise and mis-focus of the camera. Image restoration is the reverse process of image degradation and it is performed by assuming a point source and use the point source image that is called the Point Spread Function (PSF), we use PSF to restore the loss due to the degradation process. The main aim of image restoration method is to reduce noise and recover resolution loss.

A MODEL OF IMAGE DEGRADATION

An image may be described as a two-dimensional function I

$$I = f(x, y) \qquad \text{eq.(1.1)}$$

Where x and y are spatial coordinates. Amplitude of f at any pair of coordinates (x, y) is called intensity I or grey value of the image. When spatial coordinates and amplitude values are all finite, discrete quantities, the image is called digital image. If f (x, y) is the original image, h(x,y) is a degradation function and η(x, y) is the additive noise then the degraded image g(x,y) is given as

$$g(x, y) = f(x, y) * h(x, y) + \eta(x, y) \qquad \text{eq.(1.2)}$$

Where the symbol $*$ indicates spatial convolution. Since convolution in spatial domain is equal to multiplication in the frequency domain, the corresponding frequency domain representation is given below. Where the terms in capital letters are the Fourier transforms of the corresponding terms in equation.

$$G(u, v) = F(u, v)\, H(u, v) + N(u, v) \qquad \text{eq.(1.3)}$$

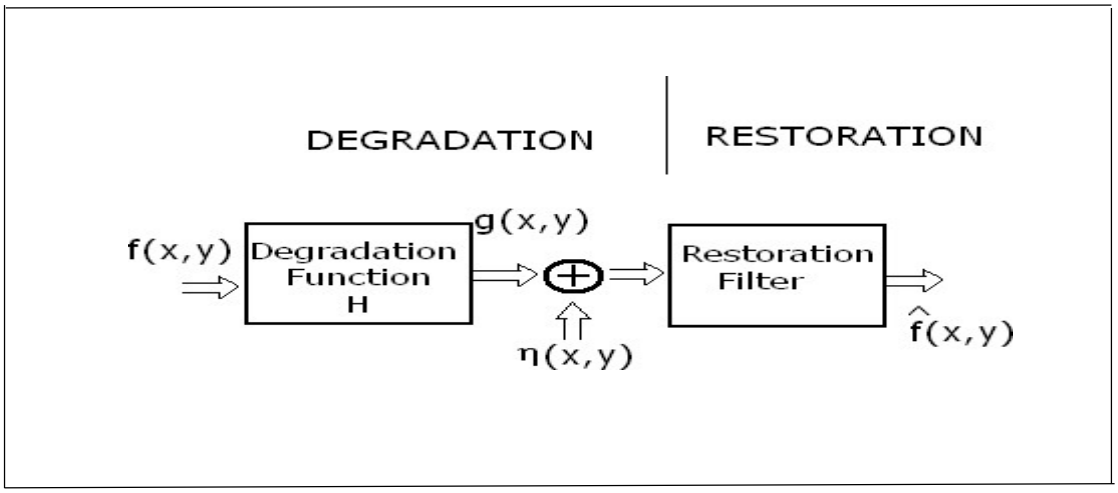


Fig.1.1 A model of the image degradation/restoration process.

Many types of the degradations can be approximated by linear, position invariant processes. Since degradations are modelled as being the result of convolution, and restoration seeks to find filters that apply the process in reverse, the term image deconvolution is used to signify linear image restoration.

1.3 POINT SPREAD FUNCTION

The linear position-invariant function h(x, y) in equation (1.2) is known as a point spread function. The point spread function gets convolved with the original image to give the degraded image. Some commonly occurring image degradations, which are linear and position-invariant are given below. It is the impulse response of a optical system. It is the response of an imaginary system to a point source. It represents a single point object. It is the optical transfer function in spatial domain. PSF has a wide range of applications in astronomical, medical imaging, electron microscopy. The point spread function (PSF)

describes the response of an imaging system to a point source or point object. A more general term for the PSF is a system's impulse response, the PSF being the impulse response of a focused optical system.

Motion blur

We often see images blurred because of camera movement during image capture. Suppose the relative motion is of velocity *v* at an angle θ with the horizontal axis and if T is the duration of exposure, then the blur length is L=vT.

Camera Defocus

Another commonly occurring blur is because of improperly focussed camera. Assuming the lens system is of circular aperture, with radius r the point spread function. Several reasons for camera defocus such as atmospheric disturbances, camera shaking, object in motion etc.,

1.4 DECONVOLUTION:

In this chapter we have seen that if we can estimate the point spread function, h(x, y) which caused the degradation, then we can get back the actual original image by deconvolution. But unfortunately, in many practical situations, the blur is often un- known, and little information is available about the true image. Therefore, the true image f (x, y) must be identified directly from g(x, y) by using partial or no information about the blurring process and the true image. Such an estimation problem, assuming the linear degradation model of equation (1.2) is called blind deconvolution. In mathematics, deconvolution is the operation inverse to convolution. Both operation are used in signal processing and image processing. The foundations are based upon a suite of methods that are designed to remove or reverse the blurring present in microscope images induced by the limited aperture of the object.

There are several motivating factors behind the use of blind deconvolution for image processing applications. In practice, it is often costly, dangerous, or physically impossible to obtain a priori information about the imaged scene. For example, in applications like remote sensing and astronomy, it is difficult to statistically model the original image or even know specific information about scenes never imaged before. In addition, the degradation from blurring cannot be accurately specified. In aerial imaging and astronomy, the blurring cannot be accurately modelled as a random process, since fluctuations in the PSF are difficult to characterize. In real-time image processing, such as medical video-conferencing, the parameters of the PSF cannot be pre-determined to instantaneously deconvolve images. Moreover, on-line identification techniques used to estimate the degradation may result in significant error, which makes the restored image useless. In other applications, the physical requirements for improved image quality are unrealizable.

For instance, in space exploration, the physical weight of a high resolution camera exceeds practical constraints. Similarly, in x-ray imaging, improved image quality occurs with increased incident x-ray beam intensity, which is hazardous to a patient's health. Thus, blurring is unavoidable. In such situations, the hardware available to measure the PSF of an imaging system is often difficult to use. Although these methods work well to

identify the PSF, they are esoteric, which limits their wide use. Blind deconvolution is a viable alternative.

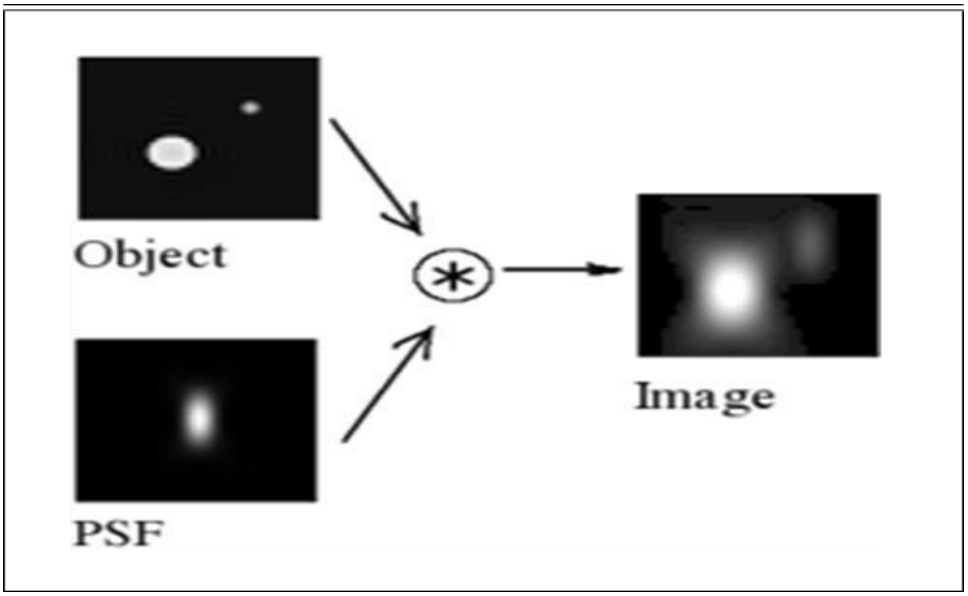


Figure 1.2: Convolution of the point spread function with a point object gives the observed image.

1.5 Filters

Image processing is based on filtering the content of images. Filtering is used to modify an image in some way. This could entail blurring, deblurring, locating certain features within an image, etc… Linear filtering is accomplished using convolution, as discussed above. A filter, or convolution kernel as it is also known, is basically an algorithm for modifying a pixel value, given the original value of the pixel and the values of the pixels surrounding it. There are literally hundreds of types of filters that are used in image processing. However, we will concentrate on several common ones. Low Pass Filters – The first filters we will talk about are low pass filters. These filters blur high frequency areas of images. This can sometimes be useful when attempting to remove unwanted noise from an image. However, these filters do not discriminate between noise and edges, so they tend to smooth out content that should not be smoothed out. Lets see some filters description.

1.5.1 WIENER FILTER:

It executes an optional trade-off between inverse filtering and noise smoothing. It removes the additive noise and inverts the blurring simultaneously. Weiner filter works 2 processes simultaneously: First, Inverse filtering part (high pass filtering) to remove blur. Second, Noise smoothing part (low pass filtering) to remove noise. The main aim using the wiener filter is to reduce mean square error value. The wiener filter is also known as the mean square error filter.

