IMAGE DEBLURRING

A Seminar Report
submitted by:

Kalicharan
(22MA60R37)

in partial fulfillment for awarded of degree

of

MASTER OF TECHNOLOGY

in

Computer Science and Data processing
at

DEPARTMENT OF MATHEMATICS

INDIAN INSTITUTE OF TECHNOLOGY,

KHARAGPUR
WEST BENGAL-721302,INDIA

Contents

ABSTRACT

In this study we dealt with the blur caused by motion in which the cameras follow out of plane translatory motion. Solving the deblurring problem of motion blur has significant importance and applications in forensic science, analysis of astronomical images gives clues to the behavior of the universe computer vision and video processing where the restoration of the features in the given image or video sequence is mandatory for the preservation of essential details which are normally blurred or fade out due to the blur caused by motion. At a more mundane level, barcode readers used in supermarkets and by shipping companies must be able to compensate for imperfections in the scanner optics.

The study formulates an effective deblurring algorithm for feature restoration and eliminating the effects of motion blur using decomposable pixel component analysis. Furthermore, the existingmethods are also analyzed and compared in the study. The algorithm is verified and testified with the several natural deblurred images which weren't as efficient as those of previous ones.

INTRODUCTION

Images are widely used in many kinds of applications such as everyday photography, monitoring, medical imaging, astronomy, microscopy, and remote sensing. Digital images are composed of picture elements or pixels that are organized in a grid. Each pixel contains an intensity value which determines the tone at a specific point. Unfortunately, all captured images end up more or less blurry. The motion of objects or the vibration of the sensor (camera) when pressing the shutter causes the image to be blurred. There are many factors that cause the blurring or degradation of the digital image, such as movement during the capture process, using long exposure times, using wide angle lens, etc. However, there are two main causes for motion blur: (*i*) the image is blurred by the camera vibration which causes all pixels in the image to be affected, and (*ii*) the image is blurred by object motion which causes a specific region to be blurred. Image blur usually devastate the images, and practically it is hard to avoid it because there is a lot of interference in the environment. Image deblurring is the process of applying and solving mathematical models to recover the original (sharp) image. Recently, image deblurring become the main and the most important sub field of digital image processing. Image deblurring is used to make images sharp and useful. There are many applications for image deblurring such as: Recovering valuable photographs, Watching distant star fields through ground based telescopes. ,Watching space vehicles and satellites, Radar imaging, Tomography and medical imaging, Microscopy, Iris recognition Mathematically, the blurring can be defined as a linear process that leads to image degradation.

Chapter 1

Preliminaries

1.1 Spectral decomposition

Eigenvalues and eigenvector A vector $\mathbf{v} \in \mathbb{R}^n,\ \mathbf{v} \neq 0$ is an eigenvector of square matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, if it satisfies a linear equation of the form

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$$

for some $\lambda \in \mathbb{R}$.Then λ is called eigenvalue corresponding to v.This equation called the eigenvalue equation.
And for the eigenvalue, equation

$$\chi(\lambda) = det(\mathbf{A} - \lambda\mathbf{I})$$

called the characteristic equation and $\chi(\lambda)$ called **characteristic polynomial**.Solutions of $\chi(\lambda)$ is the eigenvalues of matrix $\mathbf{A}$.

Let $\mathbf{A}$ be a diagonalizable square $n \times n$ matrix with n linearly independent eigenvectors $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n$. Then $\mathbf{A}$ can be factorized as

$$\mathbf{A} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{-1}$$

where $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n]$ is the square $n \times n$ matrix whose i^{th} column is the eigenvector $\mathbf{q}_i$ of $\mathbf{A}$, and $\mathbf{\Lambda} = diag(\lambda_1, \lambda_2, \ldots, \lambda_n)$ is the diagonal matrix whose diagonal elements are the corresponding eigenvalues.This decomposition is called **Eigendecomposition of a matrix** .When the matrix is a normal or real symmetric matrix, the decomposition is called **"spectral decomposition"**.

Normal matrices A complex-valued square matrix $\mathbf{A}$ is normal ($i.e\mathbf{A}^*\mathbf{A} = \mathbf{A}\mathbf{A}^*$, where A^* is the conjugate transpose) if and only if it can be decomposed as

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^*$$

where $\mathbf{U}$ is a unitary matrix ($i.e.\mathbf{U}^* = \mathbf{U}^{-1}$) and $\Lambda = diag(\lambda_1, \lambda_2, \ldots, \lambda_n)$ is a diagonal matrix. The columns $u_1, ..., u_n$ of U form an orthonormal basis and are eigenvectors of A with corresponding eigenvalues $\lambda_1, \lambda_2, ..., \lambda_n$.

If $\mathbf{A}$ is restricted to be a Hermitian matrix ($A = A^*$), then Λ has only real valued entries. If A is restricted to a unitary matrix, then Λ takes all its values on the complex unit circle, that is, $|\lambda_i| = 1$.

