**DEBLURRING OF BLURRED IMAGE USING EXTENDED BLIND DECONVOLUTION TECHNIQUE**

**A PROJECT REPORT**

***Submitted by***

Y. VENKAIAHNAIDU 314126510118

R. NAVATEJA 314126510081

R. SARAT KUMAR 314126510080

S. ARAVIND BABU 314126510088

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE ENGINEERING**



Under esteemed guidance of

**Mr. B. Ravi Kiran**

(Associative Professor)

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES**

(Affiliated to Andhra University)

**SANGIVALASA, VISAKHAPATNAM – 531162**

2017-2018

**DEBLURRING OF BLURRED IMAGE USING EXTENDED BLIND DECONVOLUTION TECHNIQUE**

**A PROJECT REPORT**

***Submitted by***

Y. VENKAIAHNAIDU 314126510118

R. NAVATEJA 314126510081

R. SARAT KUMAR 314126510080

S. ARAVIND BABU 314126510088

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE ENGINEERING**



Under esteemed guidance of

**Mr. B. Ravi Kiran**

(Associative Professor)

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES**

(Affiliated to Andhra University)

**SANGIVALASA, VISAKHAPATNAM – 531162**

2017-2018

**ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES**

**(Affiliated to Andhra University)**

**SANGIVALASA, VISAKHAPATNAM-531162**

**BONAFIDE CERTIFICATE**

Certified that this project report “**DEBLURRING OF BLURRED IMAGE USING EXTENDED BLIND DECONVOLUTION**” is the bonafide work of “ Y. VENKAIAHNAIDU ( 314126510118 ) , R. NAVATEJA ( 314126510081 ), R. SARAT KUMAR ( 314126510080) , S. ARAVIND BABU (314126510088) ” who carried out the project work under my supervision.

Dr. Sivaranjini B. Ravi Kiran

**HEAD OF THE DEPARTMENT PROJECT GUIDE**

Professor Associative Professor Department of Computer Science Department of Computer Science

& Engineering &Engineering

ANITS ANITS

**DECLARATION**

This is to certify that the project work entitled “**DEBLURRING OF BLURRED IMAGE USING EXTENDED BLIND DECONVOLUTION**” is a bonafide work carried out by **Y.VENKAIAH NAIDU,R.NAVATEJA ,R.SARAT KUMAR , S.ARAVINDBABU** as a part of **B.TECH** final year 2nd semester of **computer science Engineering** of Andhra University, Visakhapatnam during the year 2017-18.

We, **Y.VENKAIAH NAIDU, R.NAVATEJA ,R.SARAT KUMAR , S.ARAVINDBABU** of final semester B.Tech., in the department of Computer Science & Engineering from ANITS, Visakhapatnam, hereby declare that the project work entitled  **DEBLRRING OF BLURRED IMAGE USING EXTENDED BLIND DECONVOLUTION** is carried out by us and submitted in partial fulfillment of the requirements for the award of **Bachelor of Technology in Computer Science Engineering** , under Anil Neerukonda Institute of Technology & Sciences during the academic year 2017-18 and has not been submitted to any other university for the award of any kind of degree.

Y.VENKAIAHNAIDU 314126510118

R.NAVATEJA 314126510081

R.SARAT KUMAR 314126510080

S.ARAVIND BABU 314126510088

**ACKNOWLEDGEMENT**

The satisfaction and euphoria that accompany the successful completion of task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement always boosted the morale. We take a great pleasure in presenting a project, which is the result of a studied blend of both research and knowledge.

We first take the privilege to thank the Head of our Department, **Dr.Sivaranjini**, for permitting us in laying the first stone of success and providing the lab facilities, we would also like to thank the other staff in our department and lab assistants who directly or indirectly helped us in successful completion of the project.

We feel great to thank **Mr.B.Ravi Kiran**, who are our project guides and who shared their valuable knowledge with us and made us understand the real essence of the topic and created interest in us to work day and night for the project; we also thank our B.Tech coordinator **Mrs.T.Kranthi**, for her support and encouragement.

We also like to thank **Mr.P.Sainath Vital** for providing us the lab resources; we also thank our friends and college staff who extended their part of support in the successful completion of the project.

Y.VENKAIAHNAIDU 314126510118

R.NAVATEJA 314126510081

R.SARAT KUMAR 314126510080

S.ARAVIND BABU 314126510088

**ABSTRACT**

Observed images of a scene are usually degraded by blurring due to atmospheric turbulence and inappropriate camera settings. The images are further degraded by the various noises present in the environment and the system. Therefore it is essential to get a sharp clean image from the noisy blurred image. In digital imaging, blurring is a bandwidth reduction of the image due to imperfect image construction process which gives poor image quality. Some blurring always arises in the recording of a digital image. Along with these blurring effects, noise always corrupts any recorded image. Reconstructing process is divided into two categories, first is nonblind in which the blurring function is given and the degradation process is inverted using one of the restoration algorithms and second blind where blurring operator is not known. Deconvolution using blind method is very complex process where image recovery is performed with little or no prior knowledge of the degrading PSF. The PSF represent the impulse response of a point source. In this paper Blind Deconvolution method has been implemented to deblur​​ a​​ single​​ image. ​​By applying SNR to images greater the SNR then we get clear image as output.

**Keywords:** Blind​​Deconvolution, ​​Image, ​​ Noise, PSF, SNR.

**CONTENTS**

**TITLE Page No.**

Abstract 9

Keywords 9

**1. Introduction 10**

1.1 Problem statement 27

1.2 Motivation 28

1.3 Contribution 28

1.4 Research Methodology 29

**2. Literature Survey 30**

2.1 Introduction to Matlab 30

2.2 Matlab Functions 32

2.3 Deconvolution 45

2.4 Point Spread Function 46

2.5 Non Blind Deconvolution 47

2.6 Blind Deconvolution 48

**3. System Software Requirements 50**

3.1 Software Requirements 51

3.2 Hardware Requirements 51

3.2.1 User Interface 51

3.2.2 Hardware Interface 51

3.2.3 Software Interface 51

**4. Existing techniques 52**

4.1 About Deconvolution 52

### 4.2 Point Spread Function 53

### 4.2. Existing algorithms 52

4.3. Features 52

4.4. Disadvantages 53

**5. Proposed System 54**

5.1 Architecture 60

5.2 Processing Steps 61

**6. Design 62**

6.1 Structure Chart 62

6.2 UML Diagrams 65

6.2.1 Sequence Diagram 66

6.2.2 Activity Diagram 67

6.2.3 Component Diagram 68

6.2.4 Deployment Diagram 68

**7. Methodologies 70**

7.1 Comparing 70

7.2 Estimating PSF values 71

7.3 Estimating 71

**8. Implementation 72**

8.1Sample Code if the blur information is unknown 72

8.2 Sample Code if the blur information is known: 75

**9. Results 78**

9.1 deblurring of images whose blur information is known 78 9.2 deblurring of images whose blur information is unknown 79

**10. Testing 80** 10.1 Testing for SNR values: 80

**11. Conclusion 85**

**References 86**

**1.INTRODUCTION**

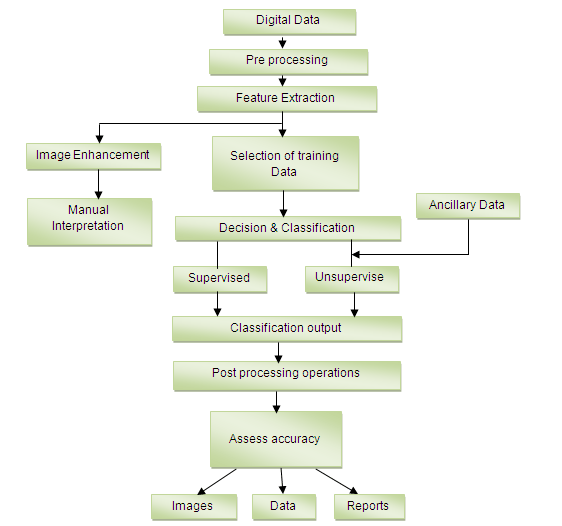
Image processingis a method to convert an image into digital form and perform some operations on it, to get an enhanced image or to extract some useful information from it. It is a type of signal dispensation in which input is image, like video frame or photograph and output may be image or characteristics associated with that image.

The purpose of image processing is divided into five categories. They are visualization, image sharpening and restoration, image retrieval, Image recognition, Measurement of pattern.

The two types of methods used for Image Processing are Analog and Digital Image Processing. Analog or visual techniques of image processing can be used for the hard copies like printouts and photographs. Image analysts use various fundamentals of interpretation while using these visual techniques. The image processing is not just confined to area that must be studied but on knowledge of analyst. Association is another important tool in image processing through visual techniques. So analysts apply a combination of personal knowledge and collateral data to image processing.

**Analog image processing** is any [image processing](https://en.wikipedia.org/wiki/Image_processing) task conducted on [two-dimensional](https://en.wikipedia.org/wiki/Two-dimensional) [analog signals](https://en.wikipedia.org/wiki/Analog_signal) by analog means (as opposed to [digital image processing](https://en.wikipedia.org/wiki/Digital_image_processing)). Basically any data can be represented in two types named as 1. Analog 2. Digital if the pictorial representation of the data represented in analog wave formats that can be named as analog image. Eg: television broadcasting in older days through the dish antenna systems.

**Digital image processing** is the use of computer [algorithms](https://en.wikipedia.org/wiki/Algorithm) to perform [image processing](https://en.wikipedia.org/wiki/Image_processing) on [digital images](https://en.wikipedia.org/wiki/Digital_image). As a subcategory or field of [digital signal processing](https://en.wikipedia.org/wiki/Digital_signal_processing), digital image processing has many advantages over [analog image processing](https://en.wikipedia.org/wiki/Analog_image_processing). It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and signal distortion during processing. Since images are defined over two dimensions (perhaps more) digital image processing may be modeled in the form of [multidimensional systems](https://en.wikipedia.org/wiki/Multidimensional_systems).



## 

Digital image processing allows the use of much more complex algorithms, and hence, can offer both more sophisticated performance at simple tasks, and the implementation of methods which would be impossible by analog means.

Digital image processing is the only practical technology for:

* **Classification:**

Digital image classification uses the quantitative spectral information contained in an image, which is related to the composition or condition of the target surface. Image analysis can be performed on multispectral as well as hyperspectral imagery. It requires an understanding of the way materials and objects of interest on the earth's surface absorb, reflect, and emit radiation in the visible, near-infrared, and thermal portions of the electromagnetic spectrum. In order to make use of image analysis results in a GIS environment, source image should be orthorectified so that the final image analysis product, whatever its format, can be overlaid with other imagery, terrain data, and other geographic data layers. Classification results are initially in raster format, but they may be generalized to polygons with further processing. There are several core principles of image analysis that pertain specifically to the extraction of information and features from remotely sensed data.

* Spectral differentiation
* Radiometric differentiation
* Spatial differentiation

While certain aspects of digital image classification are completely automated, a human image analyst must provide significant input. There are two basic approaches to classification, supervised and unsupervised, and the type and amount of human interaction differs depending on the approach chosen.

* Supervised classification
* Unsupervised classification

Classification schemes may be comprised of hard, discrete categories; in other words, each pixel is assigned to one, and only one, class. Fuzzy classification schemes allow a proportional assignment of multiple classes to pixels. The entire image scene may be processed pixel-by-pixel, or the image may be decomposed into homogeneous image patches for object-oriented classification. As stated by Jensen (2005), “no pattern classification method is inherently superior to any other.” It is up to the analyst, using his/her knowledge of the problem set, the study area, the data sources, and the intended use of the results, to determine the most appropriate, efficient, time and cost-effective approach.

* **[Feature extraction:](https://en.wikipedia.org/wiki/Feature_extraction)**

Feature extraction describes the relevant shape information contained in a pattern so that the task of classifying the pattern is made easy by a formal procedure. In pattern recognition and in image processing, feature extraction is a special form of dimensionality reduction. The main goal of feature extraction is to obtain the most relevant information from the original data and represent that information in a lower dimensionality space. When the input data to an algorithm is too large to be processed and it is suspected to be redundant (much data, but not much information) then the input data will be transformed into a reduced representation set of features (also named features vector). Transforming the input data into the set of features is called feature extraction. If the features extracted are carefully chosen it is expected that the features set will extract the relevant information from the input data in order to perform the desired task using this reduced representation instead of the full size input Pattern recognition is an emerging field of research in the area of image processing. It has been used in many applications such as character recognition,document verification, reading bank deposit slips, extracting information from cheques, applications for credit cards, health insurance, loan, tax forms, data entry, postal address reading, check sorting, tax reading, script recognition etc. Character recognition is also applicable in newly emerging areas, such as development of electronic libraries, multimedia database, and systems which require handwriting data entry.

Feature extraction is done after the preprocessing phase in character recognition system. The primary task of pattern recognition is to take an input pattern and correctly assign it as one of the possible output classes. This process can be divided into two general stages: Feature selection and Classification. Feature selection is critical to the whole process since the classifier will not be able to recognize from poorly selected features.

* **[Multi-scale signal analysis:](https://en.wikipedia.org/wiki/Multi-scale_signal_analysis)**

Multi-scale Signal processing concerns the analysis, synthesis, and modification of signals, which are broadly defined as functions conveying "information about the behavior or attributes of some phenomenon",such as sound, images, and biological measurements. For example, signal processing techniques are used to improve signal transmission fidelity, storage efficiency, and subjective quality, and to emphasize or detect components of interest in a measured signal.

* **[Pattern recognition:](https://en.wikipedia.org/wiki/Pattern_recognition)**

Although pattern recognition and image processing have developed as two separate disciplines, they are very closely related. The area of image processing consists not only of coding, filtering, enhancement, and restoration, but also analysis and recognition of images. On the other hand, the area of pattern recognition includes not only feature extraction and classification, but also preprocessing and description of patterns. It is true that image processing appears to consider only two-dimensional pictorial patterns and pattern recognition deals with one-dimensional, two-dimensional, and three-dimensional patterns in general. However, in many cases, information about onedimensional and three-dimensional patterns is easily expressed as two-dimensional pictures, so that they are actually treated as pictorial patterns. Furthermore, many of the basic techniques used for pattern recognition and image processing are very similar in nature. Differences between the two disciplines do exist, but we also see an increasing overlap in interest and a sharing of methodologies between them in the future.

Pattern recognition is concerned primarily with the description and classification of measurements taken from physical or mental processes. Many definitions of pattern recognition have been proposed to provide an effective and efficient description of patterns.

* **[Projection:](https://en.wikipedia.org/wiki/Graphical_projection)**

Graphical projection is a protocol, used in technical drawing, by which an image of a three-dimensional object is projected onto a planar surface without the aid of numerical calculation.

This can be achieved in image processing using imaginary "projectors". The projected, mental image becomes the technician’s vision of the desired, finished picture.

Parallel projection, the lines of sight from the object to the projection plane are parallel to each other. Thus, lines that are parallel in three-dimensional space remain parallel in the two-dimensional projected image. Parallel projection also corresponds to a perspective projection with an infinite focal length (the distance from a camera's lens and focal point), or "zoom".

Perspective projection also means that lines which are parallel in nature (that is, meet at the point at infinity) appear to intersect in the projected image, for example if railways are pictured with perspective projection, they appear to converge towards a single point, called vanishing point. Photographic lenses and the human eye work in the same way, therefore perspective projection looks most realistic. Perspective projection is usually categorized into one-point, two-point and three-point perspective, depending on the orientation of the projection plane towards the axes of the depicted object

Some techniques which are used in digital image processing include:

* **[Anisotropic diffusion:](https://en.wikipedia.org/wiki/Anisotropic_diffusion)**

In image processing and computer vision, anisotropic diffusion, also called Perona–Malik diffusion, is a technique aiming at reducing image noise without removing significant parts of the image content, typically edges, lines or other details that are important for the interpretation of the image.[1][2][3] Anisotropic diffusion resembles the process that creates a scale space, where an image generates a parameterized family of successively more and more blurred images based on a diffusion process. Each of the resulting images in this family are given as a convolution between the image and a 2D isotropic Gaussian filter, where the width of the filter increases with the parameter. This diffusion process is a linear and space-invariant transformation of the original image. Anisotropic diffusion is a generalization of this diffusion process: it produces a family of parameterized images, but each resulting image is a combination between the original image and a filter that depends on the local content of the original image. Therefore, anisotropic diffusion is a non-linear and space-variant transformation of the original image.