Formation of Degraded image:

{Original image[F(u,v)] + { degradation function[H(u,v)] + additive noise[N(u,v)] } = degradation image[G(u,v)]

Restored image:

$[R(u,v)]=G(u,v).H_{win}(u,v)$ eq.(1.4)

Weiner Filter Degradation function:

$H_{win}(u,v)=Sf(u,v)H^*(u,v)/\{||H(u,v)||2Sf(u,v) + Sn\,(u,v)\}$ eq.(1.5)

Divide the numerator & denominator with S_f (u,v) and by using

$H_{win}(u,v)=H^*(u,v)/\{|H(u,v)|2+S_n(u,v)/S_f(u,v)\}$ eq.(1.6)

Where:

Sn(u,v) - power spectrum of the noise process, obtained by taking the Fourier transform of the noise autocorrection.

Sf(u,v) - power spectrum of the signal process , obtained by taking Fourie Transform of the signal autocorrection.

1.5.2 CONSTRINED LEAST SQUARE FILTER:

The problem of having to know something about the degradation function H is common to all image restoration methods. However, the wiener filter presents on additional difficulty: the power spectra of the underground image & noise must be known. When we do not have info on the power spectra the wiener filter is not optimal for image restoration. It is possible to achieve excellent results using the approximation equation of the last method. However, a constant estimate of the ratio of the power spectra is not always a suitable solution. CLS filtering requires knowledge of only the mean & variance of the noise.

CLS filter is an extensive of wiener filter where the deconvolution does not require info of the noise. - The constrained approach tries to enforce a constraint to represent some degree of smoothness so the resultant image Is smooth & noise free.

Minimum of a criterion function, C, defined as:

$$C=\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[\nabla^2 f(\varkappa, y)]^2 \qquad \text{eq.(1.7)}$$

Where ∇ represents the Laplacian operation. The frequency domain solution to this optimization problem is given by the expression.

$F(u,v)=[H^*(u,v)/\{|H(u,v)|2+\gamma|P(u,v)|2\}].G(u,v)$ eq.(1.8)

In spatial domain representation, γ is a parameter that must be adjusted so that the constraint in above equation is satisfied and P(u,v) is the fourier transform of the function.

1.5.3 INVERSE FILTER:

The reverse process of restoring the image from its original image is called inverse filtering. It is the simplest approach to restore the original image once the degradation function is known.

Original image (f) *blurred kernel (h) = blurred image (g)

$G(x) = f(x) * h(x)$

$G(u) = F(u)H(u)$

$F(u)=G(u)/H(u)$

$R(u)=1/H(u)$

$F(u)=R(u)G(u)$

R(u) is inverse frequency filter

Inverse filtering is very sensitive to additive noise.

Direct inverse filtering is the simplest approach to restoration. In this method, an estimate of the Fourier transform of the image $\hat{F}(u, v)$ is computed by dividing the Fourier transform of the degraded image by the Fourier transform of the degradation function.

$$\hat{F}(u,v)=G(u,v)/H(u,v) \quad \text{eq.(1.9)}$$

This method works well when there is no additive noise in the degraded image. That is, when the degraded image is given by $g(x, y) = f(x, y) * h(x, y)$. But if noise gets added to the degraded image, then the result of direct inverse filtering is very poor. Equation gives the expression for G (u, v). Substituting for G (u, v) in the above equation, we get

$$\hat{F}(u,v)=[N(u,v)/H(u,v)]+F(u,v) \quad \text{eq.(1.91)}$$

The above equation shows that direct inverse filtering fails when additive noise is present in the degraded image. Because noise is random and so we cannot find the noise spectrum N (u, v).

1.5.4 TRUNCATED INVERSE FILTER:

It is desirable to restrict the inverse filter to be FIR, one of the simple methods to get this requirement is to truncate.

1.6 DESIGN METRICS:

1.6.1 SSIM:

The Structural Similarity Index (SSIM) is a perceptual metric that quantifies image quality degradation caused by processing such as data compression. It is a full reference metric that requires two images from the same image capture a reference image and a processed image.

In our work we took the degraded image as reference image and compared it with the processed image.

SSIM is a newer measurement tool that is designed based on three factors i.e. luminance, contrast, and structure to better suit the workings of the visual system. This 3-component SSIM model was proposed by Ran and Farvardin where an image is disintegrated into three important properties such as edge, texture and smooth region. The resulting metric is calculated as a weighted average of structural similarity for these three categories. The proposed weight measuring estimations are 0.5 for edges, 0.25 for texture and 0.25 for smooth regions. It can also be mentioned that a 1/0/0 weight measurement influences the results to be closer to the subjective ratings. This can be implied that, no textures or smooth regions rather edge regions play a dominant role in perception of image quality.

The SSIM has a maximum value of 1. The maximum value of 1 indicates that the two signals are perfectly structurally similar while a value of 0 indicates no structural similarity.so we can say that the ideal value lies between 0 and 1. How to calculate the SSIM value:

ssimval = ssim(A , ref) calculates the structural similarity (SSIM) index for grayscale image or volume A using ref as the reference image or volume.

[ssimval , ssimmap] = ssim(A , ref) also returns the local SSIM value for each pixel in A .

1.6.2 PSNR:

The term peak signal-to-noise ratio (PSNR) is an expression for the ratio between the maximum possible value (power) of a signal and the power of distorting noise that affects the quality of its representation. The PSNR computes the peak signal-to-noise ratio, in decibels, between two images. The higher the PSNR, the better the quality of the compressed, or reconstructed image. The mean-square error (MSE) and the peak signal-to-noise ratio (PSNR) are used to compare image compression quality.

The Peak signal-to-noise ratio is the most commonly used quality assessment technique to measure the quality of reconstruction of lossy image compression codecs. The signal is considered as the original data and the noise is the error yielded by the compression or distortion. The PSNR is the approximate estimation to human perception of reconstruction quality compared to the compression codecs. In a logarithmic scale. It is usually expressed in decibels (dB).Higher values of PSNR is better.

PSNR= 10 * log (Max^2/MSE) eq.(1.92)
PSNR= 20*log (Max)-10*log(MSE) eq.(1.93)

peaksnr = psnr(A , ref) calculates the peak signal-to-noise ratio (PSNR) for the image A (output image), with the image ref as the reference image(degraded image).This ratio is used as a quality measurement between the original and a compressed image. The higher the PSNR, the better the quality of the compressed, or reconstructed image.

1.6.3 MSE:

The mean-square error (MSE) and the peak signal-to-noise ratio (PSNR) are used to compare image compression quality. The MSE represents the cumulative squared error between the compressed and the original image, whereas PSNR represents a measure of the peak error. The lower the value of MSE, the lower the error. Mean square Error is another type of error measuring technique used very commonly to measure the differences between the predicted value by an estimator and the actual value. It evaluates the error magnitude. It is a perfect measure of accuracy which is used to perform the differences of forecasting errors from the different estimators for a definite variable.

Small value of MSE improves image quality and reduces the error. Cumulative difference between the compressed image and original image is MSE. Mean square error (MSE) -Mean Squared Error (MSE) is defined as the square of differences in the pixel values between the corresponding pixels of the two images.

The mean square error (MSE) of N * M size image is given by,

MSE= ΣM, N [I1 (m, n) – I2 (m, n)] 2 / (M*N) ; M &N - number of rows and columns in the input images.

CHAPTER 2

TYPES OF IMAGE RESTORATION

Image restoration techniques are of two types. They are Blind Deconvolution and Non-blind
Deconvolution. There is not much difference in both of them except about PSF. If the Point Spread
Function of the degraded image is known then it is easy to restore the image using Non-blind Deconvolution. But most of the times the degraded function or PSF of the degraded image is unknown. So we opt for Blind Deconvolution method as a core concept in our project. The most conventional technique for image restoration is deconvolution, which is performed in the frequency domain. After performing Fourier transform of image PSF and after recovering the resolution loss due to the degradation process we should go for deconvolution in frequency domain. Here PSF of the degraded image is unknown to us and hence we go for blind deconvolution method for determining the Point Spread Function and applying direct inversion to that PSF in order to find the original image. A non-blind restoration method estimates the desired image 'f' from the given degraded image 'g' and PSF 'h'.

2.1 Blind Deconvolution

If the deconvolution is performed without having prior knowledge of the degradation function, then it is called blind deconvolution. In blind deconvolution, both the degradation function and the true image are estimated from the degraded image characteristics. Partial information about the imaging system may also be utilised if available. Blind deconvolution is of great interest because in most of the practical cases, knowing the degradation function is not possible. For example, in applications like remote sensing and astronomy, it is difficult to statistically model the original image or even know specific information about scenes never imaged before.

If we can estimate the point spread function, h(x,y) which caused the degradation, then we can get back the original image by deconvolution. There are two classes of deconvolution techniques. Classical image restoration techniques and blind deconvolution techniques. In classical restoration, we need to have a prior knowledge of the PSF which caused the degradation. After estimating the point spread function, one of the three techniques [9] given below can be used for deconvolution.

Blind deconvolution techniques are used when we don’t have any prior knowledge of the degradation process.

There are two main approaches to blind deconvolution of images:

1. Identifying the PSF separately from the true image, in order to use it later with one of the known classical image restoration methods. Estimating the PSF and the true image are disjoint procedures. This approach leads to computationally simple algorithms.

2. Incorporating the identification procedure with the restoration algorithm. This merge involves simultaneously estimating the PSF and the true image, which leads to the development of more complex algorithms.