Real symmetric matrices If $\mathbf{A}$ is real symmetric matrix ($i.eA^T$ then eigenvalues of $\mathbf{A}$ are real and corresponding eigenvectors be chosen orthonormal and a real symmetric matrix $\mathbf{A}$ can be decomposed as

$$\mathbf{A} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^T$$

where $\mathbf{Q}$ is an orthogonal matrix whose columns are eigenvectors of $\mathbf{A}$, and $\boldsymbol{\Lambda}$ is a diagonal matrix whose entries are the eigenvalues of $\mathbf{A}$.

1.2 Singular value decomposition(SVD)

Singular value decomposition is the very important factorization.It has important consequences in the practical application, for example In image compression, image restoration,a biomedical application(Extracting fetal ECG from Maternal FCG) and other. Some application related to image processing, we shall discuss in coming chapter.

Definition 1. *Let $\mathbf{A}$ be the matrix of order $m \times n$. the decomposition $\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$ is called singular value decomposition of $\mathbf{A}$.*
Where $\mathbf{U} \in \mathbb{R}^{m\times m}$ and $\mathbf{V} \in \mathbb{R}^{n\times n}$ are the orthogonal matrices, whose columns called **left singular vectors** *and* **right singular vectors** *respectively. and $\boldsymbol{\Sigma} \in \mathbb{R}^{m\times n}$ is the diagonal matrix whose diagonal entries are called the* **singular values** *of matrix $\mathbf{A}$.*

Theorem 1. *(***SVD** *theorem)*
Let $\mathbf{A} \in \mathbb{R}^{m\times n}$. Then there exit orthogonal matrices $\mathbf{U} \in \mathbb{R}^{m\times m}$ and $\mathbf{V} \in \mathbb{R}^{n\times n}$ such that

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$$

where $\boldsymbol{\Sigma} = diag(\sigma_1, \sigma_2, \ldots, \sigma_p)$, $p = min\{m, n\}$ and $\sigma_1 \geq \sigma_2 \geq \ldots, \sigma_p \geq 0$.

Note: when $m \geq n$, then In SVD, $\mathbf{\Sigma}$ has form

$$\mathbf{\Sigma} = \left[\begin{array}{cccc} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \\ \hline & & \mathcal{O}_{(m-n)\times n} & \end{array}\right]_{m\times n}$$

Example: $m = 4$ and $n = 2$, then following is the an **SVD** of 4×2 matrix

$$\begin{bmatrix} * & * \\ * & * \\ * & * \\ * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} * & 0 \\ 0 & * \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} * & * \\ * & * \end{bmatrix}$$

Important points:

1. When $m \geq n$, we have n singular values. It can be shown that these are the square roots of the n eigenvalues of the matrix AA^T.

2. The singular values of the A are uniquely determined while the matrices U and V are not unique in general.

3. The SVD immediately reveals several matrix properties, including rank, norm, condition number and important information of the matrix, such as orthonormal bases of $Range\ space\ of A, R(A)$ and $Null\ space\ of\ A, N(A)$ and orthogonal Projection onto $R(A)$ and $N(A)$.

1.3 Reduced SVD

If $A = U\Sigma V^T$ is the SVD of $A \in \mathbb{R}^{m\times n}(m \geq n)$. Then we can write

$$A = U_1\Sigma_1 V^T$$

where

$$U_1 = (u_1, u_2, \ldots, u_n) \in \mathbb{R}^{m\times n}, V = (v_1, v_2, \ldots, v_n) \in \mathbb{R}^{n\times n}$$

and

$$diag(\sigma_1, \sigma_2, \ldots, \sigma_n) \in \mathbb{R}^{n\times n}$$

i.e

$$A = \begin{bmatrix} u_1 & u_2 & \dots & u_n \end{bmatrix} \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{bmatrix} \begin{bmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_n^T \end{bmatrix} = \sum_{i=1}^{n} \sigma_i u_i v_i^T.$$

Example:

Let consider a matrix Consider a matrix

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

eigenvalues $\lambda = 0, 0, 0, 0$ all zero !
only one eigenvector $(1, 0, 0, 0)$
singular values $\sigma = 3, 2, 1$
singular vectors are columns of $\mathbf{I}$
Start with $\mathbf{AA^T}$ and $\mathbf{A^TA}$. these are Diagonal matirx

$$\mathbf{AA^T} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 9 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \; and \; \mathbf{A^TA} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 9 \end{bmatrix}$$

Their eigenvectors ($\mathbf{u}'s\ for\ \mathbf{AA^T}$ *and* $\mathbf{v}'s\ for\ \mathbf{A^TA}$) go in decreasing order $\sigma_1^2 > \sigma_2^2 > \sigma_3^2$ of the eigenvalues. These eigenvalues $\sigma^2 = 9, 4, 1$ are not zero!