* **[Hidden Markov models:](https://en.wikipedia.org/wiki/Hidden_Markov_model)**

Hidden Markov Model (HMM) is a statistical Markov model in which the system being modeled is assumed to be a Markov process with unobserved (i.e. hidden) states.

A hidden Markov model can be considered a generalization of a mixture model where the hidden variables (or latent variables), which control the mixture component to be selected for each observation, are related through a Markov process rather than independent of each other. Recently, hidden Markov models have been generalized to pairwise Markov models and triplet Markov models which allow consideration of more complex data structures and the modeling of nonstationary data.

Hidden Markov models are especially known for their application in reinforcement learning and temporal pattern recognition such as speech, handwriting, gesture recognition, part-of-speech tagging, musical score following, partial discharges and bioinformatics.

* **[Image editing:](https://en.wikipedia.org/wiki/Image_editing)**

Image editing encompasses the processes of altering images, whether they are digital photographs, traditional photo-chemical photographs, or illustrations. Traditional analog image editing is known as photo retouching, using tools such as an airbrush to modify photographs, or editing illustrations with any traditional art medium. Graphic software programs, which can be broadly grouped into vector graphics editors, raster graphics editors, and 3D modelers, are the primary tools with which a user may manipulate, enhance, and transform images. Many image editing programs are also used to render or create computer art from scratch.

* Selection
* Layers
* Image size alteration
* Cropping an image
* Cutting out a part of an image from the background
* Histogram
* Noise reduction
* Removal of unwanted elements
* Selective color change
* Image orientation
* Perspective control and distortion
* Lens correction
* Enhancing images
* Sharpening and softening images
* Selecting and merging of images
* Slicing of images
* Special effects
* **[Image restoration:](https://en.wikipedia.org/wiki/Image_restoration)**

Image Restoration is the operation of taking a corrupt/noisy image and estimating the clean, original image. Corruption may come in many forms such as motion blur, noise and camera mis-focus. Image restoration is performed by reversing the process that blurred the image and such is performed by imaging a point source and use the point source image, which is called the Point Spread Function (PSF) to restore the image information lost to the blurring process.

Image restoration is different from image enhancement in that the latter is designed to emphasize features of the image that make the image more pleasing to the observer, but not necessarily to produce realistic data from a scientific point of view. Image enhancement techniques (like contrast stretching or de-blurring by a nearest neighbor procedure) provided by imaging packages use no a priori model of the process that created the image.

With image enhancement noise can effectively be removed by sacrificing some resolution, but this is not acceptable in many applications. In a fluorescence microscope, resolution in the z-direction is bad as it is. More advanced image processing techniques must be applied to recover the object.

* **[Independent component analysis:](https://en.wikipedia.org/wiki/Independent_component_analysis)**

Independent component analysis attempts to decompose a multivariate signal into independent non-Gaussian signals. As an example, sound is usually a signal that is composed of the numerical addition, at each time t, of signals from several sources. The question then is whether it is possible to separate these contributing sources from the observed total signal. When the statistical independence assumption is correct, blind ICA separation of a mixed signal gives very good results. It is also used for signals that are not supposed to be generated by mixing for analysis purposes

A common example application is the "cocktail party problem" of listening in on one person's speech in a noisy room.

* **[Linear filtering:](https://en.wikipedia.org/wiki/Linear_filter)**

Linear filters process time-varying input signals to produce output signals, subject to the constraint of linearity. This results from systems composed solely of components (or digital algorithms) classified as having a linear response. Most filters implemented in analog electronics, in digital signal processing, or in mechanical systems are classified as causal, time invariant, and linear signal processing filters.

The general concept of linear filtering is also used in statistics, data analysis, and mechanical engineering among other fields and technologies. This includes non-causal filters and filters in more than one dimension such as those used in image processing; those filters are subject to different constraints leading to different design methods.

* **[Neural networks:](https://en.wikipedia.org/wiki/Artificial_neural_networks)**

Artificial neural networks (ANNs) or connectionist systems are computing systems vaguely inspired by the biological neural networks that constitute animal brains.Such systems "learn" (i.e. progressively improve performance on) tasks by considering examples, generally without task-specific programming. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labeled as "cat" or "no cat" and using the results to identify cats in other images. They do this without any a priori knowledge about cats, e.g., that they have fur, tails, whiskers and cat-like faces. Instead, they evolve their own set of relevant characteristics from the learning material that they process.

An ANN is based on a collection of connected units or nodes called artificial neurons (a simplified version of biological neurons in an animal brain). Each connection (a simplified version of a synapse) between artificial neurons can transmit a signal from one to another. The artificial neuron that receives the signal can process it and then signal artificial neurons connected to it

* **[Partial differential equations:](https://en.wikipedia.org/wiki/Partial_differential_equations)**

In mathematics, a partial differential equation (PDE) is a differential equation that contains unknown multivariable functions and their partial derivatives. PDEs are used to formulate problems involving functions of several variables, and are either solved by hand, or used to create a relevant computer model. A special case is ordinary differential equations (ODEs), which deal with functions of a single variable and their derivatives.

PDEs can be used to describe a wide variety of phenomena such as sound, heat, electrostatics, electrodynamics, fluid dynamics, elasticity, or quantum mechanics. These seemingly distinct physical phenomena can be formalised similarly in terms of PDEs. Just as ordinary differential equations often model one-dimensional dynamical systems, partial differential equations often model multidimensional systems. PDEs find their generalisation in stochastic partial differential equations.

* **[Pixilation:](https://en.wikipedia.org/wiki/Pixelation)**

In computer graphics, pixelation (or pixellation in British English) is caused by displaying a bitmap or a section of a bitmap at such a large size that individual pixels, small single-colored square display elements that comprise the bitmap, are visible. Such an image is said to be pixelated.

Early graphical applications such as video games ran at very low resolutions with a small number of colors, resulting in easily visible pixels. The resulting sharp edges gave curved objects and diagonal lines an unnatural appearance. However, when the number of available colors increased to 256, it was possible to gainfully employ anti-aliasing to smooth the appearance of low-resolution objects, not eliminating pixilation but making it less jarring to the eye. Higher resolutions would soon make this type of pixilation all but invisible on the screen, but pixilation is still visible if a low-resolution image is printed on paper.

* **[Principal components analysis:](https://en.wikipedia.org/wiki/Principal_components_analysis)**

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. If there are n observations with p variables, then the number of distinct principal components is min(n-1,p). This transformation is defined in such a way that the first principal component has the largest possible variance (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components. The resulting vectors are an uncorrelated orthogonal basis set. PCA is sensitive to the relative scaling of the original variables.

* **[Self-organizing maps:](https://en.wikipedia.org/wiki/Self-organizing_map)**

A self-organizing map (SOM) or self-organizing feature map (SOFM) is a type of artificial neural network (ANN) that is trained using unsupervised learning to produce a low-dimensional (typically two-dimensional), discretized representation of the input space of the training samples, called a map, and is therefore a method to do dimensionality reduction. Self-organizing maps differ from other artificial neural networks as they apply competitive learning as opposed to error-correction learning (such as backpropagation with gradient descent), and in the sense that they use a neighborhood function to preserve the topological properties of the input space

While it is typical to consider this type of network structure as related to feedforward networks where the nodes are visualized as being attached, this type of architecture is fundamentally different in arrangement and motivation

* **[Wavelets:](https://en.wikipedia.org/wiki/Wavelet)**

A wavelet is a wave-like oscillation with an amplitude that begins at zero, increases, and then decreases back to zero. It can typically be visualized as a "brief oscillation" like one recorded by a seismograph or heart monitor. Generally, wavelets are intentionally crafted to have specific properties that make them useful for signal processing. Using a "reverse, shift, multiply and integrate" technique called convolution, wavelets can be combined with known portions of a damaged signal to extract information from the unknown portions.

For example, a wavelet could be created to have a frequency of Middle C and a short duration of roughly a 32nd note. If this wavelet were to be convolved with a signal created from the recording of a song, then the resulting signal would be useful for determining when the Middle C note was being played in the song. Mathematically, the wavelet will correlate with the signal if the unknown signal contains information of similar frequency. This concept of correlation is at the core of many practical applications of wavelet theory

**TYPES OF BLUR:**

There are various types of blur. It sounds odd. In fact there is a lot more to blur than most people realize. It is quite a varied subject. It is used in nearly all aspects of photography.

##### Subject-movement types of blur:

When a subject move in front of your stationary camera the resulting image has a blurred subject. This is movement blur. The types of blur which include movement can be varied. In the picture above the motor bikes are moving at around 90 miles per hour. When taking this shot I was the far bike resulting in that bike being sharp. The pan meant that my camera was not paced at the same speed as the nearest bike. As a result, its movement was relatively out of synchronization with my camera. The nearest bike was in relative movement and thus blurred.



a)subject-movement type of blur

##### More types of blur Camera movement:

When a subject is moving pan your camera with it. I did that in the bike picture at the top of the page and got a sharp bike placed against a blurred background. That is not bokeh in the background. As the camera panned with the bike it captured a stationary background. However, as the camera was moving it created a movement blur on the background.

Movement blur of the background normally occurs when [panning](http://www.photokonnexion.com/?page_id=12318). If you hold a stationary camera out of a car window and take a long exposure and the same type of blur will result. However, nothing will be sharp in that case (unless something next to you is travelling at your exact speed).

Done right background blur from camera movement has great impact. In the motorbikes above it gives a race feel. It looks really fast.

##### Out of focus types of blur:

Of course, it is possible to completely blur a shot quite deliberately. Some pleasant [aesthetic](http://www.photokonnexion.com/?page_id=6315) effects can be achieved. Wedding and romantic photographers love the “soft focus” shot. This is a deliberate very slight lack of sharpness. It emphasizes the romantic, soft nature of something kittens, brides, the first kiss, baby and so on.Google images of soft focus shots provides quite a good range of possibilities for this type of blur.

The soft-focus shot can be created different ways. Each give slightly different types of blur. You can literally set the lens to manual focus. Then when properly focused pull the focus slightly back. To create a small amount of blur. Another way to do it is to use a soft-focus filter. These are simply screwed to the end of the lens and give the same effect. When I was first starting out in photography many wedding photographers carried a flesh colored or white nylon stocking. Pulled tight over the lens while the photograph is taken it creates a soft-focus effect. Others like a skylight (ultraviolet) filter with a tiny amount of grease smeared on it. All these work, but give you a slightly different soft-focus effect.



b) out of focus type of blur

##### Zoom blur:

One of the less well-known types of blur – zoom blur. You need a steady

##### hand or better, a [tripod](http://www.photokonnexion.com/?page_id=3145). It makes the picture look like the world is rushing toward you very rapidly.

Adjusting the zoom during exposure creates zoom blur. Set your camera to have a long exposure – around one second is good. Balance the shutter speed with the ISO and aperture to get a proper exposure. You will need to use manual focus to adjust the zoom in the shot. Press the [shutter button](http://www.photokonnexion.com/?page_id=3663) and rotate the zoom focus ring. A short turn or through its full arc – the amount of turn gives different effects. With a bit of practice you can reduce hand-shake blur. A smooth zoom throughout the exposure creates some great effects.



c)zoomblur image

##### Artificial blur:

Most image editors have software filters to create types of blur. In fact, there are a variety of different software filters available. Gaussian blur is one common type. It softens or smooths the image, but also causes loss of detail. There is also rotational blur (self-explanatory); linear blur or movement blur – you choose the direction of the blur. Other editing packages will have other blur types too.

Artificial types of blur do not have the same effect as blurs created in-camera. Artificial blur tends to lack depth. Whereas, blur using [depth of field](http://www.photokonnexion.com/?page_id=4258) gives depth to a picture. The bokeh and movement blurs both have the impact of realism and depth as they vary throughout the depth of the image. Applying a uniform artificial blur can affect the realism. Applied with care and artful work you can make artificial blur look real. It is all about care and attention.



d) Artifical blurtype image

Hand movement during a shot causes all sorts of blur. You get blurred shadows, blurred faces, possibly jerky tracks not good at all. However, you can have some fun with this sort of movement. Some famous pictures have been created by deliberate hand movements. There are lots of shots,like tree shots where the movement of the camera creates a surreal or abstract view of the subject. Some people have tried [throwing their camera and triggering it in mid-air](http://www.photokonnexion.com/?p=7462) – some bizarre results can be obtained (including a smashed camera).

* 1. **PROBLEM STATEMENT:**

Inimage processing ,blind deconvolution is a deconvolution technique that permits recovery of the target scene from a single or set of "blurred" images in the presence of a poorly determined or unknownpoint spread function (psf).Regular linear and non-linear deconvolution techniques utilize a known PSF. For blind deconvolution, the PSF is estimated from the image or image set, allowing the deconvolution to be performed. Researchers have been studying blind deconvolution methods for several decades, and have approached the problem from different directions. Most of the work on blind deconvolution started in early 1970s. Blind deconvolution is used in astronomical imaging and medical imaging. Blind deconvolution can be performed iteratively, whereby each iteration improves the estimation of the PSF and the scene, or non-iteratively, where one application of the algorithm, based on exterior information, extracts the psf.

**1.2 MOTIVATION:**

* Blur is a common and unwanted artifact of image acquisition
* A lot amount of data is lost in the image due the blurriness of the image.
* Even though the work is done where are not getting exact information in the image due to blurriness.
* Important pictures taken the satellites of the motion are appeared to be blurred.
* Images captured by the secrete camera are most often blurred so we need an technique that could estimate the blur and deblur it

**1.3 CONTRIBUTION:**

We deblur the image easily when the blur information is known to us where in our project we work the images whose blur information is unknown to us, We prove that blind deconvolution is the best technique out there when the blurriness of the image is not known to us so we compare the output with different images generated by other techniques like Lucy-Richardson method, Regularized filter method, Wiener filter method.

**1.4 RESEARCH METHODOLOGY:**

We take photographs to preserve special momentums in our life or to store information most of these photographs cannot be recreated if they didn’t come out well due to blurriness. Since the blind image deconvolution techniques is good for deblurring the images whose blurriness not known we choose blind deconvolution techniques.

**2. LITERATURE SURVEY**

**2.1 Introduction to MATLAB:**

[Cleve Moler](https://en.wikipedia.org/wiki/Cleve_Moler), the chairman of the [computer science](https://en.wikipedia.org/wiki/Computer_science) department at the [University of New Mexico](https://en.wikipedia.org/wiki/University_of_New_Mexico), started developing MATLAB in the late 1970s.[[8]](https://en.wikipedia.org/wiki/MATLAB#cite_note-origins-8) He designed it to give his students access to [LINPACK](https://en.wikipedia.org/wiki/LINPACK) and [EISPACK](https://en.wikipedia.org/wiki/EISPACK) without them having to learn [Fortran](https://en.wikipedia.org/wiki/Fortran). It soon spread to other universities and found a strong audience within the [applied mathematics](https://en.wikipedia.org/wiki/Applied_mathematics) community. [Jack Little](https://en.wikipedia.org/wiki/John_N._Little), an engineer, was exposed to it during a visit Moler made to [Stanford University](https://en.wikipedia.org/wiki/Stanford_University) in 1983. Recognizing its commercial potential, he joined with Moler and Steve Bangert. They rewrote MATLAB in [C](https://en.wikipedia.org/wiki/C_(programming_language)) and founded [MathWorks](https://en.wikipedia.org/wiki/MathWorks" \o "MathWorks) in 1984 to continue its development. These rewritten libraries were known as JACKPAC. In 2000, MATLAB was rewritten to use a newer set of libraries for matrix manipulation, [LAPACK](https://en.wikipedia.org/wiki/LAPACK).