Classification of Blind Techniques

Blind deconvolution techniques can be further divided into several categories . Each of those categories contain several algorithms for blind deconvolution. Some of the algorithms are in frequency domain and some are in spatial domain. A few algorithms

make use of both frequency domain and spatial domain data. The five classes of algorithms are given below. A detailed classification of blind deconvolution techniques can be found in [12].

- A priori Blur Identification Method
- Cepstrum based motion blur identification
- Iterative Blind Deconvolution
- Non-iterative Blind Deconvolution

2.1.1 A Priori Blur Identification

A priori blur identification methods perform blind deconvolution by identifying the PSF prior to restoration. This general class of techniques makes assumptions on the characteristics of the PSF such as symmetry, and availability of a known parametric form of the blur. Popular parametric models include PSFs resulting from linear camera motion or an out-of-focus lens system. Based on these assumptions, an attempt is made to completely characterize the PSF using special features of the true/blurred image. Once the PSF has been completely identified, one of the classical restoration techniques is used to estimate the true image using deconvolution [10].

A priori blur identification techniques are the simplest class of blind deconvolution methods to implement and have low computational requirements. They are applicable to situations in which the true image is known to possess special features, or when the PSF is known to be of a special parametric form. For more general situations or when less information is available other deconvolution algorithms must be used.

2.1.2 Cepstrum based motion blur Identification

A method for identifying linear motion blur is to compute the two-dimensional cepstrum of the blurred image g(x, y) [11]. The cepstrum of g(x, y) is given by

$$C(g(x, y)) = F^{-1}(\log(|F(g(x, y))|). \quad \text{eq.(2.1)}$$

An important property of the cepstrum is that it is additive under convolution. Thus, ignoring noise, we have

$$C(g(x, y)) = C(f(x, y)) + C(h(x, y)) \quad \text{eq.(2.2)}$$

$C(h(x, y)) = F^{-1}(\log(|F(h(x, y))|)$ has large negative spikes at a distance L from the origin. By the additivity of the cepstrum, this negative peak is preserved in C(g(x, y)), also at a distance L from the origin. If the noise level of the blurred image is not too high, there will be two pronounced peaks in the cepstrum. To estimate the angle of motion blur, draw a straight line from the origin to the first negative peak. The angle of motion blur is approximated by the inverse tangent of the slope of this line.

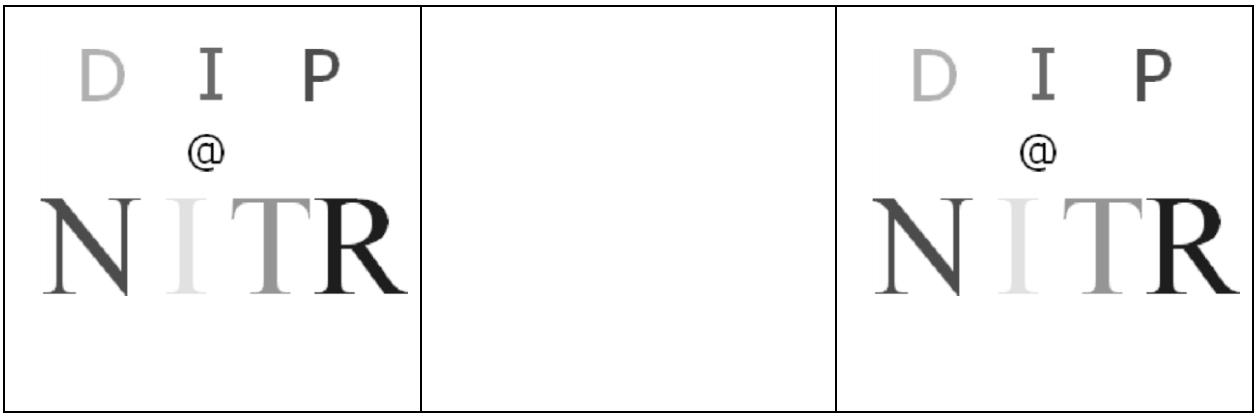


Figure 2.1. (a) The original image.

(b)The original image blurred with a motion blur of length 31pixels.

(c)Image restored using cepstral method.

Limitations of Cepstrum based Technique

From figure(1.4), it is evident that the cepstrum based method gives excellent results for horizontal motion blur. Similar results were obtained for vertical motion blur also[19]. The restoration is perfect since the length estimation was accurate. But the length estimate shows significant error when the motion blur direction is not horizontal or vertical. So we are going for iterative blind deconvolution techniques.

Coming to Blind Deconvolution method there are two types i.e., Iterative Blind Deconvolution and Non-Iterative Blind Deconvolution. Iteration means performing the same task again and again. So by iteration method each iteration improves the scene. Under iterative Blind Deconvolution there are several methods and algorithms to perform restoration process. They are

(a) Alternating Minimization (AM)

(b) Applying filters method

(c) Lucy Richardson algorithm.

- Coming to AM algorithm used to solve convex and non-convex problems[18]. In image processing, convex optimization plays a pivotal role in problems like image recovery and designing of various filters. While designing a filter wrt., frequency and time leads to convex optimisation problems and the other one is the image is perpendicular to camera's optical axis also leads to Convex optimisation problem .on the other hand if the limitation or restriction is not deals with convex then we can say that it is a non-convex problem.

- Next method is applying filters method. Restoration Filters are used in noise removal and in finding the original image. It consists inverse or reverse processes of blurring or degradation. There are several restoration filters out of those four filters we used in our project and compared the results with Structural similarity index metrics, root mean square error value and Peak signal to Noise ratio values. The filters used here are
    - Inverse filter
    - Wiener filter
    - Constrained least square filter
    - Truncated filter
- Next method is Lucy Richardson Algorithm. It is an iterative process for recovering a degraded image that has been blurred by a known point spread function. Once we find the PSF of an image by blind deconvolution method then we can apply this algorithm to find the inverse function of degraded function and thereby estimating the original image[17].

2.1.3 ITERATIVE BLIND DECONVOLUTION

In recent years, Image Deblurring techniques have played an essential role in the field of Image Processing. In image deblurring, there are several kinds of blurred image such as motion blur, defocused blur and Gaussian blur. Many methods to address this problem have been proposed by researchers in previous research, among which the iterative blind deconvolution (IBD) method is the most popular method to solve this problem. However, the convergence of this method is not ensured, and there is no effective method to choose a proper initial estimate image and PSF(point spread function). we improve the iterative blind deconvolution method by adding several constraints, which could be the type or parameters range of the PSF, on the PSF in each iteration. Experiment results validate that, with the help of these newly added constraints, our method is more likely to converge and has better deblurring performance than the IBD. The degraded image g(x, y) is represented by the convolution of the original image f (x, y) and the point spread function h(x, y).

$$g(x, y) = f(x, y) * h(x, y) \qquad \text{eq.(2.3)}$$

The equivalent equation in frequency domain is given by

$$G(x, y) = F(x, y)H(x, y) \qquad \text{eq.(2.4)}$$

Iterative blind deconvolution technique [3] uses some general a priori information concerning the original image f (x, y) and the PSF h(x, y). After a random initial guess is made for the PSF, the algorithm alternates between the image and Fourier domains, enforcing known constraints in each. The constraints are based on information available about the image and PSF .The basic deconvolution method consists of the following steps. First, a non-negative valued initial estimate of the PSF. This function is Fourier transformed to yield Hˆ(u, v)[16]. Which is then inverted to form an inverse filter and multiplied by G (u, v) to form a first estimate of the original image spectrum F (u, v). This estimated Fourier spectrum is inverse transformed to give f (x, y). The image-domain

constraint of non-negativity is now imposed by putting to zero all pixels of the image f (x, y) that have a negative value. A positive constrained estimate f̂(x, y) is formed that is Fourier transformed to give the spectrum F̂ (u, v). This is inverted to form another inverse filter and multiplied by G (u, v) to give the next spectrum estimate H(u, v). A single iterative loop is completed by inverse Fourier transforming H(u, v) to give h(x, y) and by constraining this to be nonnegative, yielding the next function estimate ĥ(x, y). The iterative loop is repeated until two positive functions with the

Formula: $c(x)=\int_{-\infty}^{+\infty} f(x_1)g(x-x)_1\,dx$ eq.(2.5)

The Fourier-domain constraint may be described as constraining the product of the Fourier spectra of f and h to be equal to the convolution spectrum, in agreement with equation [12]. It should be noted that, at the k^{th} iteration, two estimates for each Fourier spectrum are available. For example, $H_k(u, v)$ and the estimate $\hat{H}k(u, v)$ in the as shown in figure[13].

Both of these estimates have associated properties in common with the desired deconvolved solution. $\hat{H}k(u, v)$ has a nonnegative inverse transform, and the second estimate $H_k(u, v)$ obviously satisfies the Fourier-domain constraint. Therefore at each iteration the two estimates are averaged to form a composite new estimate as given in equation [14]. This averaging is not essential for convergence; however, the convergent rate is dependent on β, and a method of selecting the optimum value of β has not been found. Small, confined regions of low or zero value present in G(u, v) are dealt with by using only the estimate $\hat{H}k(u, v)$.

The estimate Hk(u, v) contributes no information to the new estimate. The Fourier-domain constraint can be summarized as follows. If we want to capture an image, then it has been motion blurred, that means the original image gets convoluted into a blurred image. So, if we want to remove the blur, then we have to deconvolute the motion blurred image to get the clear or restored image [15].

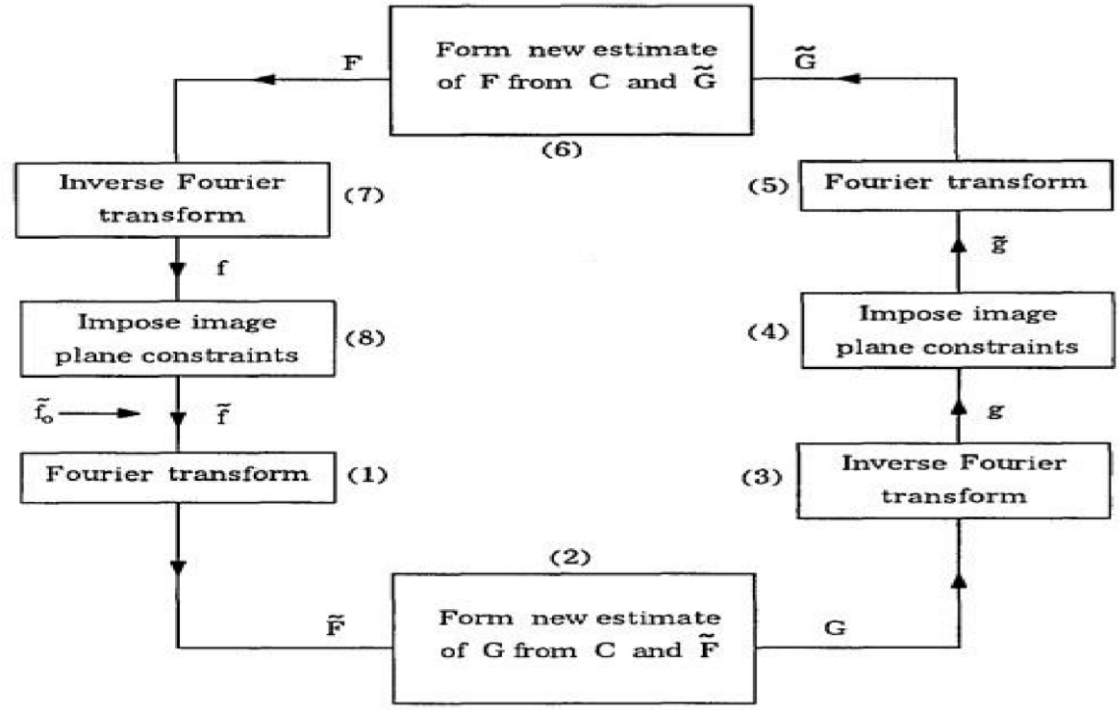


Fig.2.2. iterative blind deconvolution

The general formula: $F(x,y)*h(x,y)=g(x,y)$ eq. (2.6)

If we want to capture an image, then it has been motion blurred, that means the original image gets convoluted into a blurred image[34]. So, if we want to remove the blur, then we have to deconvolute the motion blurred image to get the clear or restored image.

Here in eq.1, we are giving output as a degraded image g(x,y), while we are giving the input image f(x,y) and its get convoluted with the degradation function i.e; PSF h(x,y).

In practical times we give the degraded image as input and we'll expect restored image as an output image.

Let f' be the restored image (because after restoring the degraded image, we may not get the same original image as the output.)

$$f'(x,y)*h(x,y)=g(x,y)$$

applying FFT on both sides

$$F'(u,v)H(u,v)=G(u,v)$$

Therefore, $F'(u,v)=G(u,v)/H(u,v)$

Applying IFFT to the above equation to get the restored image

$$f'(x,y)=g(x,y)/h(x,y)$$

Adding noise to the problem:

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

For higher frequencies F' (u,v) are amplified, due to that significantly noise gets amplified. And the resulted output was a salt-peppered noise image (inverse filter output in the project. where for certain frequencies H(u,v)=0 then F(u,v)=infinite, In this condition image is not recoverable. In order to supress the noise, we are going for filters.

2.1.4 Non-iterative blind deconvolution

Coming to Non-iterative blind deconvolution estimation of PSF is done using the outer or exterior information available from the degraded or blurred image. Examples of non-iterative techniques include SeDDaRA, the cepstrum transform and APEX.

2.2 Non-blind deconvolution

All the techniques described so far, require the knowledge of the exact degradation function. Then the restoration algorithm is applied to invert the degradation process. Those techniques are called the classical Restoration techniques. Classical restoration techniques need prior knowledge of the degradation process. The degradation can be estimated by one of the several techniques [9] given below. Non-blind deconvolution comes under Classical restoration technique.

- Estimation by image observation.
- Estimation by experimentation
- Estimation by modelling

In non-blind deconvolution the most used algorithm is the alternating direction method of multipliers (ADMM). This algorithm solves convex problems by fragmenting them into parts so that it is easy to solve each problematic part separately. ADMM is a type of algorithm used more often when it comes to non-blind deconvolution[11].

2.2.1 Alternating Direction Method of Multipliers (ADMM)

The alternating direction method of multipliers (ADMM) is an algorithm that solves convex optimization problems by breaking them into smaller pieces, each of which are then easier to handle. It has recently found wide application in a number of areas. The method was originally known as the method of multipliers, and was studied much in the 1970 and 1980s as a good alternative to penalty methods. It was first discussed by Magnus Hestenes,[1] and by Michael Powell in 1969.[2] The method was studied by R. Tyrrell Rockafellar in relation to Fenchel duality, particularly in relation to proximal-point methods, Moreau–Yosida regularization, and maximal monotone operators: These methods were used in structural optimization. The method was also studied by Dimitri Bertsimas, notably in his 1982 book,[3] together with extensions involving non quadratic regularization functions, such as entropic regularization, which gives rise to the "exponential method of multipliers," a method that handles inequality constraints with a twice differentiable augmented Lagrangian function.

2.3 NOISE MODELS

The spatial component of noise is based on the statistical behaviour of the intensity values. These may be considered as random variables, characterized by a probability

density function (PDF)[33]. Some commonly found noise models [9] and their corresponding PDFs are given below.

2.3.1 Gaussian Noise

Gaussian noise is noise that has a probability density function of the normal distribution (also known as Gaussian distribution). In other words, the values that the noise can take on are Gaussian distributed. If *z* is a Gaussian random variable representing noise, then its PDF is given by

$$p(z) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right) * e^{-(z-\mu)^2/2\sigma^2} \qquad \text{eq.(2.7)}$$

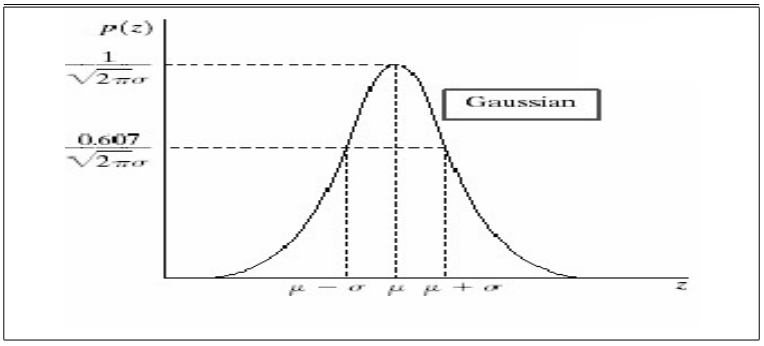


Fig.2.3. the probability density function if a Gaussian distribution system

Gaussian noise is additive noise. That is the Gaussian distributed noise values get added to the intensity values of the image.

Fig.2.4. Original image

Fig.2.5. Blurred image

2.3.2 Uniform Noise

Another commonly observed image noise is uniform noise. In this case the noise can take on values in an interval [a,b] with uniform probability[32]. The plot of the probability density

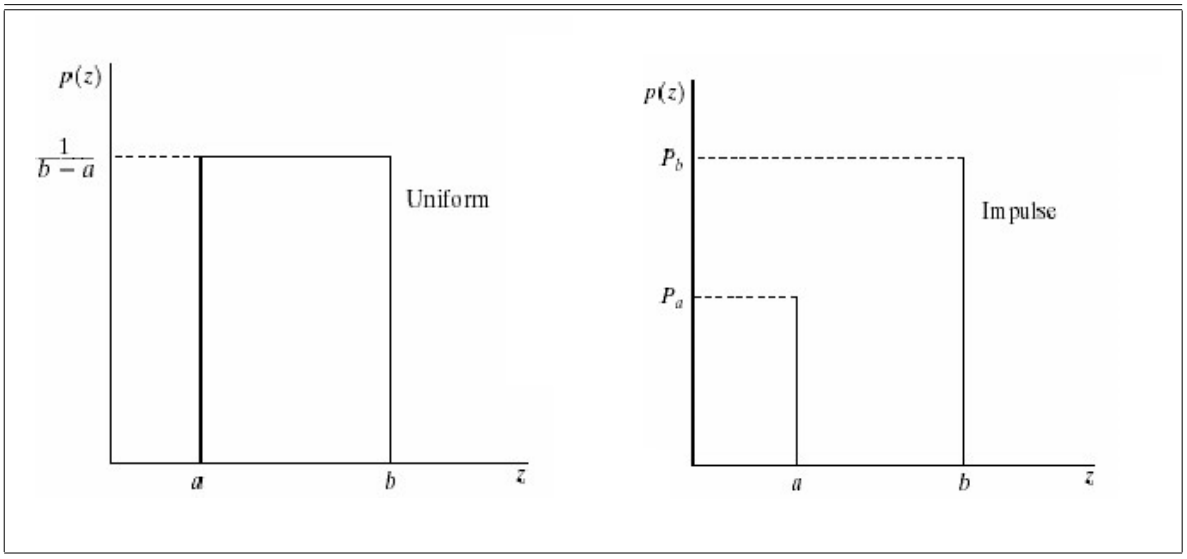


Fig 2.6: (a) Probability density function for Uniform Noise.
(b) Probability density function for salt and pepper noise

2.3.3 Impulse noise

Impulse noise is characterized by a noise spike replacing the actual pixel value. Impulse noise is further divided into two classes. Random valued impulse noise (RVIN) and salt & pepper noise (SPN)[31]. In RVIN, the impulse value at a particular pixel may be a random value between a particular intervals. But in SPN the impulses are either the minimum value or the maximum value allowed in the intensity values. For example 0 and 255 in the case of 8-bit image.