Although MATLAB is intended primarily for numerical computing, an optional toolbox uses the [MuPAD](https://en.wikipedia.org/wiki/MuPAD" \o "MuPAD) [symbolic engine](https://en.wikipedia.org/wiki/Computer_algebra_system), allowing access to [symbolic computing](https://en.wikipedia.org/wiki/Symbolic_computing) abilities. An additional package, [Simulink](https://en.wikipedia.org/wiki/Simulink), adds graphical multi-domain simulation and [model-based design](https://en.wikipedia.org/wiki/Model-based_design) for [dynamic](https://en.wikipedia.org/wiki/Dynamical_system) and [embedded systems](https://en.wikipedia.org/wiki/Embedded_system).

As of 2017, MATLAB has roughly 1 million users across industry and academia. MATLAB users come from various backgrounds of [engineering](https://en.wikipedia.org/wiki/Engineering), [science](https://en.wikipedia.org/wiki/Science), and [economics](https://en.wikipedia.org/wiki/Economics).

MATLAB was first adopted by researchers and practitioners in [control engineering](https://en.wikipedia.org/wiki/Control_engineering), Little's specialty, but quickly spread to many other domains. It is now also used in education, in particular the teaching of [linear algebra](https://en.wikipedia.org/wiki/Linear_algebra), [numerical analysis](https://en.wikipedia.org/wiki/Numerical_analysis), and is popular amongst scientists involved in [image processing](https://en.wikipedia.org/wiki/Image_processing).

MATLAB is widely used in all areas of applied mathematics, in education and research at universities, and in the industry. MATLAB stands for MATrix LABoratory and the software is built up around vectors and matrices. This makes the software particularly useful for linear algebra but MATLAB is also a great tool for solving algebraic and differential equations and for numerical integration. MATLAB has powerful graphic tools and can produce nice pictures in both 2D and 3D. It is also a programming language, and is one of the easiest programming languages for writing mathematical programs. MATLAB also has some tool boxes useful for signal processing, image processing, optimization, etc.

We gather this information from documentation of image processing and learn the several websites and implementation for above algorithm using matlab programming language developed by mathworks.

### Variables:

Variables are defined using the assignment operator, =. MATLAB is a [weakly typed](https://en.wikipedia.org/wiki/Strong_and_weak_typing) programming language because types are implicitly converted. It is an inferred typed language because variables can be assigned without declaring their type, except if they are to be treated as symbolic objects,[[13]](https://en.wikipedia.org/wiki/MATLAB#cite_note-13) and that their type can change. Values can come from [constants](https://en.wikipedia.org/wiki/Constant_(computer_science)), from computation involving values of other variables, or from the output of a function.

### Vectors and matrices:

A simple array is defined using the colon syntax: *initial*:*increment*:*terminator*.

[Indexing](https://en.wikipedia.org/wiki/One-based_indexing) is one-based,[[14]](https://en.wikipedia.org/wiki/MATLAB#cite_note-14) which is the usual convention for [matrices](https://en.wikipedia.org/wiki/Matrix_(mathematics)) in mathematics, although not for some programming languages such as C, C++, and Java.Matrices can be defined by separating the elements of a row with blank space or comma and using a semicolon to terminate each row. The list of elements should be surrounded by square brackets: []. Parentheses: () are used to access elements and subarrays (they are also used to denote a function argument list).

### Functions:

When creating a MATLAB function, the name of the file should match the name of the first function in the file. Valid function names begin with an alphabetic character, and can contain letters, numbers, or underscores. Functions are often case sensitive.

### Structures:

MATLAB has structure data types. Since all variables in MATLAB are arrays, a more adequate name is "structure array", where each element of the array has the same field names. In addition, MATLAB supports dynamic field names (field look-ups by name, field manipulations, etc.). Unfortunately, MATLAB JIT does not support MATLAB structures, therefore just a simple bundling of various variables into a structure will come at a cost.

### Classes and object-oriented programming:

MATLAB supports [object-oriented programming](https://en.wikipedia.org/wiki/Object-oriented_programming) including classes, inheritance, virtual dispatch, packages, pass-by-value semantics, and pass-by-reference semantics. However, the syntax and calling conventions are significantly different from other languages. MATLAB has value classes and reference classes, depending on whether the class has *handle* as a super-class (for reference classes) or not (for value classes).

**2.2 Matlab Functions:**

There are lot functions in Matlab we have used few of them

**Imread():**

A = imread(FILENAME, FMT) reads a grayscale or color image frthe file specified by the string FILENAME. FILENAME must be in the current directory, in a directory on the MATLAB path, or include a full or relative path to a file. The text string FMT specifies the format of the file by its standardfile extension. For example, specify 'gif' for Graphics InterchangeFormat files. To see a list ofsupported formats, with their fileextensions, use the IMFORMATS function. If imread cannot find a file named FILENAME, it looks for a file named FILENAME.FMT.

The return value A is an array containing the image data. If the filecontains a grayscale image, A is an M-by-N array. If the file containsa truecolor image, A is an M-by-N-by-3 array. For TIFF files containing color images that use the CMYK color space, A is an M-by-N-by-4 array.

The class of A depends on the bits-per-sample of the image data,rounded to the next byte boundary. For example, imread returns 24-bit color data as an array of uint8 data because the sample size for each color component is 8 bits. See the Remarks section for a discussion of bitdepths, and see the Format-Specific Information section for more detail about supported bitdepths and sample sizes for a particular format.

[X,MAP] = imread(FILENAME,FMT) reads the indexed image in FILENAME into X and its associated colormap into MAP. Colormap values in the image file are automatically rescaled into the range [0,1].

[...] = imread(FILENAME) attempts to infer the format of the file from its content.

[...] = imread(URL,...) reads the image from an Internet URL.

Bitdepth is the number of bits used to represent each image pixel. Bitdepth is calculated by multiplying the bits-per-sample with the samples-per-pixel. Thus, a format that uses 8-bits for each color component (or sample) and three samples per pixel has a bitdepth of 24.Sometimes the sample size associated with a bitdepth can be ambiguous: does a 48-bit bitdepth represent six 8-bit samples or three 16-bit samples? The following format-specific sections provide sample size information to avoid this ambiguity.

'ReductionLevel' A non-negative integer specifying the reduction in the resolution of the image. For a reduction level 'L', the image resolution is reduced by a factor of 2^L. The default value is 0 implying no reduction. The reduction level is limited by the total number of decomposition levels as provided by 'WaveletDecompositionLevels' field in the structure returned from IMFINFO function.

'PixelRegion' {ROWS, COLS}. imread returns the sub-image specified by the boundaries in ROWS and COLS.ROWS and COLS must both be two-element vector that denote the 1-based indices [START STOP]. If 'ReductionLevel' is greater than 0, then ROWS and COLS are coordinates in the reduced-sized image.

[...] = imread(...,'BackgroundColor',BG) composites any transparent pixels in the input image against the color specified in BG. If BG is'none', then no compositing is performed. Otherwise, if the input imageis indexed, BG should be an integer in the range [1,P] where P is thecolormap length. If the input image is grayscale, BG should be a value in the range [0,1]. If the input image is RGB, BG should be a three-element vector whose values are in the range [0,1]. The string 'BackgroundColor' may be abbreviated.

The class of A depends on the bits-per-sample of the image data,rounded to the next byte boundary. For example, imread returns 24-bit color data as an array of uint8 data because the sample size for each color component is 8 bits. See the Remarks section for a discussion of bitdepths, and see the Format-Specific Information section for more detail about supported bitdepths and sample sizes for a particular format.

Example:

imdata = imread('ngc6543a.jpg');

**Imshow():**

imshow Display image in Handle Graphics figure. imshow(I) displays the grayscale image I.

imshow(I,[LOW HIGH]) displays the grayscale image I, specifying the display range for I in [LOW HIGH]. The value LOW (and any value less than LOW)displays as black, the value HIGH (and any value greater than HIGH) displays as white. Values in between are displayed as intermediate shades of gray,using the default number of gray levels. imshow(I,[]) displays the grayscale image I scaling the display based on the range of pixel values in I. imshow uses [min(I(:)) max(I(:))] as the display range, that is, the minimum value in I is displayed as black, and the maximum value is displayed as white.

* imshow(RGB) displays the truecolor image RGB.
* imshow(BW) displays the binary image BW. imshow displays pixels with the value 0 (zero) as black and pixels with the value 1 as white.
* imshow(X,MAP) displays the indexed image X with the colormap MAP. imshow(FILENAME) displays the image stored in the graphics file FILENAME. The file must contain an image that IMREAD or DICOMREAD can read.

imshow calls IMREAD or DICOMREAD to read the image from the file, but does not store the image data in the MATLAB workspace. If the file contains multiple images, the first one will be displayed. The file must be in the current directory or on the MATLAB path. (DICOMREAD capability and the NITF file format require the Image Processing Toolbox.)

* imshow(IMG,RI,\_\_\_) displays the image IMG with associated 2-D spatial referencing object RI. IMG may be a grayscale, RGB, or binary image.

IMG may also be a graphics file FILENAME. (This syntax requires the

Image Processing Toolbox.)

* imshow(I,RI,[LOW HIGH]) displays the grayscale image I with associated 2-D spatial referencing object RI and a specified display range for I

in [LOW HIGH]. (This syntax requires the Image Processing Toolbox.)

* imshow(X,RX,MAP) displays the indexed image X with associated 2-D

spatial referencing object RX and colormap MAP. (This syntax requiresthe Image Processing Toolbox.)

* HIMAGE = imshow(\_\_\_) returns the handle to the image object created byimshow.

**Parameters include:**

* Border:String that controls whether a border is displayed around the image in the figure window. Valid strings are 'tight' and 'loose'. Note: There can still be a border if the image is very small, or if there are other objects besides the image and its axes in the figure.By default, the border is set to 'loose'.
* Colormap:2-D, real, M-by-3 matrix specifying a colormap. imshow uses this to set the colormap of the axes object containing the displayed image.Use this parameter to view grayscale images in false color.
* 'DisplayRange' Two-element vector [LOW HIGH] that controls the display range of a grayscale image. See above for more details about how to set [LOW HIGH].Including the parameter name is optional,except when the image is specified by afilename. The syntax imshow(I,[LOW HIGH]) is equivalent to imshow(I,'DisplayRange',[LOW HIGH]). The parameter name must be specified when using imshow with a filename, as in the syntax imshow(FILENAME,'DisplayRange',[LOW HIGH]). If I is an integer type, 'DisplayRange' defaults to the minimum and maximum representable values for that integer class. For images with floating point data, the default is [0 1].
* 'InitialMagnification' A numeric scalar value, or the text string 'fit',that specifies the initial magnification used to display the image. When set to 100, the image is displayed at 100% magnification. When set to 'fit' imshow scales the entire image to fit in the window.On initial display, the entire image is visible.If the magnification value would create an image that is too large to display on the screen, imshow warns and displays the image at the largest magnification that fits on the by default, the initial magnification is set to 100%.

Note: If you specify the axes position (using subplot or axes), imshow ignores any initial magnification you might have specified and defaults to the 'fit' behavior.

**Peak Signal-to-Noise Ratio (PSNR):**

[peaksnr](about:blank) = psnr([A](about:blank),[ref](about:blank)) calculates the peak signal-to-noise ratio for the image A, with the image ref as the reference. A and ref must be of the same size and class.

[peaksnr](about:blank) = psnr([A](about:blank),[ref](about:blank),[peakval](about:blank)) uses peakval as the peak signal value for calculating the peak signal-to-noise ratio for image A.

[[peaksnr](about:blank),[snr](about:blank)] = psnr(**\_\_\_**) returns the simple signal-to-noise ratio, snr, in addition to the peak signal-to-noise ratio.

Read image and create a copy with added noise. The original image is the reference image.

ref = imread('pout.tif');

A = imnoise(ref,'salt & pepper', 0.02);

[peaksnr, snr] = psnr(A, ref);

fprintf('\n The Peak-SNR value is %0.4f', peaksnr);

fprintf('\n The SNR value is %0.4f \n', snr);

The Peak-SNR value is 22.6437

The SNR value is 15.5524

**Title:**

Graph title.title('text') adds text at the top of the current axis title('text','Property1',PropertyValue1,'Property2',PropertyValue2,...)sets the values of the specified properties of the title

title(AX,...) adds the title to the specified axes.

H = title(...) returns the handle to the text object used as the title.

fspecial:

Create predefined 2-D filter.

h = fspecial(*type*)  
h = fspecial(*type*, parameters)

h = fspecial(*type*) creates a two-dimensional filter h of the specified type.

fspecial returns h as a correlation kernel, which is the appropriate form to use with imfilter. *type* is a string having one of these values.

|  |  |
| --- | --- |
| **Value** | **Description** |
| average | Averaging filter |
| disk | Circular averaging filter (pillbox) |
| gaussian | Gaussian lowpass filter |
| laplacian | Approximates the two-dimensional Laplacian operator |
| Log | Laplacian of Gaussian filter |
| motion | Approximates the linear motion of a camera |
| prewitt | Prewitt horizontal edge-emphasizing filter |
| Sobel | Sobel horizontal edge-emphasizing filter |

h = fspecial(*type*, parameters) accepts the filter specified by type plus additional modifying parameters particular to the type of filter chosen. If you omit these arguments, fspecial uses default values for the parameters.

The following list shows the syntax for each filter type. Where applicable, additional parameters are also shown.

* h = fspecial('average', hsize) returns an averaging filter h of size hsize. The argument hsize can be a vector specifying the number of rows and columns in h, or it can be a scalar, in which case h is a square matrix. The default value for hsize is [3 3].
* h = fspecial('disk', radius) returns a circular averaging filter (pillbox) within the square matrix of size 2\*radius+1. The default radius is 5.
* h = fspecial('gaussian', hsize, sigma) returns a rotationally symmetric Gaussian lowpass filter of size hsize with standard deviation sigma(positive). hsize can be a vector specifying the number of rows and columns in h, or it can be a scalar, in which case h is a square matrix. The default value for hsize is [3 3]; the default value for sigma is 0.5. Not recommended. Use [imgaussfilt](about:blank) or [imgaussfilt3](about:blank) instead.
* h = fspecial('laplacian', alpha) returns a 3-by-3 filter approximating the shape of the two-dimensional Laplacian operator. The parameteralpha controls the shape of the Laplacian and must be in the range 0.0 to 1.0. The default value for alpha is 0.2.
* h = fspecial('log', hsize, sigma) returns a rotationally symmetric Laplacian of Gaussian filter of size hsize with standard deviation sigma(positive). hsize can be a vector specifying the number of rows and columns in h, or it can be a scalar, in which case h is a square matrix. The default value for hsize is [5 5] and 0.5 for sigma.
* h = fspecial('motion', len, theta) returns a filter to approximate, once convolved with an image, the linear motion of a camera by len pixels, with an angle of theta degrees in a counterclockwise direction. The filter becomes a vector for horizontal and vertical motions. The default len is 9 and the default theta is 0, which corresponds to a horizontal motion of nine pixels.

To compute the filter coefficients, h, for 'motion':

* 1. Construct an ideal line segment with the desired length and angle, centered at the center coefficient of h.
  2. For each coefficient location (i,j), compute the nearest distance between that location and the ideal line segment.
  3. h = max(1 - nearest\_distance, 0);
  4. Normalize h:h = h/(sum(h(:)))

h = fspecial('prewitt') returns the 3-by-3 filter h (shown below) that emphasizes

horizontal edges by approximating a vertical gradient. If you need to emphasize vertical edges, transpose the filter h'.

1 1 1

0 0 0

-1 -1 -1

To find vertical edges, or for *x*-derivatives, use h'.

* h = fspecial('sobel') returns a 3-by-3 filter h (shown below) that emphasizes horizontal edges using the smoothing effect by approximating a vertical gradient. If you need to emphasize vertical edges, transpose the filter h'.