2.4 Denoising techniques

Another type of image degradation is due to noise. Random values get added to the intensity values. Denoising involves the application of some filtering technique to get the true image back. The denoising algorithms vary greatly depending on the type of noise present in the image. Each type of image is characterized by a unique noise model. Each noise model corresponds to a probability density function which describes the distribution of noise within the image. Denoising techniques exist in both spatial domain as well as frequency domain. When the only degradation present in an image is noise,

then equation (1.2) becomes

$$g(x, y) = f(x, y) + \eta(x, y) \qquad \text{eq.(2.8)}$$

and equation (1.3) becomes

$$G(u, v) = F(u, v) + N(u, v) \qquad \text{eq.(2.9)}$$

Denoising techniques exist in both spatial domain as well as frequency domain.

CHAPTER 3
MATLAB

3.1 Introduction

The most important things to learn are how to enter matrices, how to use the: (colon) operator, and how to invoke functions[26]. At the heart of MATLAB is a new language you must learn before you can fully exploit its power. You can learn the basics of MATLAB quickly, and mastery comes shortly after. You will be rewarded with high productivity, high-creativity computing power that will change the way you work.[25]

Describing the components of the MATLAB system

Development Environment - introduces the MATLAB development environment, including information about tools and the MATLAB desktop.

Manipulating Matrices - introduces how to use MATLAB to generate Matrices and perform mathematical operations on matrices.

Graphics - introduces MATLAB graphic capabilities, including information about plotting data, annotating graphs, and working with images.

Programming with MATLAB - describes how to use the MATLAB language to create scripts and functions, and manipulate data structures, such as cell arrays and multidimensional arrays.[23]

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation [24].

Typical uses include

- Math and computation
- Algorithm development
- Modelling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including graphical user interface building.

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar non interactive language such as C or FORTRAN. The name MATLAB stands for matrix laboratory. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects. MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instrumental tool for introductory and advanced courses in mathematics, engineering and science. In industry, MATLAB is the tool of choice for high-productivity research, development and analysis [22]. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others.

3.2 Matlab system

The MATLAB system consists of five main parts:

- **Development Environment**

This is the set of tools and facilities that help you use MATLAB functions and files. Many of these tools are graphical user interfaces. It includes the MATLAB desktop and Command Window, a command history, and browsers for viewing help, the workspace, files, and the search path[21].

- **The Matlab Mathematical Function Library**

This is a vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix eigenvalues, Bessel functions, and fast Fourier transforms.

- **The Matlab Language**

This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features[20]. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create complete large and complex application programs.

- **Handle Graphics**

This is the MATLAB graphics system. It includes high-level commands for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level commands that allow you to fully customize the appearance of graphics as well as to build complete graphical user interfaces on your MATLAB applications.

- **The Matlab Application Program Interface (Api)**

This is a library that allows you to write C and FORTRAN programs that interact with MATLAB. It include facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

- **Development environment**

This chapter provides a brief introduction to starting and quitting MATLAB, and the tools and functions that help you to work with MATLAB variables and files. For more information about the topics covered here, see the corresponding topics under Development Environment in the MATLAB documentation, which is available online as well as in print [27].

3.3 Starting and Quitting MATLAB

Starting Matlab

On a Microsoft Windows platform, to start MATLAB, double-click the MATLAB shortcut icon on your Windows desktop. On a UNIX platform, to start MATLAB, type matlab at the operating system prompt. After starting MATLAB, the MATLAB desktop opens - see MATLAB Desktop. You can change the directory in which MATLAB starts,

define start up options including running a script upon start up, and reduce start up time in some situations.

Quitting Matlab

To end your MATLAB session, select Exit MATLAB from the File menu in the desktop, or type quit in the Command Window. To execute specified functions each time MATLAB quits, such as saving the workspace, you can create and run a finish.m script.

Matlab Desktop

When you start MATLAB, the MATLAB desktop appears, containing tools (graphical user interfaces) for managing files, variables, and applications associated with MATLAB.

3.4 Image processing with MATLAB

The Command Window is the window on the right hand side of the screen. This window is used to both enter commands for MATLAB to execute, and to view the results of these commands. The Command History window, in the lower left side of the screen, displays the commands that have been recently entered into the Command Window. In the upper left hand side of the screen there is a window that can contain three different windows with tabs to select between them[28]. The first window is the Current Directory, which tells the user which M-files are currently in use. The second window is the Workspace window, which displays which variables are currently being used and how big they are. The third window is the Launch Pad window, which is especially important since it contains easy access to the available toolboxes, of which, Image Processing is one. If these three windows do not all appear as tabs below the window space, simply go to View and select the ones you want to appear[29]. In order to gain some familiarity with the Command Window, try Example 2.1, below. You must type code after the >> prompt and press return to receive a new prompt. If you write code that you do not want to reappear in the MATLAB Command Window, you must place a semi colon after the line of code. If there is no semi colon, then the code will print in the command window just under where you typed it.

Example 2.1

X = 1; %press enter to go to next line

Y = 1; %press enter to go to next line

Z = X + Y %press enter to receive result

As you probably noticed, MATLAB gave an answer of Z = 2 under the last line of typed code. If there had been a semi colon after the last statement, the answer would not have been printed[30]. Also, notice how the variables you used are listed in the Workspace Window and the commands you entered are listed in the Command History window. If you want to retype a command, an easy way to do this is to press the ↑ or ↓ arrows until you reach the command you want to reenter.

3.5 The M-file

M-file – An M-file is a MATLAB document the user creates to store the code they write for their specific application. Creating an M-file is highly recommended, although not entirely necessary. An M-file is useful because it saves the code the user has written for

their application. It can be manipulated and tested until it meets the user's specifications. The advantage of using an Mfile is that the user, after modifying their code, must only tell MATLAB to run the M-file, rather than renter each line of code individually. 3.2. Creating an M-file – To create an M-file, select File\New ►M-file.

Saving – The next step is to save the newly created M-file. In the M-file window, select File\Save As… Choose a location that suits your needs, such as a disk, the hard drive or the U drive. It is not recommended that you work from your disk or from the U drive, so before editing and testing your M-file you may want to move your file to the hard drive. Opening an M-file – To open up a previously designed M-file, simply open MATLAB in the same manner as described before. Then, open the M-file by going to File\Open…, and selecting your file. Then, in order for MATLAB to recognize where your M-file is stored, you must go to File\Set Path… This will open up a window that will enable you to tell MATLAB where your M-file is stored. Click the Add Folder… button, then browse to find the folder that your M-file is located in, and press OK. Then in the Set Path window, select Save, and then Close.

If you do not set the path, MATLAB may open a window saying your file is not in the current directory. In order to get by this, select the "Add directory to the top of the MATLAB path" button, and hit OK. This is essentially the same as setting the path, as described above. Writing Code – After creating and saving your M-file, the next step is to begin writing code. A suggested first move is to begin by writing comments at the top of the M-file with a description of what the code is for, who designed it, when it was created, and when it was last modified. Comments are declared by placing a % symbol before them. Comments appear in green in the M-file window. See Figure 3.1, below, for Example 3.1. Resaving – After writing code, you must save your work before you can run it. Save your code by going to File\Save. Running Code – To run code, simply go to the main MATLAB window and type the name of your M-file after the >> prompt. Other ways to run the M-file are to press F5 while the M-file window is open, select Debug\Run, or press the Run button (see Figure 3.1) in the M-file window toolbar.

M-file for Loading Images

The class of the new image is the same as that of the colour image. As you can see from the example M-file in Figure 4.1, MATLAB has the capability of loading many different image formats, two of which are shown. The function imread is used to read an image file with a specified format. Consult imread in MATLAB's help to find which formats are supported. The function imshow displays an image, while figure tells MATLAB which figure window the image should appear in. If figure does not have a number associated with it, then figures will appear chronologically as they appear in the M-file. Figures loaded in bitmap file, the image and converted to a grayscale image, a loaded JPEG file, and the image converted to a grayscale image, respectively. The images used in this example are both MATLAB example images. In order to demonstrate how to load an image file, these images were copied and pasted into the folder denoted in the M-file.