1 2 1

0 0 0

-1 -2 -1

**deconvblind:**

Deblur image using blind deconvolution

[J,PSF] = deconvblind(I, INITPSF)  
[J,PSF] = deconvblind(I, INITPSF, NUMIT)  
[J,PSF] = deconvblind(I, INITPSF, NUMIT, DAMPAR)  
[J,PSF] = deconvblind(I, INITPSF, NUMIT, DAMPAR, WEIGHT)  
[J,PSF] = deconvblind(I, INITPSF, NUMIT, DAMPAR, WEIGHT, READOUT)  
[J,PSF] = deconvblind(..., FUN, P1, P2,...,PN)  
[J,PSF] = deconvblind(I, INITPSF)

deconvolves image I using the maximum likelihood algorithm, returning both the deblurred image J and a restored point-spread function PSF. The restored PSF is a positive array that is the same size as INITPSF, normalized so its sum adds up to 1. The PSFrestoration is affected strongly by the size of the initial guess INITPSF and less by the values it contains. For this reason, specify an array of 1's as yourINITPSF.

I can be an N-dimensional array.

To improve the restoration, deconvblind supports several optional parameters, described below. Use [] as a placeholder if you do not specify an intermediate parameter.

[J,PSF] = deconvblind(I, INITPSF, NUMIT) specifies the number of iterations (default is 10).

[J,PSF] = deconvblind(I, INITPSF, NUMIT, DAMPAR) specifies the threshold deviation of the resulting image from the input image I (in terms of the standard deviation of Poisson noise) below which damping occurs. The iterations are suppressed for the pixels that deviate within the DAMPAR value from their original value. This suppresses the noise generation in such pixels, preserving necessary image details elsewhere. The default value is 0 (no damping).

[J,PSF] = deconvblind(I, INITPSF, NUMIT, DAMPAR, WEIGHT) specifies which pixels in the input image I are considered in the restoration. By default, WEIGHT is a unit array, the same size as the input image. You can assign a value between 0.0 and 1.0 to elements in the WEIGHT array. The value of an element in the WEIGHT array determines how much the pixel at the corresponding position in the input image is considered. For example, to exclude a pixel from consideration, assign it a value of 0 in the WEIGHT array. You can adjust the weight value assigned to each pixel according to the amount of flat-field correction.

[J,PSF] = deconvblind(I, INITPSF, NUMIT, DAMPAR, WEIGHT, READOUT), where READOUT is an array (or a value) corresponding to the additive noise (e.g., background, foreground noise) and the variance of the read-out camera noise. READOUT has to be in the units of the image. The default value is0.

[J,PSF] = deconvblind(..., FUN, P1, P2,...,PN), where FUN is a function describing additional constraints on the PSF.

**Deconvlucy:**

deconvlucy Deblur image using Lucy-Richardson method.

J = deconvlucy(I,PSF) deconvolves image I using Lucy-Richardson algorithm, returning deblurred image J. The assumption is that the image I was created by convolving a true image with apoint-spread function PSF and possibly by adding noise.

I can be an N-Dimensional array.

J = deconvlucy(I,PSF,NUMIT)

J = deconvlucy(I,PSF,NUMIT,DAMPAR)

J = deconvlucy(I,PSF,NUMIT,DAMPAR,WEIGHT)

J = deconvlucy(I,PSF,NUMIT,DAMPAR,WEIGHT,READOUT)

J=deconvlucy(I,PSF,NUMIT,DAMPAR,WEIGHT,READOUT,SUBSMPL), where NUMIT (optional) is the number of iterations (default is 10).

DAMPAR (optional) is an array that specifies the threshold deviation of the resulting image from the image I (in terms of the standard deviation of Poisson noise) below which the damping occurs. The iterations are suppressed for the pixels that deviate within the DAMPAR value from their original value. This suppresses the noise generation in such pixels, preserving necessary image details elsewhere. Default is 0 (no damping).

WEIGHT (optional) is assigned to each pixel to reflect its recording quality in the camera. A bad pixel is excluded from the solution by assigning it zero weight value. Instead of giving a weight of one for good pixels, you can adjust their weight according to the amount of flat-field correction. Default is a unit array of the same size as input image I. READOUT (optional) is an array (or a value) corresponding to the additive noise (e.g., background, foreground noise) and the variance of the read-out camera noise. READOUT has to be in the units of the image. Default is 0.

SUBSMPL (optional) denotes subsampling and is used when the PSF is given on a grid that is SUBSMPL times finer than the image. Default is 1.

**Deconvwnr:**

deconvwnr Deblur image using Wiener filter.

J = deconvwnr(I,PSF,NSR) deconvolves image I using the Wiener filter algorithm, returning deblurred image J. Image I can be an N-dimensional array. PSF is the point-spread function with which I was convolved. NSR is the noise-to-signal power ratio of the additive noise. NSR can be ascalar or a spectral-domain array of the same size as I. Specifying for the NSR is equivalent to creating an ideal inverse filter.

The algorithm is optimal in a sense of least mean square error between the estimated and the true images.

J = deconvwnr(I,PSF,NCORR,ICORR)deconvolves image I, where NCORR is the autocorrelation function of the noise and ICORR is the autocorrelation function of the original image. NCORR and ICORR can be of any size or dimension, not exceeding the original image. If NCORR or ICORR are N-dimensional arrays, the values correspond to the autocorrelation within each dimension. If NCORR or ICORR are vectors, and PSF is also a vector, the values represent the autocorrelation function in the first dimension. If PSF is an array, the 1-D autocorrelation function is extrapolated by symmetry to all non-singleton dimensions of PSF. If NCORR or ICORR is a scalar, this value represents the power of the noise of the image.

**Deconvreg:**

Deconvreg Deblur image using regularized filter.

J = deconvreg(I,PSF) deconvolves image I using regularized filter algorithm, returning deblurred image J. The assumption is that the image I was created by convolving a true image with a point-spread function PSF and possibly by adding noise. The algorithm is a constrained optimum in a sense of least square error between the estimated and the true images under requirement of preserving image smoothness. I can be an N-Dimensional array.

To improve the restoration, additional parameters can be passed in (use [] as a place holder if an intermediate parameter is unknown):

J = deconvreg(I,PSF,NP)

J = deconvreg(I,PSF,NP,LRANGE)

J = deconvreg(I,PSF,NP,LRANGE,REGOP), where

NP (optional) is the additive noise power. Default is 0.

LRANGE (optional) is a vector specifying range where search for the optimal solution is performed. The algorithm finds an optimal Lagrange multiplier, LAGRA, within the LRANGE range. If LRANGE is a scalar, the algorithm assumes that LAGRA is given and equal to LRANGE; the NP value is then ignored. Default is [1e-9 and 1e9].

REGOP (optional) is the regularization operator to constrain the deconvolution. To retain the image smoothness, the Laplacian regularization operator is used by default. The REGOP array dimensions must not exceed the image dimensions, any non-singleton dimensions must correspond to the non-singleton dimensions of PSF.

[J, LAGRA] = deconvreg(I,PSF,...) outputs value of the Lagrange multiplier, LAGRA, in addition to the restored image J.

**2.3 Deconvolution:**

This is an algorithm-based process used to reverse the effects of convolution on recorded data.The concept of deconvolution is widely used in the techniques of signal processing and image processing. Because these techniques are in turn widely used in many scientific and engineering disciplines, deconvolution finds many applications.In general, the object of deconvolution is to find the solution of a convolution equation of the form: Usually, h is some recorded signal, and ƒ is some signal that we wish to recover, but has been convolved with some other signal g before we recorded it. The function g might represent the transfer function of an instrument or a driving force that was applied to a physical system. The input is the corrupted natural image and one of the many existing deconvolution techniques is used to retrieve the true image. But this restored image is the estimation of the true image and hence the convergence of deconvolution techniques should provide the approximate and closest estimate of the true image. The blind image deconvolution on similar concept estimate the true image but there are almost no or partial information about the cause of degradation function. The partial information can be in the form of some finite support or non negativity of the image, coined as physical properties of the image. Similarly, this partial information can also be in the form of any statistical data such as entropy or probability distribution function of the signal. The different optimality criteria along with this partial information form the strong ground in image estimation.

**2.4 Point Spread Function:**

This is the function which describes the response of an imaging system to a point source or point object. The degradation producing ill-effect of blur is termed as the point spread function, PSF. Any type of blur is characterized by the PSF. The electromagnetic radiation or other imaging waves propagated from a point source or point object is known as the PSF. The degree of spreading(blurring) of the point object is a measure for the quality of an imaging system. g(n1,n2) = d(n1,n2) \* f(n1,n2) + w(n1,n2) where, g(n1,n2) is the degraded imaged(n1,n2) is the original imagef(n1,n2) is the point spread functionw(n1,n2) is the noise. Figure 4: Image formation with a Point Spread function 4.2 Classification of Restoration Algorithms Based on PSF Based on the PSF , the restoration algorithms are classified into two types. 1.Non-blind deconvolution. 2.Blind deconvolution.

**2.5 Non Blind Deconvolution:**

It refers to the deconvolution with explicit knowledge of the impulse response function used in the convolution. That is, the point spread function is known in advance in this technique.

**Richardson–Lucy deconvolution:**

The Richardson–Lucy algorithm, is an [iterative procedure](https://en.wikipedia.org/wiki/Iterative_procedure) for recovering a [latent image](https://en.wikipedia.org/wiki/Latent_image) that has been [blurred](https://en.wikipedia.org/wiki/Convolution) by a known [point spread function](https://en.wikipedia.org/wiki/Point_spread_function). It was named after William Richardson and Leon Lucy, who described it independently.

When an image is recorded on a detector such as [photographic film](https://en.wikipedia.org/wiki/Photographic_film) or a [charge coupled device](https://en.wikipedia.org/wiki/Charge_coupled_device), it is generally slightly blurred, with an ideal [point source](https://en.wikipedia.org/wiki/Point_source) not appearing as a point but being spread out, into what is known as the [point spread function](https://en.wikipedia.org/wiki/Point_spread_function).

**Wiener deconvolution** :

In [mathematics](https://en.wikipedia.org/wiki/Mathematics), **Wiener deconvolution** is an application of the [Wiener filter](https://en.wikipedia.org/wiki/Wiener_filter) to the [noise](https://en.wikipedia.org/wiki/Noise) problems inherent in [deconvolution](https://en.wikipedia.org/wiki/Deconvolution). It works in the [frequency domain](https://en.wikipedia.org/wiki/Frequency_domain), attempting to minimize the impact of deconvolved noise at frequencies which have a poor [signal-to-noise ratio](https://en.wikipedia.org/wiki/Signal-to-noise_ratio).

The Wiener deconvolution method has widespread use in [image](https://en.wikipedia.org/wiki/Image) deconvolution applications, as the frequency spectrum of most visual images is fairly well behaved and may be estimated easily. Wiener deconvolution is named after [Norbert Wiener](https://en.wikipedia.org/wiki/Norbert_Wiener)

**Regularization Deconvolution:**

In signal processing, total variation denoising, also known as total variationregularization, is a process, most often used in digital [image processing](https://en.wikipedia.org/wiki/Image_processing), that has applications in noise removal. It is based on the principle that signals with excessive and possibly spurious detail have high [total variation](https://en.wikipedia.org/wiki/Total_variation), that is, the integral of the absolute [gradient](https://en.wikipedia.org/wiki/Gradient) of the signal is high. According to this principle, reducing the total variation of the signal subject to it being a close match to the original signal, removes unwanted detail whilst preserving important details such as edges. The concept was pioneered by Rudin, Osher, and Fatemi in 1992 and so is today known as the *ROF model*.

This noise removal technique has advantages over simple techniques such as [linear smoothing](https://en.wikipedia.org/wiki/Gaussian_blur) or [median filtering](https://en.wikipedia.org/wiki/Median_filter) which reduce noise but at the same time smooth away edges to a greater or lesser degree. By contrast, total variation denoising is remarkably effective at simultaneously preserving edges whilst smoothing away noise in flat regions, even at low signal-to-noise ratios.

**2.6 Blind Deconvolution:**

It refers to the deconvolution without explicit knowledge of the impulse response function used in the convolution. Blind deconvolution is a deconvolution technique that permits recovery of the target scene from a single or set of "blurred" images in the presence of a poorly determined or unknown point spread function (PSF). Blind deconvolution can be performed iteratively, whereby each iteration improves the estimation of the PSF and the scene. Blind deconvolution techniques have always been a challenging and critical problem. But, since the techniques are more useful for the practical scenario compared to classical ones, the methods cannot be ignored. Different blind image deconvolution techniques assume various parameters to solve the problem. The literature review on different techniques reveal that some strong underlying concept has to be used as a key to crack the problem. Though the convergence is not welldefined as well as not sure. This motivates to search for the parameters responsible for degradation. The parameter once estimated is used to reverse the ill-effect.

**3.SYSTEM REQUIREMENT SPECIFICATION**

**3.1 SOFTWARES USED IN THIS PROJECT:**

**Language**: MATLAB

**OPERATING SYSTEM:** Windows

**3.2 HARDWARES USED IN THIS PROJECT:**

**PROCESSOR** :intel Multi Core processor

**RAM**: 2 GB or above

**HARDDISK**:500 GB or above

**3.2.1 USER INTERFACE:**

The work of the user is to take blur image and deblur it.

**3.2.2 HARDWARE INTERFACE:**

**MONITOR:** the outputs are displayed on the monitor screen.

**3.2.3 SOFTWARE INTERFACE:**

Matlab is use to implement Digital image processing which is used to create computer algorithms , process, communicate, and display digital images. Digital image processing algorithms can be used to Convert signals from an image sensor into digital images, Improve clarity, and remove noise and other artifacts. We take blurred image and we apply deconvolution techniques using matlab functions.

**4. EXISTING TECHNIQUE**

Blind deconvolution algorithm is a deconvolution technique that permits recovery of the target scene from a single or set of blurred images in the presence of a poorly determined or unknown point spread function. The algorithm maximizes the likelihood that the resulting image, when convolved with the resulting PSF, is an instance of the blurred image, assuming Poisson noise statistics. The blind deconvolution algorithm can be used effectively when no information about the distortion (blurring and noise) is known. The deconvblind function restores the image and the PSF simultaneously, using an iterative process like the accelerated, damped Lucy-Richardson algorithm.

**4.1About Deconvolution:**

This is an algorithm-based process used to reverse the effects of convolution on recorded data. The concept of deconvolution is widely used in the techniques of signal processing and image processing. Because these techniques are in turn widely used in many scientific and engineering disciplines, deconvolution finds many applications. In general, the object of deconvolution is to find the solution of a convolution equation of the form: Usually, h is some recorded signal, and ƒ is some signal that we wish to recover, but has been convolved with some other signal g before we recorded it. The function g might represent the transfer function of an instrument or a driving force that was applied to a physical system. The input is the corrupted natural image and one of the many existing deconvolution techniques is used to retrieve the true image. But this restored image is the estimation of the true image and hence the convergence of deconvolution techniques should provide the approximate and closest estimate of the true image. The blind image deconvolution on similar concept estimate the true image but there are almost no or partial information about the cause of degradation function. The partial information can be in the form of some finite support or non-negativity of the image, coined as physical properties of the image. Similarly, this partial information can also be in the form of any statistical data such as entropy or probability distribution function of the signal. The different optimality criteria along with this partial information form the strong ground in image estimation.

**4.2 Point Spread Function:**

This is the function which describes the response of an imaging system to a point source or point object. The degradation producing ill-effect of blur is termed as the point spread function, PSF. Any type of blur is characterized by the PSF. The electromagnetic radiation or other imaging waves propagated from a point source or point object is known as the PSF. The degree of spreading(blurring) of the point object is a measure for the quality of an imaging system. g(n1,n2) = d(n1,n2) \* f(n1,n2) + w(n1,n2) where, g(n1,n2) is the degraded imaged(n1,n2) is the original imagef(n1,n2) is the point spread functionw(n1,n2) is the noise. Figure 4: Image formation with a Point Spread function 4.2 Classification of Restoration Algorithms Based on PSF Based on the PSF , the restoration algorithms are classified into two types. 1.Non-blind deconvolution. 2.Blind deconvolution.

### 4.3. Existing algorithms:

For this project, these algorithms were chosen based on the most widely used and most successful. They also suit the requirement of this paper as well as good instances for reference for further readings and information.

The algorithms are as follow:

A. Blind Deconvolution Algorithms

B. Generating Point Spread function.

**4.4. Features**

The following are some of the many features that these algorithms provide:

• Effective to deblur the image using these algorithms.

• Detection of edges and subject of the image easily.

• It is a Simple Approach.

• Computationally simple than probability cases.

•Number of images are generated with the corresponding images, so we can select best image.

• Very simple to implement.

• Do feature selection resulting in comparatively simple classifier Fast.

• Simple and easy to Program.

• No former knowledge needed about weak learner.

• Less computationally expensive

### 4.5. Disadvantages:

There are certain drawbacks in the existing system which led for the idea of proposed system. The limitations are :

•The image can only deblurred only if the psf value is known.

•No source to calculate the quality of the image and amount of blur it contains.

•No camparisons between the images generated by the other algorithms is done.

• The result is not so much accurate.

• The methodology is complex.

**5. PROPOSED SYSTEM**

In addition to blind deconvolution algorithm we are finding signal to noise ratio (snr) values for an input image and for different output images getting by different PSF values respectively. By this we are comparing the output images snr values to original image then the highest value of snr image will consider as most appropriate deblurred image for input blurred image consider as that more noise for an input image will get reduced.

**ABOUT SNR:**

The signal-to-noise ratio (SNR) is used in imaging as a physical measure of the [sensitivity](https://en.wikipedia.org/wiki/Sensitometry) of a [(digital or film) imaging system](https://en.wikipedia.org/wiki/Photography). Industry standards measure SNR in [decibels](https://en.wikipedia.org/wiki/Decibels) (dB) of [power](https://en.wikipedia.org/wiki/Power_%28physics%29) and therefore apply the [10 log rule](https://en.wikipedia.org/wiki/10_log_rule) to the "pure" SNR *ratio* (a ratio of 1:1 yields 0 decibels, for instance). In turn, yielding the "sensitivity." Industry standards measure and define sensitivity in terms of the [ISO film speed](https://en.wikipedia.org/wiki/Film_speed) equivalent; SNR:32.04 dB = excellent image quality and SNR:20 dB = acceptable image quality. There are several options (and definitions) for the calculation of the SNR of or in an image.

First, there are two incompatible definitions of the SNR: SNR is frequently (e. g., in many engineering applications) defined as the ratio of the signal *power* and the noise *power* (which is consistent with the definition by Gonzalez given above), but – particularly in imaging – an alternative definition can be found, where SNR is given as the ratio of the *mean value* of the signal and the *standard deviation* of the noise. The power ratio (first definition above) is frequently expressed in dB (using the logarithm), while the signal ratio (second definition) is more often given as a number (of dimension 1).

Second, you must define in which part of the image the signal (power or mean value) is determined. Typical choices are: (1) the maximum power or intensity within the image; this gives you the peak-signal-to-noise ratio (PSNR); (2) the mean power or intensity; or (3) the power or signal of a reference structure within the image (e. g., in medical images with large amounts of (zerosignal) background this is more useful than including the background into the mean power or signal).

If you already know the noise standard deviation (and the statistical distribution of noise and its spatial distribution – noise may be distributed non-uniformly over the image), then you are done and you can calculate the SNR of your choice. (If you do not know the noise level (standard deviation or variance), it can be difficult to measure it reliably in an image – at least, if you do not have a noise-free image.)

**Blind Deconvolution:**

It refers to the deconvolution without explicit knowledge of the impulse response function used in the convolution. Blind deconvolution is a deconvolution technique that permits recovery of the target scene from a single or set of "blurred" images in the presence of a poorly determined or unknown point spread function (PSF). Blind deconvolution can be performed iteratively, whereby each iteration improves the estimation of the PSF and the scene. Blind deconvolution techniques have always been a challenging and critical problem. But, since the techniques are more useful for the practical scenario compared to classical ones, the methods cannot be ignored. Different blind image deconvolution techniques assume various parameters to solve the problem. The literature review on different techniques reveal that some strong underlying concept has to be used as a key to crack the problem. Though the convergence is not well defined as well as not sure.

**Richardson lucy:**

                         The Richardson–Lucy algorithm, also known as Lucy–Richardson [deconvolution](https://en.wikipedia.org/wiki/Deconvolution), is an [iterative procedure](https://en.wikipedia.org/wiki/Iterative_procedure) for recovering a [latent image](https://en.wikipedia.org/wiki/Latent_image) that has been [blurred](https://en.wikipedia.org/wiki/Convolution) by a known [point spread function](https://en.wikipedia.org/wiki/Point_spread_function). It was named after William Richardson and Leon Lucy, who described it independently.

        When an image is recorded on a detector such as [photographic film](https://en.wikipedia.org/wiki/Photographic_film) or a [hcharge coupled device](https://en.wikipedia.org/wiki/Charge_coupled_device), it is generally slightly blurred, with an ideal [point source](https://en.wikipedia.org/wiki/Point_source) not appearing as a point but being spread out, into what is known as the [point spread function](https://en.wikipedia.org/wiki/Point_spread_function). Non-point sources are effectively the sum of many individual point sources, and pixels in an observed image can be represented in terms of the point spread function and the latent image as

d i = ∑ j p i j u j { d\_{i}=\sum \_{j}p\_{ij}u\_{j}\,}

where p i j { p\_{ij}} is the point spread function (the fraction of light coming from true location j { j} that is observed at position i { i} ), u j { u\_{j}}  is the pixel value at location j { j} in the latent image, and d i { d\_{i}} is the observed value at pixel location i { i} . The statistics are performed under the assumption that u j { u\_{j}}  are [Poisson distributed](https://en.wikipedia.org/wiki/Poisson_distribution), which is appropriate for [photon noise](https://en.wikipedia.org/wiki/Photon_noise) in the data.

The basic idea is to calculate the [most likely](https://en.wikipedia.org/wiki/Maximum_likelihood) u j { u\_{j}} given the observed d i { d\_{i}} and known p i j { p\_{ij}} . This leads to an equation for u j { u\_{j}} which can be solved iteratively according to

u j ( t + 1 ) = u j ( t ) ∑ i d i c i p i j { u\_{j}^{(t+1)}=u\_{j}^{(t)}\sum \_{i}{\frac {d\_{i}}{c\_{i}}}p\_{ij}}

where

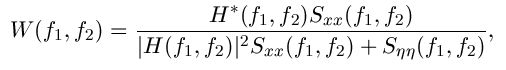
c i = ∑ j p i j u j ( t ) . { c\_{i}=\sum \_{j}p\_{ij}u\_{j}^{(t)}.}

It has been shown empirically that if this iteration converges, it converges to the maximum likelihood solution for u j { u\_{j}}.

**WIENER FILTERING:**

                         The inverse filtering is a restoration technique for deconvolution, i.e., when the image is blurred by a known lowpass filter, it is possible to recover the image by inverse filtering or generalized inverse filtering. However, inverse filtering is very sensitive to additive noise. The approach of reducing one degradation at a time allows us to develop a restoration algorithm for each type of degradation and simply combine them. The Wiener filtering executes an optimal tradeoff between inverse filtering and noise smoothing. It removes the additive noise and inverts the blurring simultaneously.

The Wiener filtering is optimal in terms of the mean square error. In other words, it minimizes the overall mean square error in the process of inverse filtering and noise smoothing. The Wiener filtering is a linear estimation of the original image. The approach is based on a stochastic framework. The orthogonality principle implies that the Wiener filter in Fourier domain can be expressed as follows:

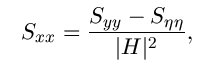


where https://lh4.googleusercontent.com/xFBPZojC97BL1jOsPPO9Dw0JmR6EahCUe0hbnjshhjCs40AE68Q9hKmLSVmWo5N6xu_9ek0MG8seQ6ZLQ-RdqohiQ9TEC0fqk4h-5ZbHpUKEAH5tkaaXiaKi4DBoLu_sDsM1xHcv are respectively power spectra of the original image and the additive noise, and https://lh3.googleusercontent.com/GwfVZVQFmNbrpuqj5OZvm24HhazOmXyR_EWSmp_spJOTcDj05WiTm2hxlt04uM4fscJOZe2klOdG8uPB7HOO8LuahH1MYHowUxwW7st58tloCuUetfIjKWOCmL0G-nx6OotGsZPU is the blurring filter. It is easy to see that the Wiener filter has two separate part, an inverse filtering part and a noise smoothing part. It not only performs the deconvolution by inverse filtering (highpass filtering) but also removes the noise with a compression operation (lowpass filtering). Implementation:

To implement the Wiener filter in practice we have to estimate the power spectra of the original image and the additive noise. For white additive noise the power spectrum is equal to the variance of the noise. To estimate the power spectrum of the original image many methods can be used. A direct estimate is the periodogram estimate of the power spectrum computed from the observation:

https://lh4.googleusercontent.com/iYPUCzQnWb3KCmoDQ3YSoNEwpVbFTVleUR1WGkUMf2CtYTR8nV4fu-ymV6OdtHcZuwLhi3Pt1OeoJDvxf-On0ckW70ragzV1topXTWLky28mU6EGs12QdwnbRRqoEn2-QY63mmTX

where **Y(k,l)** is the DFT of the observation. The advantage of the estimate is that it can be implemented very easily without worrying about the singularity of the inverse filtering. Another estimate which leads to a cascade implementation of the inverse filtering and the noise smoothing is



which is a straightforward result of the fact: https://lh3.googleusercontent.com/mHj_ShQa0kP4xiZ_Mqx5H0UC3258T7xC1SXeqq8KMh2R6aessX2h23zk3CDrifh5rcvv1-4m9r7GQTiy-WPmV8ijIXu1rIhE5k1mEoRnTLmVfpZ_bblCZ92FCJdLkZcDj2rycIX2 The power spectrum https://lh5.googleusercontent.com/oJ1hQQ6otHMrO00R_cihEsIKqAZAOWdv4lG13EJlCwzmZsL8ZPJEbpAm3SZOOn24Wmxg2lfysbyyF8jeRfDxFHjieKhuRh9bJDxkPrPJuCFGvHPu66lcEYnpC5c5wGXt0s32BcCi can be estimated directly from the observation using the periodogram estimate. This estimate results in a cascade implementation of inverse filtering and noise smoothing:

https://lh5.googleusercontent.com/hZws3K4FGbfF4Ey9fTOiXhDpNgqFQRzJsezjm7tZQdruJpaOHxlePn3FZfOENopTHqNaxUeAr1I0DKwolMMwAQeoq8FXQxBgH-4Y_QrsyjfcDxafb5aTnXCOw0DWvAPkMc7AI5GW

The disadvantage of this implementation is that when the inverse filter is singular, we have to use the generalized inverse filtering. People also suggest the power spectrum of the original image can be estimated based on a model such as the https://lh6.googleusercontent.com/dglYv9DMDXEHG0OQyXmjECKeeLa5fBrd7W7ZyWTCXCMbzQ8_-BZgFEtaEAb0sVKvYFwi33vs5-uGSKBBHQEbSpbBglfpVzggkXu--S6J6eAuV8I1COnKJxjTzOr_FuQPLOndoYS- model

**REGULARIZED FILTER:**

Use the [deconvreg](https://in.mathworks.com/help/images/ref/deconvreg.html) function to deblur an image using a regularized filter. A regularized filter can be used effectively when limited information is known about the additive noise.

Description

[example](https://in.mathworks.com/help/images/ref/deconvreg.html#bvighsz)

[J](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33254) = deconvreg([I](https://in.mathworks.com/help/images/ref/deconvreg.html" \l "d119e33070),[psf](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33109)) deconvolves image I using the regularized filter algorithm, returning deblurred image J. The assumption is that the image I was created by convolving a true image with a point-spread function (PSF), psf, and possibly by adding noise. The algorithm is a constrained optimum in the sense of least square error between the estimated and the true images under requirement of preserving image smoothness.

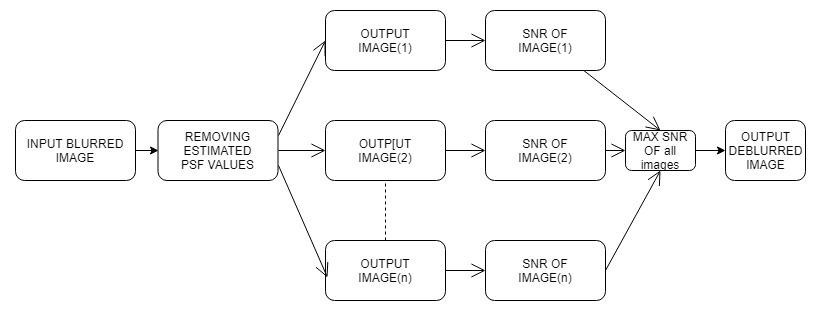
[J](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33254) = deconvreg([I](https://in.mathworks.com/help/images/ref/deconvreg.html" \l "d119e33070),[psf](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33109),[np](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33136)) specifies the additive noise power, np.

[J](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33254) = deconvreg([I](https://in.mathworks.com/help/images/ref/deconvreg.html" \l "d119e33070),[psf](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33109),[np](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33136),[lrange](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33167)) specifies the range, lrange, where the search for the optimal solution is performed. The algorithm finds an optimal Lagrange multiplier [lagra](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33281) within the lrange range.

[J](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33254) = deconvreg([I](https://in.mathworks.com/help/images/ref/deconvreg.html" \l "d119e33070),[psf](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33109),[np](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33136),[lrange](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33167),[regop](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33213)) constrains the deconvolution using regularization operator regop. The default regularization operator is the Laplacian operator, to retain the image smoothness.

[[J](https://in.mathworks.com/help/images/ref/deconvreg.html" \l "d119e33254),[lagra](https://in.mathworks.com/help/images/ref/deconvreg.html#d119e33281)] = deconvreg(\_\_\_) outputs the value of the Lagrange multiplier, lagra in addition to the restored image, J.

**5.1 ARCHITECTURE:**

****

**5.2 Processing Steps:**

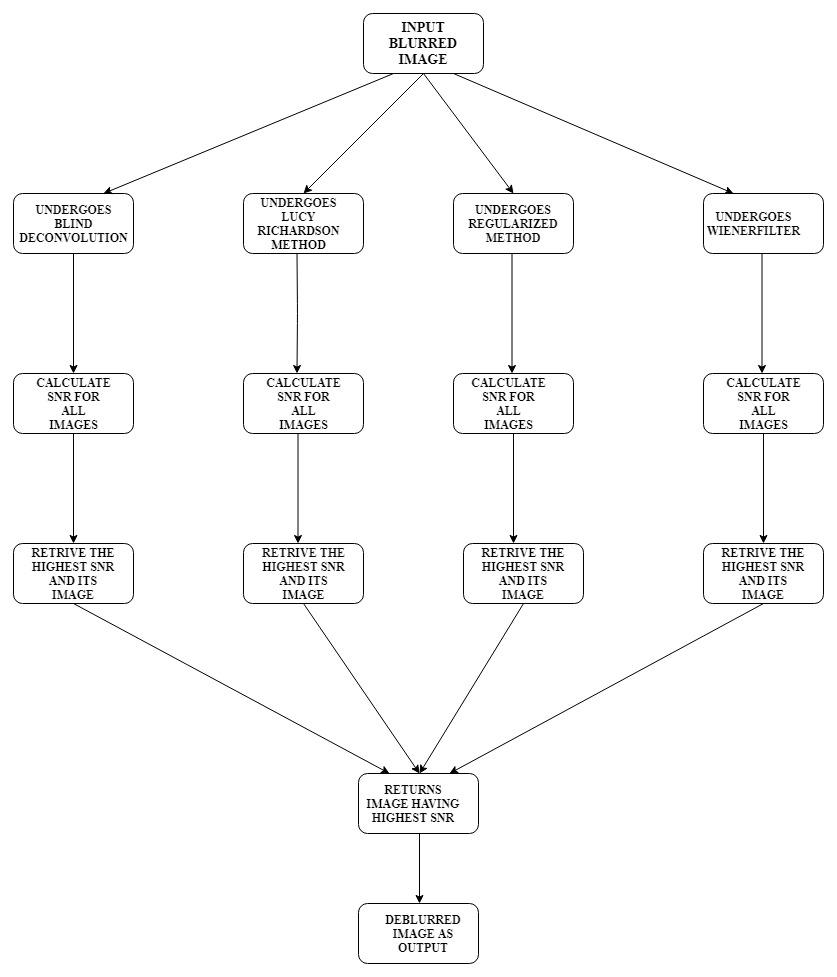
1. Read the blurred image as input through matlab and convert that image in matrix form for calculation.
2. After that estimate the PSF values which is generated randomly from given values.
3. Psf values form matrix and by comparing both image and psf values remove psf from the blurred image and generate multiple images by removing various psf from image.
4. Here multiple images are generated as output for those images we need to select one image as deblurred image.
5. After generating multiple images, we need to calculate SNR value for all the images that are generated.
6. Calculating SNR for all images then pick the image which having maximum SNR value is the Main image that is deblurred image and best of all images that are generated.
7. After completion Deblurred image is obtained as output.