CHAPTER 4
2D IMAGE RESTORATION

4.1 Introduction:

The goal of the image restoration is to recover an image that has been blurred in some way. In computational image processing blurring is usually modelled by a convolution of image matrix and a blur kernel. A blur kernel in this case is a two-dimensional matrix which describes the response of an imaging system to a point light source or a point object. Another term for it is Point Spread Function. Let's suppose we have three two-dimensional matrices: f(x,y) for the original image, h(m,n) for the blurring kernel and *g(x,y)* for the blurred image. Than we can write the convolution:

$$g(x,y) = \sum_m \sum_n h(m,n) \cdot f(x+m, y+n) \qquad \text{eq.(4.1)}$$

4.2 2D Image restoration methodology

We are expecting a clear/restored image, by giving the degraded image input. The degraded image has to be sharpen in order to reconstruct. Convert the reconstructed image and degraded image in Fast Fourier Transform (FFT). The degradation function i.e.; Point Spread Function (PSF) of the image can be estimated by using the reconstructed and degraded image.

H= (abs (F)>0.01).*G./F

Where

G= FFT (degraded image)

H=FFT (reconstructed image)

After finding the PSF, the kernel should be determined. Kernel is a small matrix, which is used for sharpening, edge detecting and more. The kernel is taken in the spatial domain, which is the Inverse Fast Fourier Transform (IFFT) of the degraded function.

Kernel=ifft (H)

For coloured images, we have 3 planes, so we take average over each plane. The degraded matrix and kernel matrix get convolved to form a new matrix. In order to remove additive noise, we are going for restoration filters. After applying filter, the noise gets removed and final restored image can be observed.

4.3 Results

Input mage:

Fig: 4.3.1. Input image

Weiner filter:

Fig 4.3.2. Output of wiener filter

The fig 4.3.2 represents the wiener filter output of fig: 4.3.1. The PSNR value for this output is9.2337 dB, it has SSIM value of0.16496and the RMSE value is 38.149

Truncated Inverse Filter:

Fig 4.3.3.output of truncated inverse filter

The fig 4.3.3 represents the Truncated Inverse filter output of fig: 4.3.1 . The PSNR value for this output is9.9076 dB, it has SSIM value of 0.16841 and the RMSE value is 38.7926

Inverse filter:

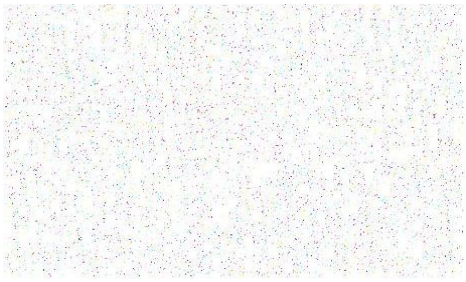


Fig 4.3.4. Output of inverse filter

The fig 4.3.4 represents the Inverse filter output of fig: 4.3.1. The PSNR value for this output is4.9076 dB, it has SSIM value of 0.0481 and the RMSE value is 75.47

CLS Filter:

Fig 4.3.5. Output of CLS filter

The fig 4.3.5 represents the CLS filter output of fig: 4.3.1. The PSNR value for this output is9.3253 dB, it has SSIM value of 0.16854 and the RMSE value is 38.1252

Comparisons:

Parameters	**Weiner Filter**	**Inverse Filter**	**Truncated Inverse Filter**	**CLS Filter**
RMS	38.149	75.47	38.7926	38.1252
PSNR(dB)	9.2337	4.9076	9.9076	9.3253
SSIM	0.16496	0.1681	0.0481	0.16854

For 2nd image:

Input image:

Fig 4.3.6. Input image

Weiner filter output:

Fig 4.3.7.Output of wiener filter

The fig 4.3.7 represents the wiener filter output of fig: 4.3.6. The PSNR value for this output is9.512 dB, it has SSIM value of 0.2574 and the RMSE value is 38.123

Inverse filter:

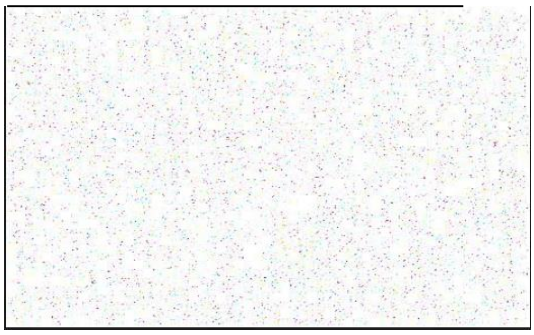


Fig 4.3.8. Output of inverse filter

The fig 4.3.8 represents the Inverse filter output of fig: 4.3.6. The PSNR value for this output is4.51 dB, it has SSIM value of 0.062 and the RMSE value is 77.91

Truncated Inverse Filter:

Fig 4.3.9. Output of Truncated inverse filter

The fig 4.3.9 represents the Truncated Inverse filter output of fig': 4.3.6. The PSNR value for this output is 38.751 dB, it has SSIM value of 0.0453 and the RMSE value is 9.4532

CLS Filter:

Fig 4.3.10. Output of CLS filter

The fig 4.3.10 represents the CLS filter output of fig: 4.3.6. The PSNR value for this output is 9.5491 dB, it has SSIM value of 0.1534 and the RMSE value is 37.2143.

Comparisons:

Parameters	**Weiner Filter**	**Inverse Filter**	**Truncated Inverse Filter**	**CLS Filter**
RMS	38.123	77.91	9.4532	37.2143
PSNR(dB)	9.512	4.51	38.751	9.5491
SSIM	0.2574	0.062	0.0453	0.1534

For 3rd image:
Input image:

Fig 4.3.11 Input image.

Wiener filter:

Fig 4.3.12 Output of wiener filter

The fig 4.3.12 represents the wiener filter output of fig: 4.3.11. The PSNR value for this output is9.1332 dB, it has SSIM value of 0.1892 and the RMSE value is 37.295

Truncated inverse filter:

Fig 4.3.13 Output of Truncated inverse filter

The fig 4.3.13 represents the Truncated Inverse filter output of fig': 4.3.11. The PSNR value for this output is 37.684 dB, it has SSIM value of 0.0472 and the RMSE value is 9.7432

Inverse filter:

Fig 4.3.14 Output of Inverse filter

The fig 4.3.14 represents the Inverse filter output of fig: 4.3.11. The PSNR value for this output is4.86 dB, it has SSIM value of 0.054 and the RMSE value is 75.57

CLS filter:

Fig 4.3.15 Output of CLS filter

The fig 4.3.15 represents the CLS filter output of fig: 4.3.11. The PSNR value for this output is 9.4122 dB, it has SSIM value of 0.1784 and the RMSE value is 39.1324.

Comparisons:

Parameters	**Wiener Filter**	**Inverse Filter**	**Truncated Inverse Filter**	**CLS Filter**
RMS	37.295	76.66	9.7432	39.1324
PSNR(dB)	9.1332	4.78	37.684	9.4122
SSIM	0.1892	0.045	0.0472	0.1784

CHAPTER 5
3D RESTORATION AND RESULTS

5.1 Introduction

By the use of the 3D model the desired structure is output in the form of a boundary surface that separates the structure from the background. In this approach, the desired features in each acquired 2D image are classified and segmented. The operator outlines the boundaries of the areas of interest on the 2D images manually or with automated computerized methods. Moreover, these boundaries can be distinguished from adjacent tissues by assigning a value or color, and thus a 3D model is obtained. The main advantage of this technique is related to the reduction of the amount of 3D data, consisting of only a few boundaries or structure information. Furthermore, the contrast between different structures is artificially increased. The major limitation of this approach is the segmentation data process. In fact, accurate segmentation of ultrasound data, essential for high-quality rendering, is a difficult problem because, to date, high-performance automatic segmentation systems are not available and the manual approach is tedious and time-consuming. As a consequence, this method for 3D image reconstruction is not used for many applications.

5.2 Algorithms and 3D Types

- **3D Rotation:**

3D Rotation is more complicated than 2D rotation since we must specify an axis of rotation. In 2D the axis of rotation is always perpendicular to the xy plane, i.e., the Z axis, but in 3D the axis of rotation can have any spatial orientation.3D Rotation is a process of rotating a pixel in an image with respect to an angle in a three-dimensional plane. Consider a point object O has to be rotated from one angle to another in a 3D plane. The 3D case requires full 3D modelling of the object 3D objects moving in a plane perpendicular to the camera optical axis with in-plane rotation. Extension to 3D rotation is a significant conceptual change that would require modelling of f and m in 3D and introducing their projection onto the image plane.

- **IMAGE PRE-PROCESSING:**

The pre-processing stage is the essential step towards retaining the important information. The filtering and preparation of image will take a considerable amount of processing time. Image pre-processing includes cleaning, normalization, transformation, feature extraction and selection, etc. In order to the improve the quality of the image, the raw image is preprocessed so as to improve the efficiency and ease of the mining process. Preprocessing has a major impact on the quality of feature extraction and a profound output of image analysis. A dataset has a common feature descriptor method that corresponds to mathematical normalization in preprocessing.

- **EDGE DETECTION AND TYPES:**

Edge detection for colour images presents additional challenges because of the three colour components used. Edge detectors are very useful for locating objects within images. There are many different kinds of edge detectors, but we will concentrate on two: the Sobel edge detector and the canny edge detector. The Sobel edge detector is able to look for strong edges in the horizontal direction, vertical direction, or both directions. The

Canny edge detector detects all strong edges plus it will find weak edges that are associated with strong edges. Both of these edge detectors return binary images with the edges shown in white on a black background. The Example, below, demonstrates the use of these edge detectors. The Canny and Sobel edge detectors are both demonstrated in this. Two methods for performing edge detection using the Sobel method are shown. The first method uses the MATLAB functions, fspecial, which creates the filter, and imfilter, which applies the filter to the image. The second method uses the MATLAB function, edge, in which you must specify the type of edge detection method desired. Sobel was used as the first edge detection method, while Canny was used as the next type.