**6.DESIGN**

**6.1 STRUCTURE CHART:**

Structure chart in software engineering and organizational theory, is a chart which shows the breakdown of a system to its lowest manageable levels. They are used in structured programming to arrange program modules in a tree. Each module is represented by a box, which contains the module’s name. A structure chart is top-down modular design tool, constructed of squares representing the different modules in the system, and lines that connect them. The lines represent the connection and or ownership between activities and sub-activities as they are used in organization chart.

According to Wolber (2009), "a structure chart can be developed starting with the creating of a structure, which places the root of an upside-down tree which forms the structure chart. The next step is to conceptualize the main sub-tasks that must be performed by the program to solve the problem. Next, the programmer focuses on each sub-task individually, and conceptualizes how each can be broken down into even smaller tasks. Eventually, the program is broken down to a point where the leaves of the tree represent simple methods that can be coded with just a few program statements".A structure chart is also used to [diagram](https://en.wikipedia.org/wiki/Diagram) associated elements that comprise a run stream or thread. It is often developed as a [hierarchical diagram](https://en.wikipedia.org/w/index.php?title=Hierarchical_diagram&action=edit&redlink=1), but other representations are allowable. The representation must describe the breakdown of the [configuration system](https://en.wikipedia.org/wiki/Configuration_system) into [subsystems](https://en.wikipedia.org/wiki/Subsystem) and the lowest manageable level. An accurate and complete structure chart is the key to the determination of the configuration items (CI), and a visual representation of the configuration system and the internal interfaces among its CI’s.

**Structure chart of this project: **

**Description of Modules:**

**6.1.1 I=imread('image.tif');**

It reads the image we want to deblur.

**6.1.2 figure,imshow(I);**

It shows the image we have just read to deblur it.

**6.1.3 PSF = fspecial('motion',l,t);**

It creates the PSF to the image of motion type with length and theta.

**6.1.4 INITPSF = ones(size(PSF));**

It will initiates the all the PSF values with ones.

**6.1.5** **[J, P] = deconvblind(I,PSF,3);**

This function applies blind deconvolution technique to the image.

**6.1.6 J = deconvlucy(I,INITPSF,100);**

This function applies lucy richardsontechnique to the image.

**6.1.7 [J, P] = deconvreg(I,INITPSF,2);**

This function applies deconvolution blind technique to the image

**6.1.8 [snrL, snrK] = psnr(I,J);**

It use to calculate snr of the images. Above function gives snr of I compared to j.

**6.2 UML DIAGRAMS:**

The UML is a language. It provides vocabulary and the results for combining words in that vocabulary for communication. A modelling language is language whose vocabulary and rules flows on the conceptual and physical representation of a system. A modelling language such as UML is a standard language for software blue prints.

The UML is a language for visualizing, specifying, constructing and documenting. The software intensive articrafts of a system.

UML diagram are classified into two categories:

1. Structural or static

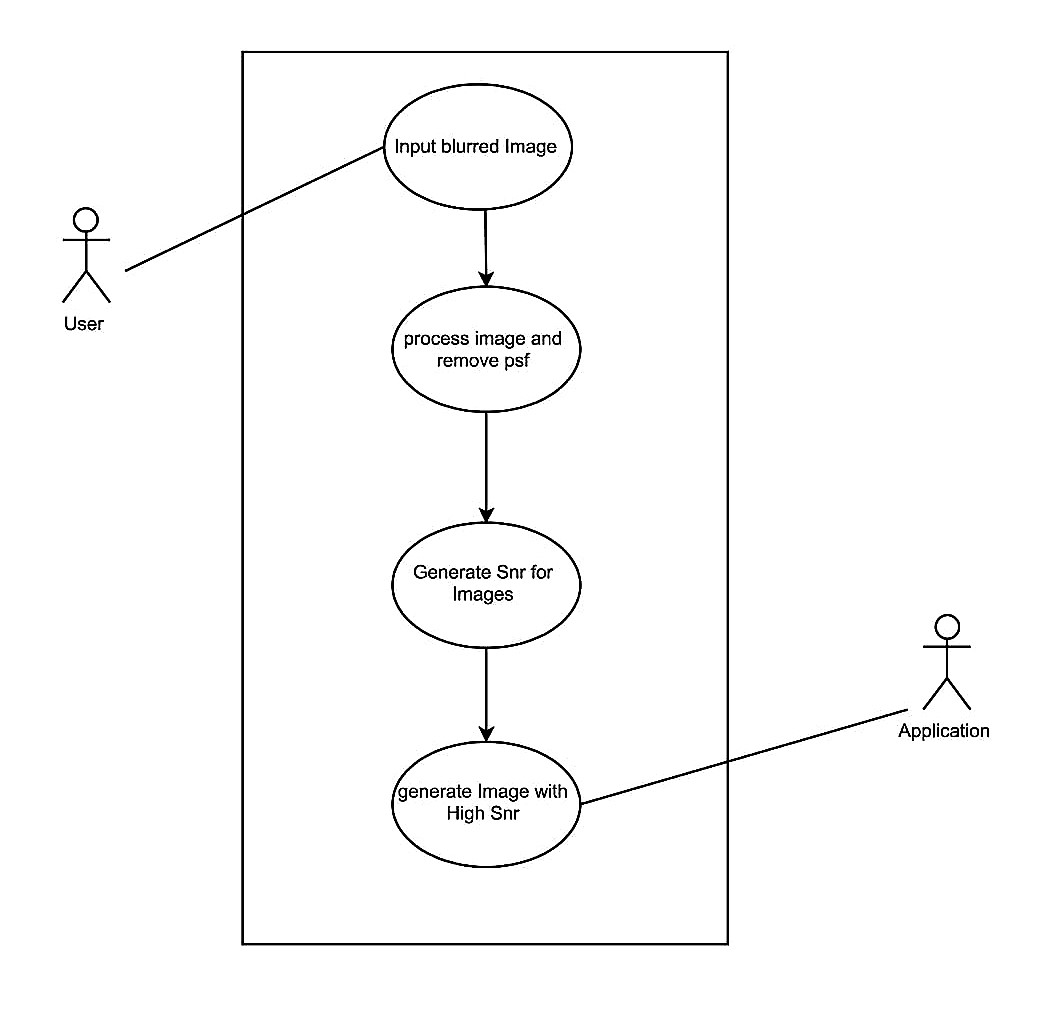
2. Dynamic or behavioural

Structural Model contains

Classes, object, use case, component and deployment.

Behavioural Model contains:

Collaboration, State chart and activity.



**6.2.1 Sequence Diagram:**

Interaction diagram is called is sequence diagram. Interaction diagram describes patterns of communication among a set of interaction objects. An object interacts with another object by sending messages, arguments may be passed along with a message and they are found to be parameters of executing methods in the receiving objects.

**In this Diagram, the Objects are:**

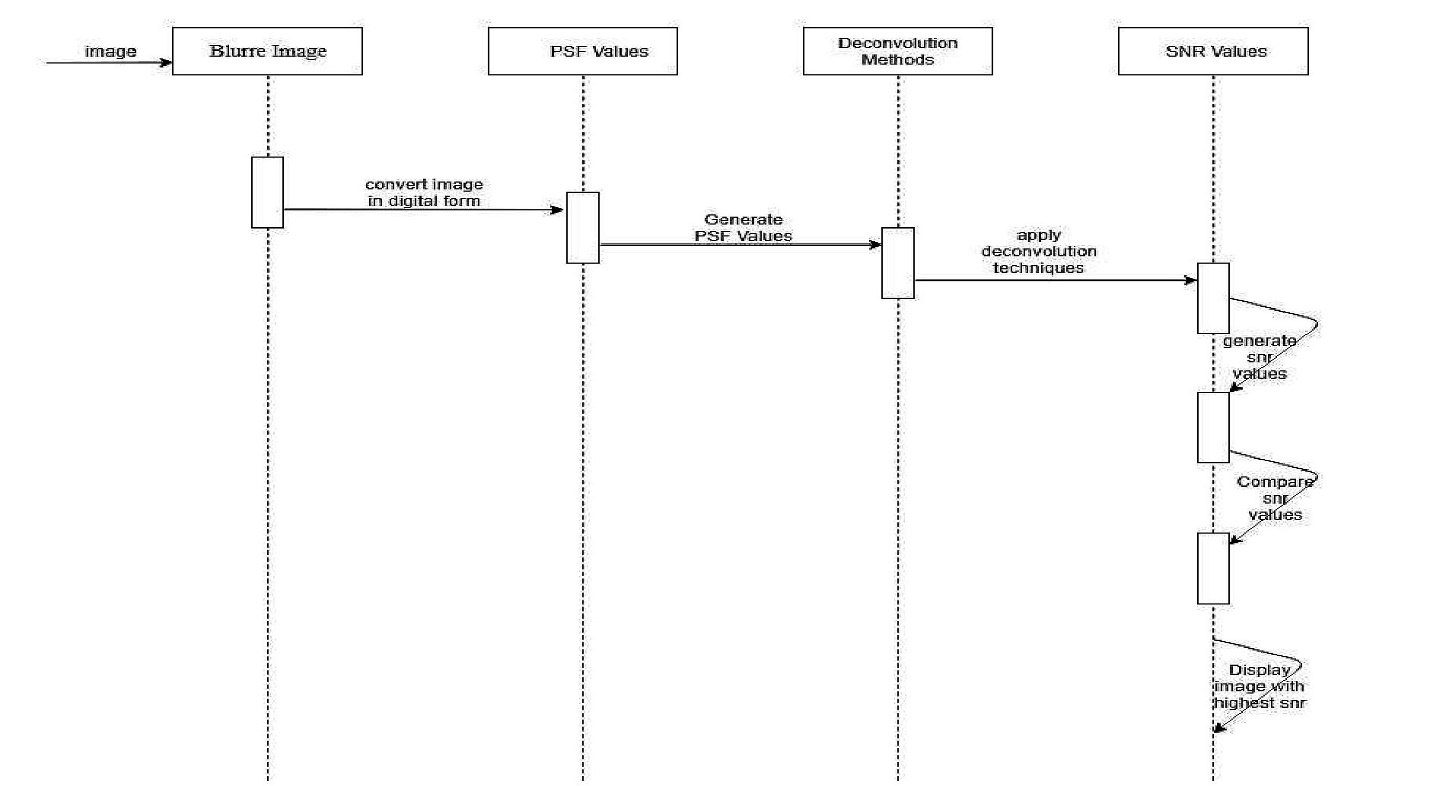
1.Blur image

2.PSF values

3.Deconvolution Methods

4.SNR values

**In this Diagram, the Pre-Defined Functions are:**

* Converting image into digital form.
* Appling deconvolution functions .
* Generating SNR values.

**6.2.2Activity Diagram:**

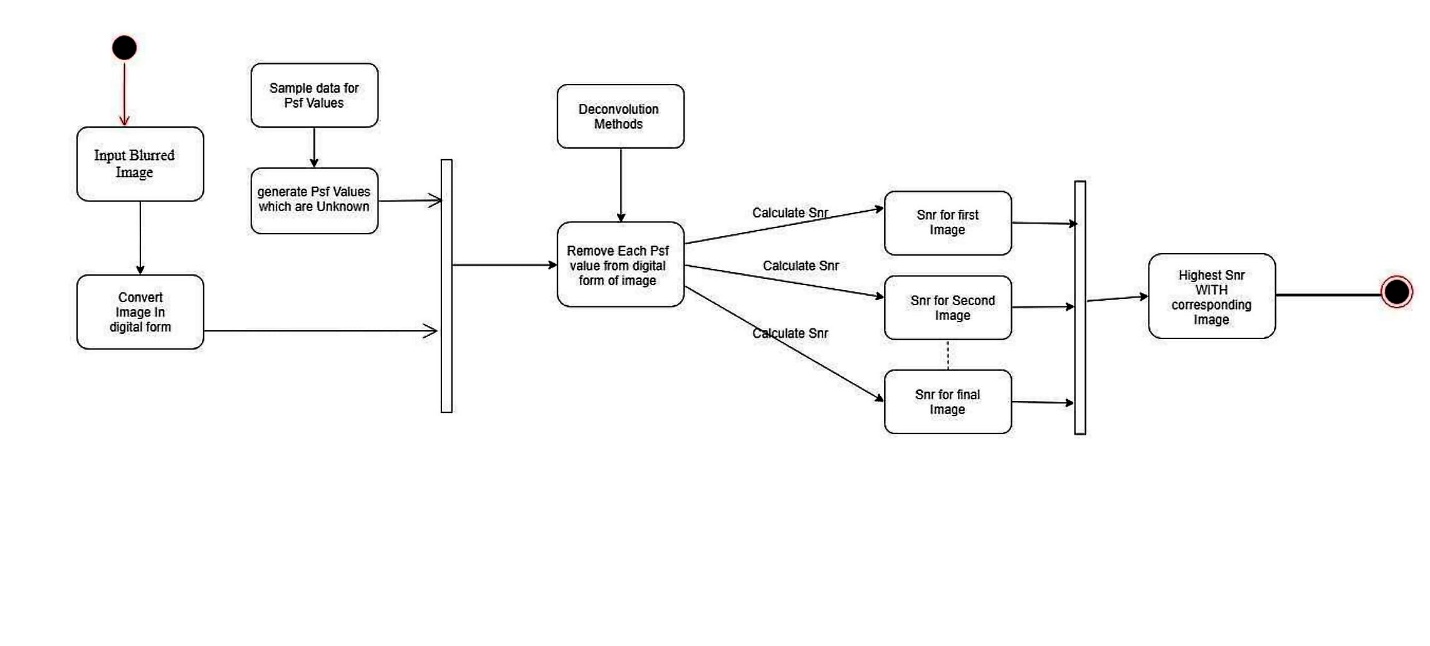
An Activity diagram describes the work flow in steps between activities and actions with support for interaction and concurrency behaviour of the system. These are similar to state diagram because activities of the state of doing something. These can show activities, that are conditional or parallel.The Objects identified are Converting image into digital form,Appling deconvolution functions ,Generating SNR values.

**Components:**

1.Blur image

2.PSF values

3.Deconvolution Methods

4.SNR values

**6.2.3 Component Diagram:**

Component diagram in uml depicts how components are wired together to form longer components and software systems. The components are wired together while using an assembly connector to connect the required.

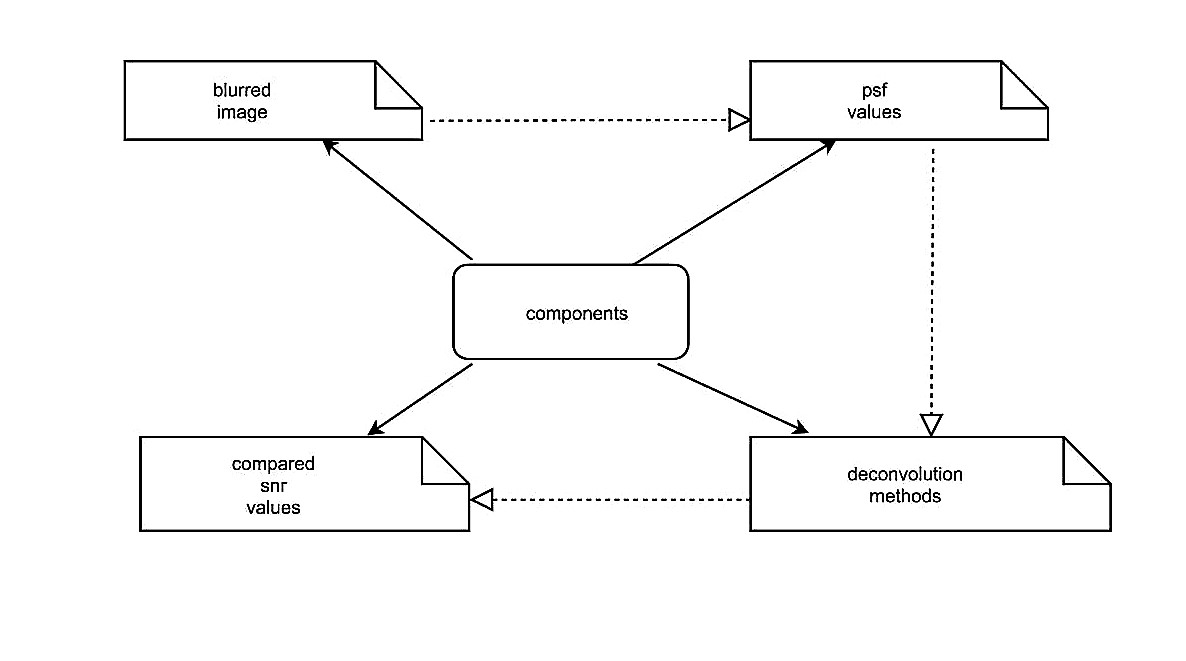
An assembly connector is the connector between two components, provides the service that another component requires.

**Components are:**

1.Blur image

2.PSF values

3.Deconvolution Methods

4.SNR values****

**6.2.4 Deployment diagram:**

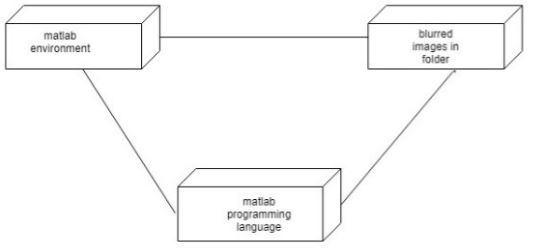
It provides different perspective of the application. It captures the configuration of runtime elements of the application. This diagram is more useful when a system is build and ready to be deployed.

**NODES ARE:**

1.Matlab Environment

2.Blurred images in folder

3.Matlab Programming Language



**7.METHODOLOGIES**

In this project, we follow these steps to deblur the image

1. Open the image in matlab.
2. Perform all deconvolution techniques.
3. Retrieve the highest SNR image of the correspond techniques.
4. Retrieve the image with highest SNR image of all the technique.

**7.1 Comparing:**

Here we generating different images under different techniques a lot of images are produces with corresponding SNR. So comparing between them is the major issue. So we create arrays for the snr created and store them the array and compare between them and the highest snr value image similar we done to all images generated by different techniques all the highest snr are kept in a separate array and highest snr value is consider and that corresponding image is retrieved and it the output image of our project.

**7.2** **Estimating PSF:**

For applying deconvolution technique we are creating PSF to a image(Point Spread Function) in matlab.  
fspecial Create predefined 2-D filters.

H = fspecial(TYPE) creates a two-dimensional filter H of the specified type. Possible values for TYPE are:

'average' averaging filter  
 'disk' circular averaging filter

Here psf is the difficult task here we consider that blur is caused by the motion between the subject and the camera so we create the motion psf with len and thetha.

H = fspecial('motion',LEN,THETA) returns a filter to approximate, Once convolved with an image, the linear motion of a camera by LEN pixels, with an angle of THETA degrees in a counter-clockwise direction. The filter becomes a vector for horizontal and vertical motions. The default LEN is 9, the default THETA is 0, which corresponds to a horizontal motion of 9 pixels.

**7.3 Estimating SNR:**

**Signal-to-noise ratio** (abbreviated **SNR** or ***S/N***) is a measure used in [science and engineering](https://en.wikipedia.org/wiki/Science_and_engineering) that compares the level of a desired [signal](https://en.wikipedia.org/wiki/Signal_(electrical_engineering)) to the level of background [noise](https://en.wikipedia.org/wiki/Noise_(signal_processing)).

SNR is defined as the ratio of signal power to the noise power, often expressed in [decibels](https://en.wikipedia.org/wiki/Decibel). A ratio higher than 1:1 (greater than 0 dB) indicates more signal than noise.

While SNR is commonly quoted for electrical signals, it can be applied to any form of signal, such as [isotope](https://en.wikipedia.org/wiki/Isotope) levels in an [ice core](https://en.wikipedia.org/wiki/Ice_core), [biochemical signaling](https://en.wikipedia.org/wiki/Biochemical_signaling) between cells, or financial trading signals.

[r](https://in.mathworks.com/help/signal/ref/snr.html#bt2g8x3-r) = snr([x](https://in.mathworks.com/help/signal/ref/snr.html" \l "bt2g8x3-x),[y](https://in.mathworks.com/help/signal/ref/snr.html#bt2g8x3-y)) returns the signal-to-noise ratio (SNR) in decibels of a signal, x, by computing the ratio of its summed squared magnitude to that of the noise, y. y must have the same dimensions as x. Use this form when the input signal is not necessarily sinusoidal and you have an estimate of the noise.

**8.IMPLEMENTATION**

* 1. **Sample Code if the blur information is unknown:**

%reading image and appling blind deconvolution

y= imread('aravind1.tif');

I=im2double(y);

z=1;

str='dfigure';

[snr1, snr2] = psnr(y,y);

fprintf('\n The SNR of blurred image value is %0.4f', snr2);

figure

imshow(I)

title('blurredimage');

xx=1; big1=-1;big5=-1;big2=-1;big3=-1;big4=-1;

for l=25.0:-5.3:0.0

for t=25.0:-5.2:0.0

PSF = fspecial('motion',l,t);

INITPSF = ones(size(PSF));

[J, P] = deconvblind(I,PSF,3);

figure

imshow(J)

title([str ':' num2str(z)])

[snr4, snr3] = psnr(I,J);

snr3\_arr(xx)=snr3;

xx=xx+1;

z=z+1;

end

end

[val1, ind1] = max(snr3\_arr(:));

if ind1 > big1

new\_img1 = J;

big1=ind1;

end

%Appling Weiner deconvolution

yy=1;

for l=18.0:-6.3:0.0

for t=15.0:-6.2:0.0

PSF = fspecial('motion',l,t);

INITPSF = ones(size(PSF));

J = deconvwnr(I,INITPSF,0);

title([str ':' num2str(z)])

[snr6,snr5] = psnr(I,J);

snr5\_arr(yy)=snr5;

yy=yy+1;

z=z+1;

end

end

[val2, ind2] = max(snr5\_arr(:));

if ind2 > big2

new\_img2 = J;

big2=ind2;

end

%Appling Regular Deconvolution

zz=1;

for l=18.0:-6.3:0.0

for t=15.0:-6.2:0.0

PSF = fspecial('motion',l,t);

INITPSF = ones(size(PSF));

[J, P] = deconvreg(I,INITPSF,2);

%figure

%imshow(J)

title([str ':' num2str(z)])

[snr8, snr7] = psnr(I,J);

snr7\_arr(zz)=snr7;

zz=zz+1;

z=z+1;

end

end

[val3, ind3] = max(snr7\_arr(:));

if ind3 > big3

new\_img3 = J;

big3=ind3;

end

%Appling Lucy Richardson deconvultion

aa=1;

for l=18.0:-6.3:0.0

for t=15.0:-6.2:0.0

PSF = fspecial('motion',l,t);

INITPSF = ones(size(PSF));

J = deconvlucy(I,INITPSF,100);

%figure

%imshow(J)

%title([str ':' num2str(z)])

[snr10, snr9] = psnr(I,J);

snr9\_arr(aa)=snr9;

aa=aa+1;

z=z+1;

end

end

[val4, ind4] = max(snr9\_arr(:));

if ind4 > big4

new\_img4 = J;

big4=ind4;

end

dz=[val1 ,val2 , val3 , val4];

[rrr, sss] = max(dz(:));

if rrr == val1

new\_img5 = new\_img1;

figure, imshow(new\_img5);

title(['dfigure:' num2str(ind1), 'snr' [num2str(rrr)]])

elseif rrr==val2

new\_imag5=new\_img2;

figure, imshow(new\_img5);

title(['dfigure:' num2str(ind2), 'snr' [num2str(rrr)]])

elseif rrr==val3

new\_img5=new\_img3;

figure, imshow(new\_img5);

title(['dfigure:' num2str(ind3), 'snr' [num2str(rrr)]])

else

new\_img5=new\_img4;

figure, imshow(new\_img5);

title(['dfigure:' num2str(ind4), 'snr' [num2str(rrr)]])

end

* 1. **Sample code if blur is know:**

I = im2double((imread('download.jpg')));

len=20;

theta=25;

psf = fspecial('motion',len,theta);

blurred = imfilter(I,psf,'conv','circular');

figure,imshow(blurred);

title('blurred image');

recovered=deconvblind(blurred,psf,12);

figure,imshow(recovered);

title('deconvoblind');

X= deconvreg(blurred,psf,2);

figure,imshow(X);

title('deconvreg');

J = deconvlucy(blurred,psf,100);

figure,imshow(J);

title('deconvlucy');

S**ample Input:**

Blurred image with unkown blur information

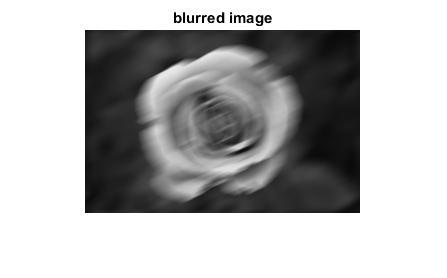
**Sample Output:**

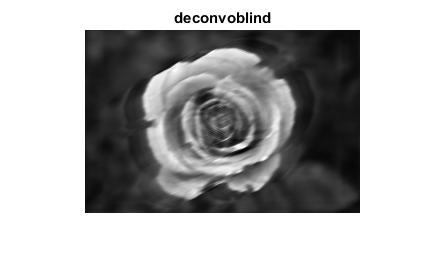
Deblurred image with calculated SNR value.

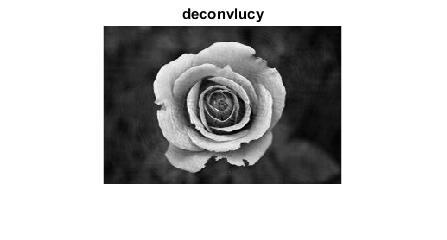
**8.RESULTS**

**9.1 Deblurring of images whoes blur information is known:**

Blurred image which we have taken to deblur.





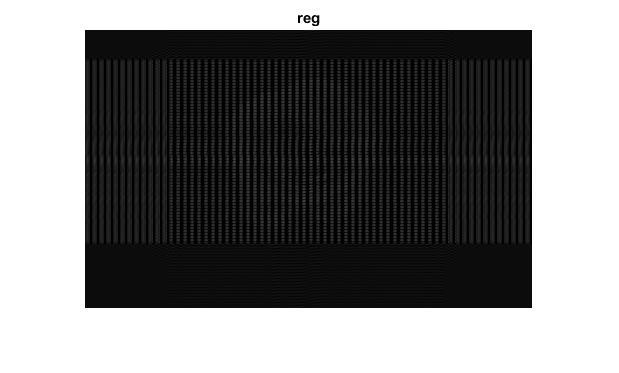




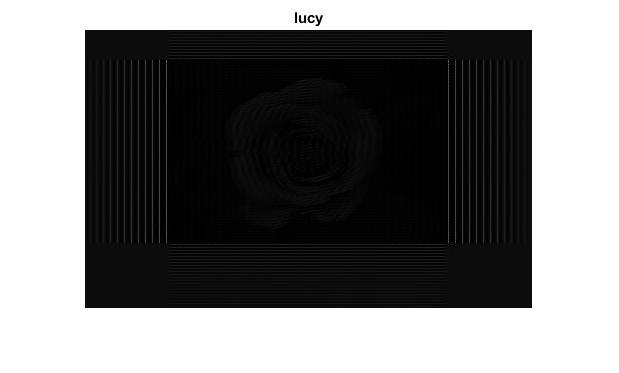
**9.2 deblurring of images whoes blur information is unknown:**

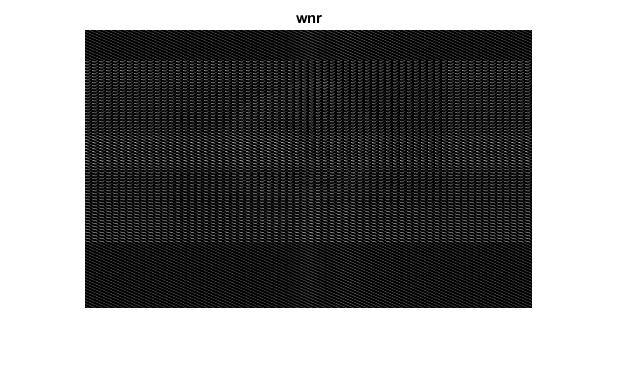


Best image genrated by blind when blur information is not known.



Best image genrated by regular filter when blur information is unknown.





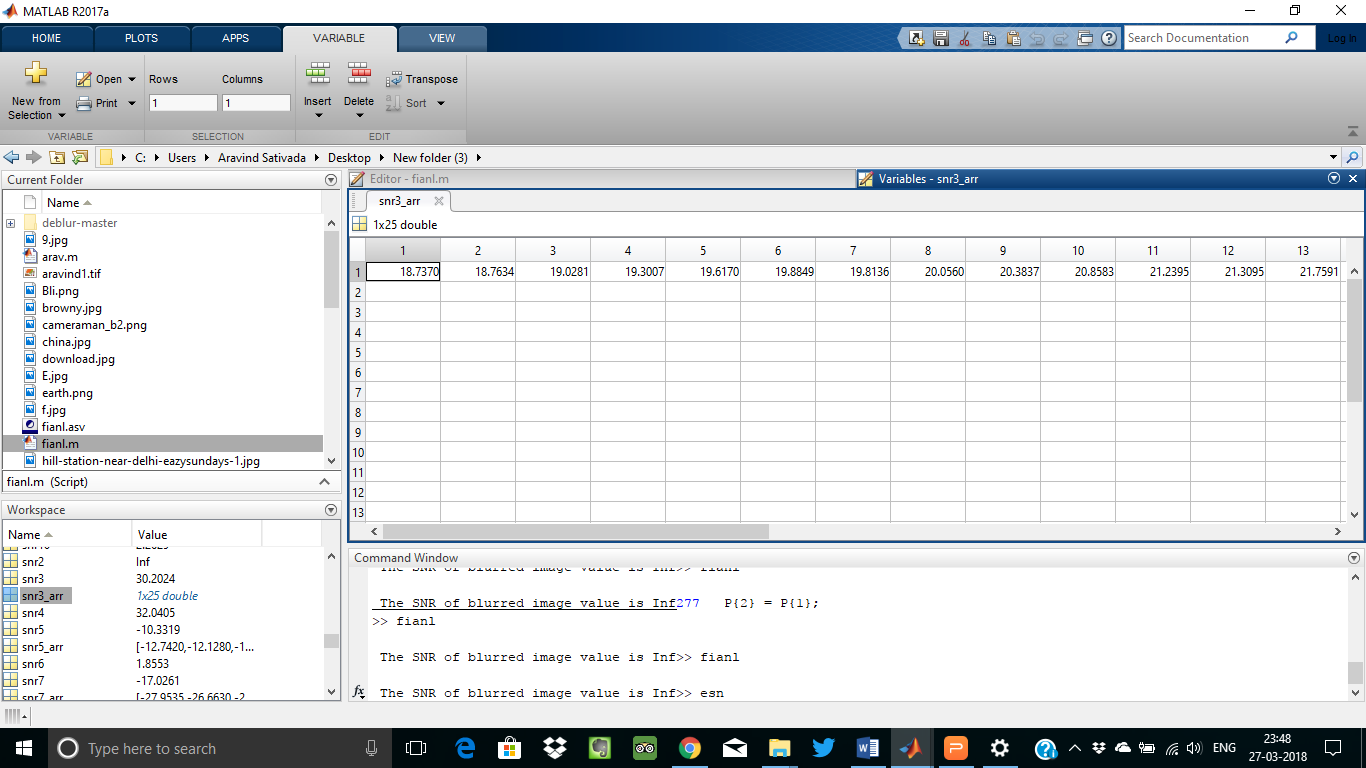
Best image generated by weiner when blur information is not known.