The first image is the original image; the image denoted Horizontal Sobel is the result of using fspecial and imfilter. The image labelled Sobel is the result of using the edge filter with Sobel specified, while the image labelled Canny has Canny specified.

The Zoom In tool was used to depict the detail in the images more clearly. As you can see, the filter used to create the Horizontal Sobel image detects horizontal edges much more readily than vertical edges. The filter used to create the Sobel image detected both horizontal and vertical edges. This resulted from MATLAB looking for both horizontal and vertical edges independently and then summing them. The Canny image demonstrates how well the Canny method detects all edges.

The Canny method does not only look for strong edges, as in the Sobel method, but also will look for weak edges that are connected to strong edges and show those, too. Next Conversion into grey scale. For RGB, computing the luminance image is usually easy and efficient. The main drawback is that important edges are often not confined to the component. Therefore, we convert into grey level for proper edge detection. Next step is Denoise by median filter. One of the fundamental challenges in the field of image processing is image denoising, where the goal is to estimate the original image by suppressing noise from a noise-contaminated version of the image.

- **MEDIAN FILTERING:**

The median filter is the filtering technique used for noise removal from images and signals. Median filter is very crucial in the image processing field as it is well known for the preservation of edges during noise removal. The prior duty of the filter is to scan every input data interceding the overall entries with the median function known as “window” method. The window tends to be a little bit complex over the higher-dimensional signals. The number of medians is set according to the number of windows, falling under odd and even categories.

Since the median value must actually be the value of one of the pixels in the neighbourhood, the median filter does not create new unrealistic pixel values when the filter straddles an edge. For this reason, the median filter is much better at preserving sharp edges than the mean filter.

The pre-processing stage is the essential step towards retaining the important information. The filtering and preparation of image will take a considerable amount of processing time. Image pre-processing includes cleaning, normalization, transformation, feature extraction and selection, etc. In order to the improve the quality of the image, the raw image is pre-processed so as to improve the efficiency and ease of the mining process.

Pre-processing has a major impact on the quality of feature extraction and a profound output of image analysis. A dataset has a common feature descriptor method that corresponds to mathematical normalization in pre-processing.

- **HISTOGRAM:**

Histogram-based methods have been proven their ability in image enhancement. To improve low contrast while preserving details and high brightness in near-infrared images, a novel method called adaptive gamma correction based on cumulative histogram (AGCCH) is studied in this paper. This novel image enhancement method improves the contrast of local pixels through adaptive gamma correction (AGC), which is formed by incorporating a cumulative histogram or cumulative sub-histogram into the weighting distribution. Both qualitatively and quantitatively, experimental results demonstrate that the proposed image enhancement with the AGCCH method can perform well in brightness preservation, contrast enhancement, and detail preservation, and it is superior to previous state-of-the-art methods.

Histogram Equalization is a computer image processing technique used to improve contrast in images. It accomplishes this by effectively spreading out the most frequent intensity values, i.e., stretching out the intensity range of the image. This method usually increases the global contrast of images when its usable data is represented by close contrast values. This allows for areas of lower local contrast to gain a higher contrast.

A histogram is bar graph that shows a distribution of data. In image processing histograms are used to show how many of each pixel value are present in an image. Histograms can be very useful in determining which pixel values are important in an image. From this data you can manipulate an image to meet your specifications. Data from a histogram can aid you in contrast enhancement and thresholding. In order to create a histogram from an image, use the imhist function. Contrast enhancement can be performed by the histeq function, while thresholding can be performed by using the grayish function and the im2bw function. See Figure 14,15,16,17 for a demonstration of imhist, imadjust, graythresh, and im2bw. If you want to see the resulting histogram of a contrast enhanced image, simply perform the imhist operation on the image created with histeq.

- **IMAGE SEGMENTATION:**

Image Segmentation is the process by which a digital image is partitioned into various subgroups (of pixels) called Image Objects, which can reduce the complexity of the image, and thus analysing the image becomes simpler. We use various image segmentation algorithms to split and group a certain set of pixels together from the image. By doing so, we are actually assigning labels to pixels and the pixels with the same label fall under a category where they have some or the other thing common in them.

Using these labels, we can specify boundaries, draw lines, and separate the most required objects in an image from the rest of the not-so-important ones. In the below example, from a main image on the left, we try to get the major components, e.g., chair,

table etc. and hence all the chairs are coloured uniformly. In the next tab, we have detected instances, which talk about individual objects, and hence the all the chairs have different colours. This is how different methods of segmentation of images work in varying degrees of complexity and yield different levels of outputs.

- **THERSHOLDING:**

Thresholding is a very popular segmentation technique, used for separating an object from its background. The process of thresholding involves, comparing each pixel value of the image (pixel intensity) to a specified threshold. This divides all the pixels of the input image into 2 groups:

Pixels having intensity value lower than threshold.

Pixels having intensity value greater than threshold.

These 2 groups are now given different values, depending on various segmentation types. A digital image is made up of various components that need to be "analysed", let's use that word for simplicity sake and the "analysis" performed on such components can reveal a lot of hidden information from them. This information can help us address a plethora of business problems – which is one of the many end goals that are linked with image processing.

5.3 3D METHODOLOGY

- The image used is the MATLAB image, image.tif, which can be found in the manner described.
- The first method uses the MATLAB functions, fspecial, which creates the filter, and imfilter, which applies the filter to the image.
- The second method uses the MATLAB function, edge, in which you must specify the type of edge detection method desired. Sobel was used as the first edge detection method, while Canny was used as the next type.
- The first image is the original image; the image denoted Horizontal Sobel is the result of using fspecial and imfilter.
- The image labelled Sobel is the result of using the edge filter with Sobel specified, while the image labelled Canny has Canny specified.
- The Zoom In tool was used to depict the detail in the images more clearly. As you can see, the filter used to create the Horizontal Sobel image detects horizontal edges much more readily than vertical edges. The filter used to create the Sobel image detected both horizontal and vertical edges. This resulted from MATLAB looking for both horizontal and vertical edges independently and then summing them. The Canny image demonstrates how well the Canny method detects all edges.
- The Canny method does not only look for strong edges, as in the Sobel method, but also will look for weak edges that are connected to strong edges and show those, too. Next Conversion into grey scale. For RGB, computing the luminance image is usually easy and efficient. The main drawback is that important edges are often not confined to the component. Therefore, we convert into grey level for proper edge detection.

- Next step is Denoise by median filter. One of the fundamental challenges in the field of image processing is image denoising, where the goal is to estimate the original image by suppressing noise from a noise-contaminated version of the image.
- Here denoising is done with median filter. Median Filters can be very useful for removing noise from images. A median filter is like an averaging filter in some ways. The averaging filter examines the pixel in question and its neighbour's pixel values and returns the mean of these pixel values. The median filter looks at this same neighbourhood of pixels, but returns the median value. In this way noise can be removed, but edges are not blurred as much, since the median filter is better at ignoring large discrepancies in pixel values.
- Next step is threshold segmentation. Thresholding is a type of image segmentation, where we change the pixels of an image to make the image easier to analyse. In thresholding, we convert grayscale image into a binary image, i.e., simply black and white.
- Most frequently, we use thresholding as a way to select areas of interest of an image, while ignoring the other parts. The process works like this. First, we will load the original image, convert it to grayscale, and blur it with one of the methods. Then, we will use the > operator to apply the threshold t, a number in the closed range [0.0, 1.0].in our code we applied the value 0.5, Pixels with colour values on one side of t will be turned "on," while pixels with colour values on the other side will be turned "off."
- After threshold segmentation we have to go for finding the ratio of white and greyish regions, that's is nothing but finding histogram of the image.
- The histogram output of the image appears like a white background with some spikes (graphically). Since the image has a white background, most of the pixels in the image are white. The histogram output will be like some spikes according to the image. If we want to select the shapes and not the background, we want to turn off the white background pixels and the pixels for the shapes turned on. So, we should choose a value of t somewhere before the large peak. here we choose t=0.5 that means the peak will be somewhere after the 0.5 value.
- Next is adaptive gamma correction, Gamma correction changes the brightness of an image based on a fixed parameter gamma. Contrast enhancement improves the visibility of an image by enhancing the difference in brightness of objects and their backgrounds. Gamma correction helps in displaying an image accurately on screen. Gamma correction controls the overall brightness of an image. Images which are not properly corrected can look either bleached out, or too dark. Adaptive gamma correction is a modification function combining traditional gamma correction and histogram equalization methods.
- Contrast Limited AHE (CLAHE) is a variant of adaptive histogram equalization in which the contrast amplification is limited, so as to reduce this problem of noise amplification. 'adapthisteq' function enhances the contrast of each pixel.
- After performing the equalization, adapthisteq combines neighbouring pixels using bilinear interpolation to eliminate artificially induced boundaries. The 'haze' function used to provide thickness T, a rough approximation of the depth D of the scene, defined up to an unknown multiplication factor.