Best image generated by lucy and blur information is not known.

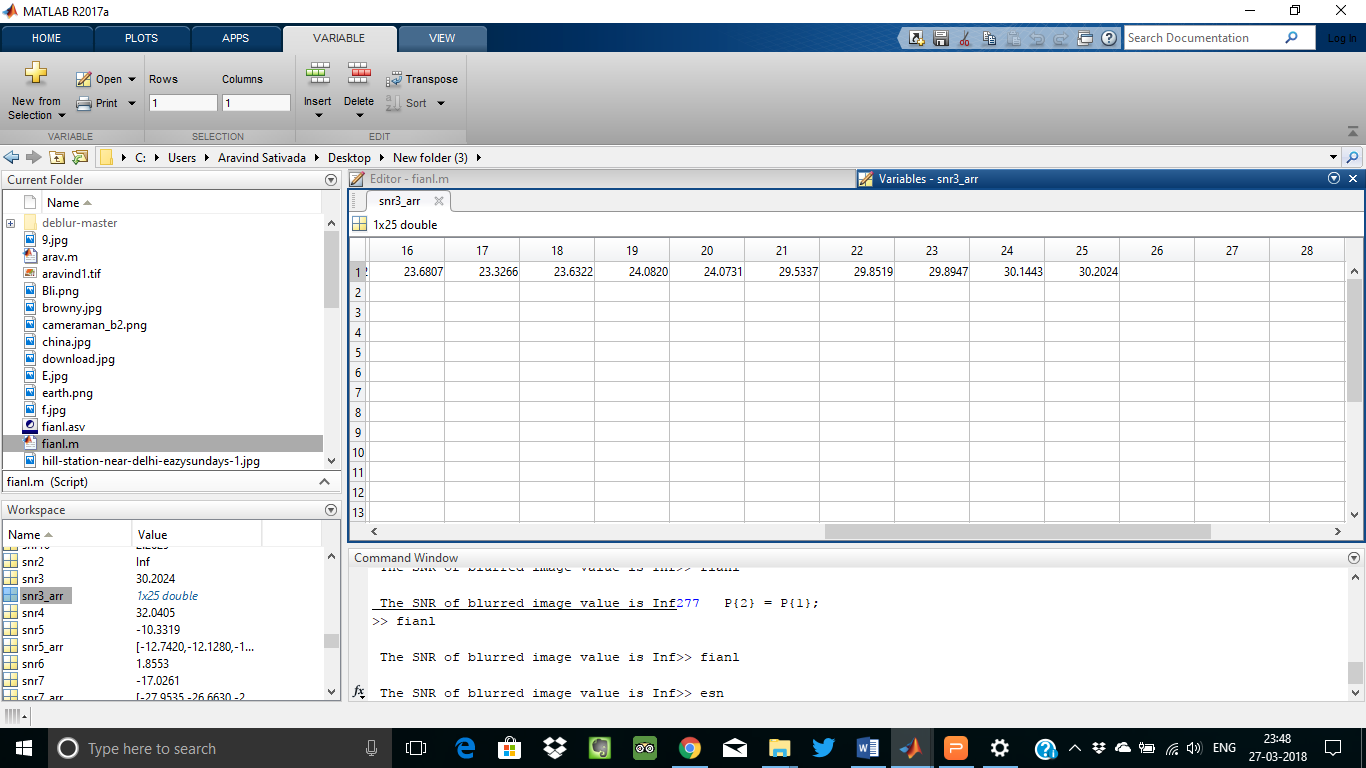
**10.TESTING**

It is a way giving inputs to the system to find the difference between observed behavior and specified behavior of an application /software system.In this we are testing we are comparing our results based on other techniques we are conformed my results are perfect.

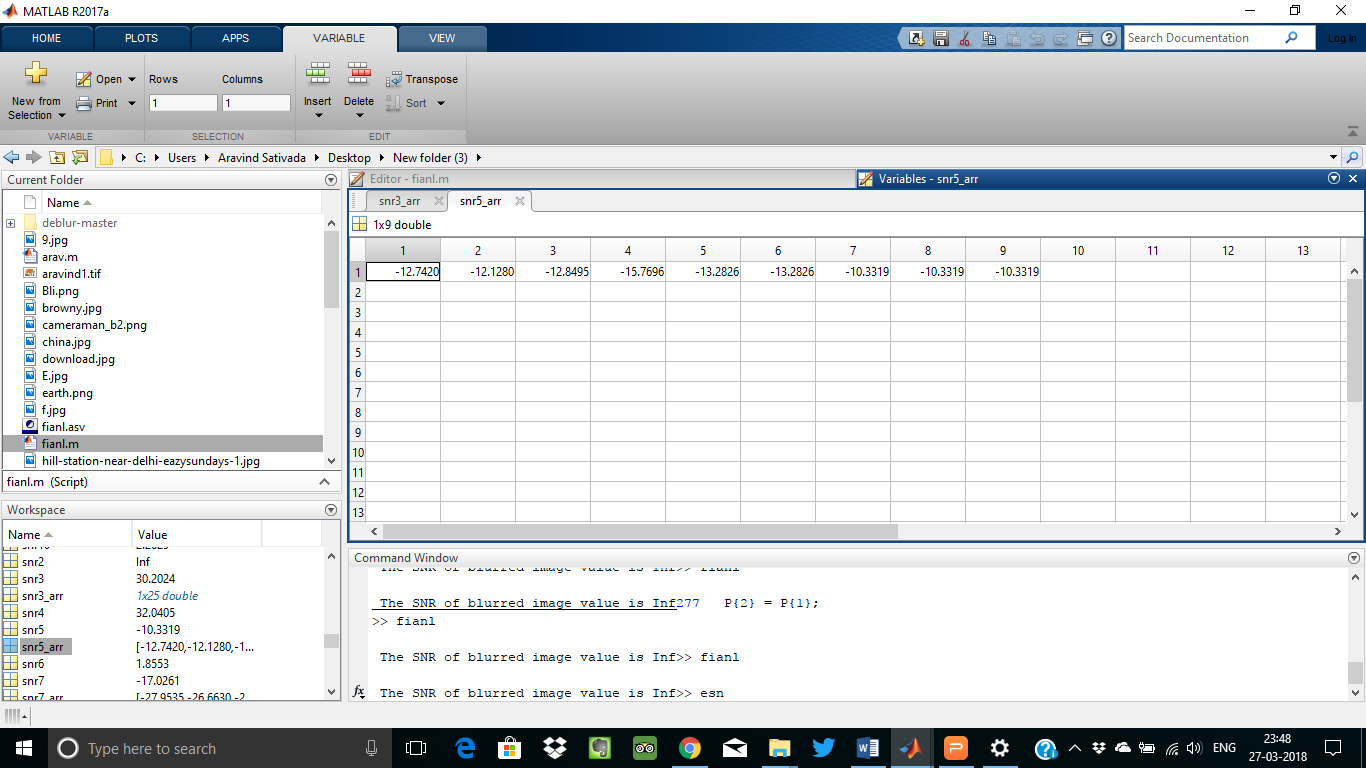
**10.1 Testing for SNR values:**

****

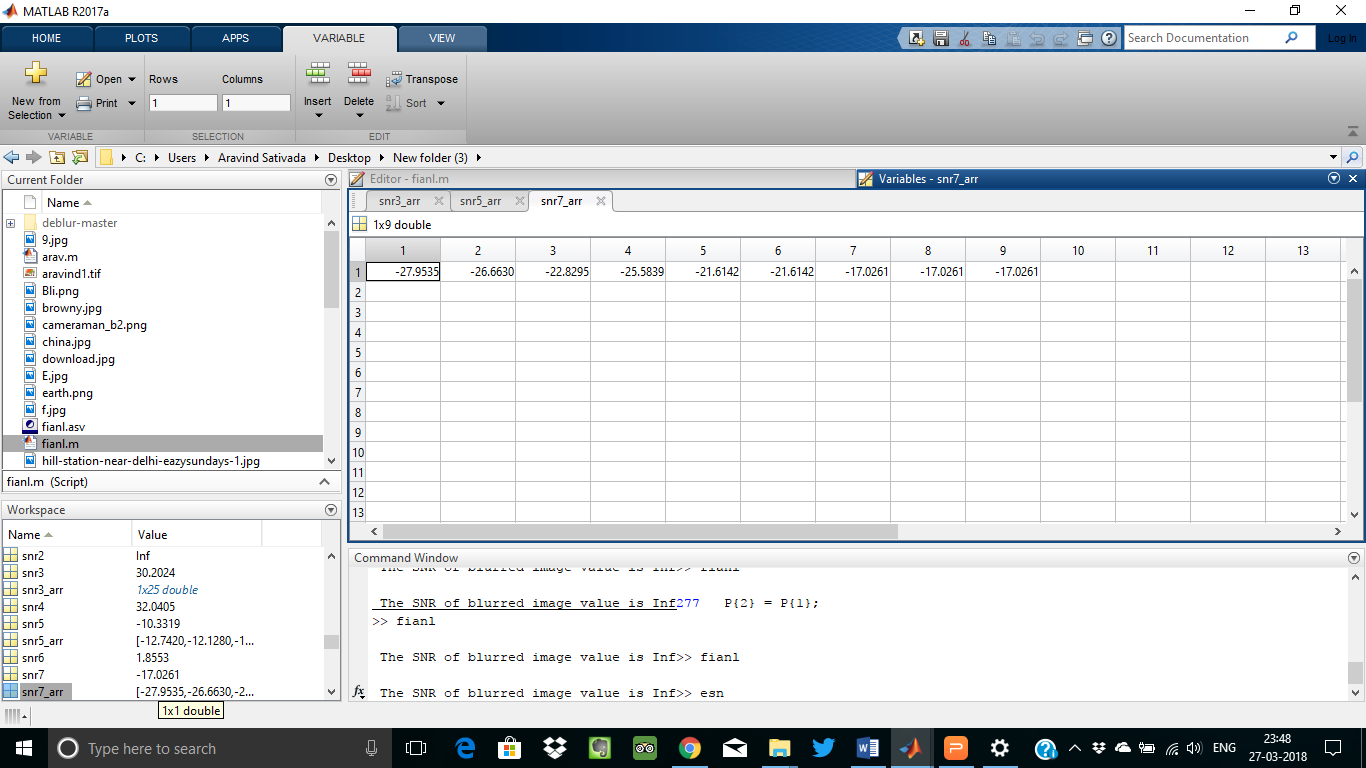
Snr values generated by the blind deconvolution technique are seen here and highest snr values is taken or not and that corresponding image is retrived or not is checked here.

****

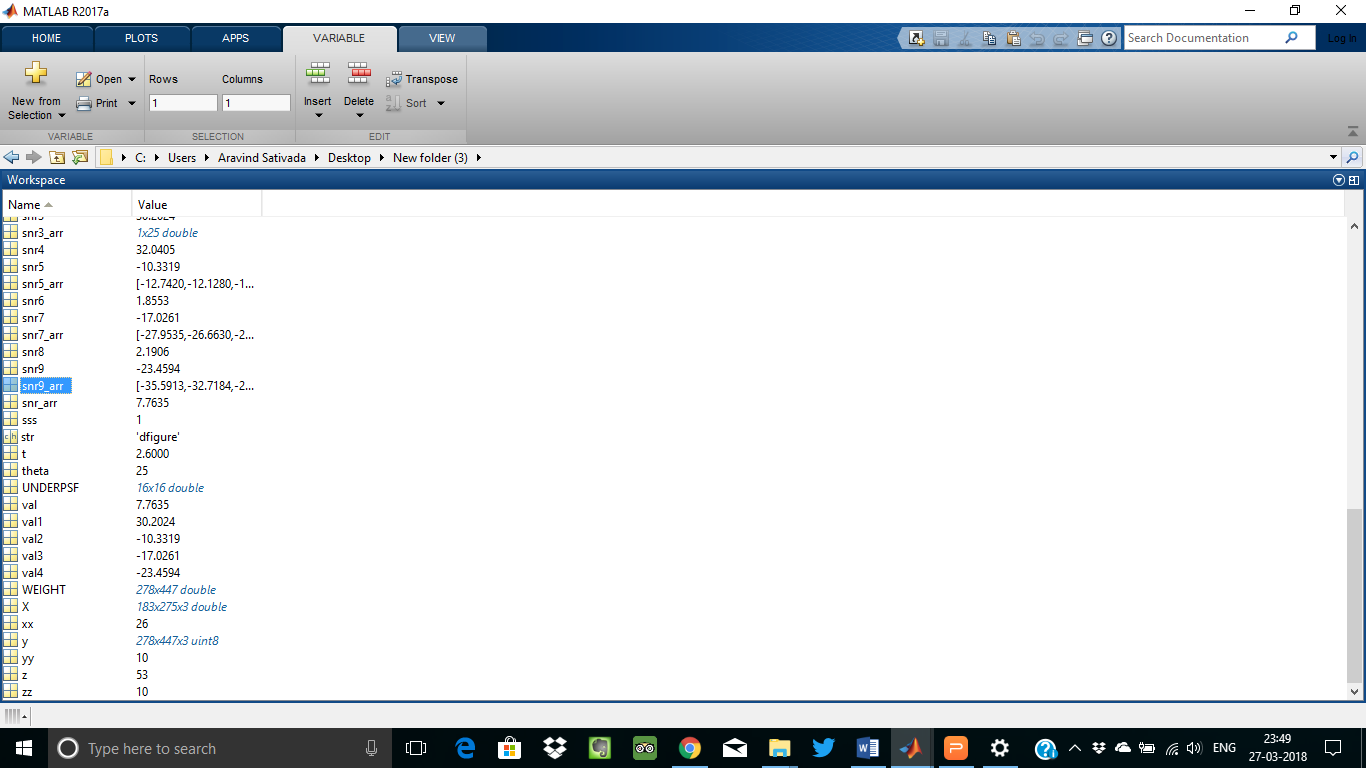
Snr values generated by the wiener filter techniques

****

Snr values generated by the Regularized filter techniques

****

Snr values generated by the Lucy Richardson techniques

****

Testing whether correct Snr values generated for corresponding images

1. **CONCLUSION**

In this, we developed one of the best process for generating deblurred image from blurred image.It is different from previous techniques here we improve all the

issues that are present in earlier system by using PSF and SNR values.Previous deblurring image techniques not work for all images it works with some limitations.

That is not happened in Our technique it works for all the blurred images.For this we use blind deconvolution technique as base for generating deblrred image from blurred image successfully.

**REFERENCES**

**WEB SITES:**

1. <https://in.mathworks.com/help/images/examples/deblurring-images-using-the-blind-deconvolution-algorithm.html>
2. <https://en.wikipedia.org/wiki/Blind_deconvolution>
3. http://ieeexplore.ieee.org/document/5963691/
4. https://en.wikipedia.org/wiki/Point\_spread\_function
5. https://en.wikipedia.org/wiki/Signal-to-noise\_ratio\_(imaging)

**BOOKS:**

* Digital image processing Using Matlab By Rafael C. Gonzalez,Richard E. Woods.
* Image Processing: The Fundamentals By Maria Petrou,Costas Petrou
* Principles of Digital Image Processing: Core Algorithms (Undergraduate Topics in Computer Science) By Wilhelm Burger, Mark J. Burge