- A region in an image is a group of connected pixels with similar properties. Regions are important for the interpretation of an image because they may correspond to objects in a scene. So, we estimated White regions + other parts and grey regions + other parts.
- Dehaze function used to remove the multiplication factor applied through 'haze' function during the histogram.
- Final step is displaying the 3D output. By following these steps, we achieved 3d image restoration of fast-moving objects.

5.4 RESULTS

Fig.5.4.1 output of 3D image restoration for different inputs

Result for figure 5.4.1 (a, b, c)

Images	PSNR	RMSE	SSIM
Fig a	9.6753 dB	38.5334	0.1687
Fig b	10.5621 dB	39.6783	0.1734
Fig c	6.2130 dB	38.7652	0.1654

5.5 Comparison between 2D and 3D

In 2D, deblurring process is performed all over the image. Edges are not properly recovered by the deconvolution process as we are finding the kernel of the degraded image and its get convolved with the original image to get the resultant output image. Ringing effect occurs to the restored images. But in 3D, Deblurring process is performed on the desired part of the image using thresholding. Edges are properly recovered because here we are using the median filter, which does not alter the edges. Smooth image can be observed as an output without any ringing effect or sharpening. In 2D image restoration CLS filter gives the best result so we used the same filter for 3D to get better design metric values.

Fig.5.5.1. 2D CLS filter output

PSNR=9.3253 dB

Fig.5.5.2. 3D CLS filter output

PSNR=9.6753 dB

Fig.5.5.3. 2D CLS filter output

PSNR= 9.5491 dB

Fig.5.5.4. 3D CLS filter output

PSNR= 10.5621 dB

Fig.5.5.5. 2D CLS filter output

PSNR=4.8612 dB

Fig.5.5.6. 3D CLS filter output

PSNR= 6.2130 dB

CONCLUSION:

Deblatting means deblurring plus image matting and this problem is ill-posed due to the composition of finding image shape, size , motion etc., of how the image being blurred or degraded under a static background. We showed that this deblatting problem is different from the blind deconvolution in many ways. The main difference is in the estimation of shape. We proposed an effective restoration method that is applying restoration filters methods for solving the convex optimisation problems with 2D translation and rotation. 3D restoration algorithm is used for 3D rotation method. Performance of each filter was compared qualitatively by using metrics like PSNR, SSIM, MSE values that we got experimentally. The proposed method to 3D image restoration motion increases its practical usefulness and so this is the main interest of our work. Blind deconvolution method is applicable to only 2D image restoration, as we extended our work to 3D image restoration, 3D rotation algorithm should come into picture. The main differences between 2D and 3D restorations are in 2D restoration irrespective of the blurred part, the restoration is performed all over the image. Unlike in 2D restoration, in 3D restoration the degraded part is highlighted using thresholding and restoration is performed on the highlighted part. In 2D restoration we cannot recover the edges, while in 3D the edges of the image is correctly recovered. After restoring the image in 2D, the resultant image gets sharpened and resulting in the ringing effect. Whereas in 3D, the output image is smoothed.

REFERENCES:

1. D. Rozumnyi, J. Kotera, F. Sroubek, L. Novotny, and J. Matas, “The world of fast moving objects,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 4838– 4846.
2. M. Sorel, F. Sroubek, and J. Flusser, “Towards super-resolution in the presence of spatially varying blur,” in Super-resolution imaging (P. Milanfar, ed.), pp. 187–218, CRC Press, 2010.
3. R. Molina, J. Mateos, and A. K. Katsaggelos, “Blind deconvolution using a variational approach to parameter, image, and blur estimation,” IEEE Trans.
4. Image Process. vol. 15, no. 12, pp. 3715–3727, Dec. 2006.
5. M. Tico and M. Vehvilainen, “Bayesian estimation of motion blur point spread function from differently exposed image frames,” in Proc. EUSIPCO 2006, 2006.
6. L. Li, W. Huang, I. Y. H. Gu, and Q. Tian, “Foreground object detection in changing background based on colour co-occurrence statistics,” in Proc. IEEE Workshop on Applications of Computer Vision
7. Strekalovskiy E, Cremers D (2014) Real-time minimization of the piecewise smooth mumford-shah functional. In: Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings,
8. Part II, pp 127–141
9. A. K. Katsaggelos, Ed., Digital Image Restoration. New York: Springer-Verlag, 1991.
10. Richardson, W.H.: Bayesian-based iterative method of image restoration, Journal of the Optical Society of America (1917-1983) (1972).

11. C. Vogel, Computational Methods for Inverse Problems, SIAM, 2002.

12. S. Cho, J. Wang, and S. Lee, “Handling outliers in non-blind image deconvolution,” in Proc. Int. Conf. Comput. Vis., Nov. 2011, pp. 495–502.

13. J. Pan, Z. Lin, Z. Su, and M.-H. Yang, “Robust kernel estimation withoutliers handling for image deblurring,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 2800–2808.

14. J. Kotera, V. Smidl, and F. Sroubek, “Blind deconvolution withmodel discrepancies,” IEEE Trans. Image Process., vol. 26, no. 5,pp. 2533–2544, May 2017.

15. C. J. Schuler, M. Hirsch, S. Harmeling, and B. Scholkopf, “Learningto deblur,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 38, no. 7, pp. 1439–1451, Jul. 2016.

16. O. Whyte, J. Sivic, A. Zisserman, and J. Ponce, “Non-uniform deblurringfor shaken images,”Int. J. Comput. Vis., vol. 98, no. 2, pp. 168–186, Jun. 2012.

17. M. Hirsch, C. J. Schuler, S. Harmeling, and B. Scholkopf, “Fastremoval of non-uniform camera shake,” inProc. Int. Conf. Comput.Vis., Nov. 2011, pp. 463–470.

18. O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas,“DeblurGAN: Blind motion deblurring using conditional adversarialnetworks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.,Jun. 2018, pp. 8183–8192.

19. M. Jin, G. Meishvili, and P. Favaro, “Learning to extract a videosequence from a single motion-blurred image,” in Proc. IEEE/CVF Conf.Comput. Vis. Pattern Recognit., Jun. 2018, pp. 6334–6342.

20. X. Tao, H. Gao, X. Shen, J. Wang, and J. Jia, “Scale-recurrent networkfor deep image deblurring,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 8174– 8182.

21. O. Kupyn, T. Martyniuk, J. Wu, and Z. Wang, “DeblurGAN-v2: Deblurring (orders-ofmagnitude) faster and better,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 8877–8886.

22. T. H. Kim, B. Ahn, and K. M. Lee, “Dynamic scene deblurring,” in Proc. IEEE Int. Conf. Comput. Vis., Dec. 2013, pp. 3160–3167.

23. T. H. Kim and K. M. Lee, “Segmentation-free dynamic scene deblurring,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2014, pp. 2766–2773.

24. J. Pan, Z. Hu, Z. Su, H.-Y. Lee, and M.-H. Yang, “Soft-segmentationguided object motion deblurring,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 459– 468.

25. S. Nah, T. H. Kim, and K. M. Lee, “Deep multi-scale convolutionalneural network for dynamic scene deblurring,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 257–265.

26. H. Gao, X. Tao, X. Shen, and J. Jia, “Dynamic scene deblurringwith parameter selective sharing and nested skip connections,” in Proc.IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 3843–3851.

27. [25] F. Šroubek and J. Flusser, “Multichannel blind iterative image restoration,” IEEE Trans, Image Process., vol. 12, no. 9, pp. 1094–1106, Sep. 2003.

28. A. Rav-Acha and S. Peleg, “Two motion-blurred images are better than one,” Pattern Recognit. Lett., vol. 26, no. 3, pp. 311–317, Feb. 2005.

29. H. Zhang, D. Wipf, and Y. Zhang, “Multi-observation blind deconvolution with an adaptive sparse prior,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 36, no. 8, pp. 1628–1643, Aug. 2014.

30. G. Harikumar and Y. Bresler, “Perfect blind restoration of images blurredby multiple filters: Theory and efficient algorithms,” IEEE Trans. Image Process., vol. 8, no. 2, pp. 202– 219, Feb. 1999.

31. G. B. Giannakis and R. W. Heath, “Blind identification of multichannelFIR blurs and perfect image restoration,” IEEE Trans. Image Process.,vol. 9, no. 11, pp. 1877–1896, Nov. 2000.

32. F. Sroubek and J. Flusser, “Multichannel blind deconvolution of spatially misaligned images,” IEEE Trans. Image Process., vol. 14, no. 7,pp. 874–883, Jul. 2005. [31] J. Jia, “Single image motion deblurring using transparency,” in Proc.IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2007, pp. 1–8.

33. Q. Shan, W. Xiong, and J. Jia, “Rotational motion deblurring of a rigidobject from a single image,” in Proc. IEEE 11th Int. Conf. Comput. Vis.,Oct. 2007, pp. 1–8.

34. S. Dai and Y. Wu, “Motion from blur,” in Proc. IEEE Conf. Comput.Vis. Pattern Recognit., Jun. 2008, pp. 1–8.

35. A. Levin, D. Lischinski, and Y. Weiss, “A closed-form solution to natural image matting,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 30, no. 2,pp. 228–242, Feb. 2008.

36. C. Dong, S. Barbero, and J. Portilla, “Nonconvex Bayesian restoration of blurred foreground images,” in Proc. IEEE Int. Conf. Image Process. (ICIP), Sep. 2019, pp. 754–758.