$$\mathbf{U} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; \mathbf{\Sigma} = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \; \mathbf{V} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 9 \end{bmatrix}$$

Those first columns $\mathbf{u}_1$ and $\mathbf{v}_1$ have $1's$ in positions 3 and 4. Then $\mathbf{u}_1\sigma_1\mathbf{v}_1^T$ picks out the biggest number $A_{34} = 3$ in the original matrix $\mathbf{A}$. The three rank-one matrices in the SVD come exactly from the numbers $3, 2, 1$ in $\mathbf{A}$.

$$\mathbf{A} = \mathbf{U\Sigma V^T} = 3\mathbf{u}_1\mathbf{v}_1^T + 2\mathbf{u}_2\mathbf{v}_2^T + 1\mathbf{u}_3\mathbf{v}_3^T + 0\mathbf{u}_4\mathbf{v}_4^T$$

. **Note:** Suppose **I** remove the last row of **A** (all zeros). Then **A** is a 3 by 4 matrix and $\mathbf{AA^T}$ is 3 by 3, its fourth row and column will disappear. We still have eigenvalues $\lambda = 1, 4, 9$ in $\mathbf{A^TA}$ and $\mathbf{AA^T}$, producing the same singular values $\sigma = 3, 2, 1$ in $\mathbf{\Sigma}$.

Removing the zero row of $\mathbf{A}$ (now 3×4) just removes the last row of $\mathbf{\Sigma}$ together with the last row and column of $\mathbf{U}$. Then $(3 \times 4) = (3 \times 3)(3 \times 4)(4 \times 4)$. The SVD is totally adapted to rectangular matrices.

Theorem 2. *Let $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n$ be the n singular values of $m \times n$ matrix $A(m \geq n)$. Then*

1. $||A||_2 = \sigma_1 = \sigma_{max}$.
2. $||A||_F = (\sigma_1^2 + \sigma_2^2 + \ldots + \sigma_n^2)^{\frac{1}{2}}$.
3. $||A^{-1}||_2 = \frac{1}{\sigma_n}$, *where A is $n \times n$ and non singular.*
4. $Cond_2(A) = ||A||_2||A^{-1}||_2 = \frac{\sigma_1}{\sigma_n} = \frac{\sigma_{max}}{\sigma_{min}}$ *if A is non singular.*
5. $rank(A) = number\ of\ nonzero\ singular\ values.$

Chapter 2

The Image Deblurring Problem

When we use a camera, we want the recorded image to be faithful representation of the scene that we see, but every image is more or less blurry. thus image deblurring is fundamental in making picture sharp and useful.

A digital image is composed of picture element called **pixels**. Each pixel is assigned an intensity, meant to charaterize the color of small rectangular segment of scene.

A small image typically has around $256^2 = 65536$ pixel while a high resolution image often has 5 to 10 million pixels.

Cause of blurring: Some blurring always arises in the recording of the digital image, because it is unavoidable that scene information "spill out" to neighboring pixels. for example, the optical system in a camera lens may be out of focus, so that the coming light is smeared out. The same problem arise in astronomical imaging where the coming light in the telescope has been slightly bent by turbulence in the atmosphere. In these and similar situation, inevitable result is that we record a blurred image.

In image deblurring, we seek to recover the original, sharp image by using a mathematical model of blurring process. Key issue is that some information on the some information on the lost detail indeed present in the blurred image, but this information is "hidden" and can only be recovered the if we know the details of the blurring process.

unfortunately there is no hope that we can recover the original image exactly! This is due to various unavoidable error in the recorded image. The most important errors are fluctuation in the recording process and approximation error when representing the image with a limited number of digits.The influence of this noise put a limit on the size of the details that we can hope to recover in the reconstructed image, and the limit depends on both the noise and the blurring process.

2.1 Image as an Arrays of Numbers

A digital image is the a two- or three- dimensional array of numbers representing intensity on gray scale or color scale. Let consider example of two dimension array of numbers representing intensity on gray scale.Consider the following 10×16 array:

$$X = \begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&8&8&0&0&5&5&0&0&3&3&3&3&3&3&0 \\ 0&8&8&0&0&5&5&0&0&3&3&3&3&3&3&0 \\ 0&8&8&0&0&5&5&0&0&0&0&3&3&0&0&0 \\ 0&8&8&0&0&5&5&0&0&0&0&3&3&0&0&0 \\ 0&8&8&0&0&5&5&0&0&0&0&3&3&0&0&0 \\ 0&8&8&0&0&5&5&0&0&0&0&3&3&0&0&0 \\ 0&8&8&0&0&5&5&0&0&0&0&3&3&0&0&0 \\ 0&8&8&0&0&5&5&0&0&0&0&3&3&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix}$$

If we enter the into a **MATLAB** variable X and display the with the commands `imagesc(X), axis image, colormap(gray)`, then we obtain picture of Figure 2.1. The entries with value 8 are displayed as white. entries to zero are black, and values in between are shades of gray.

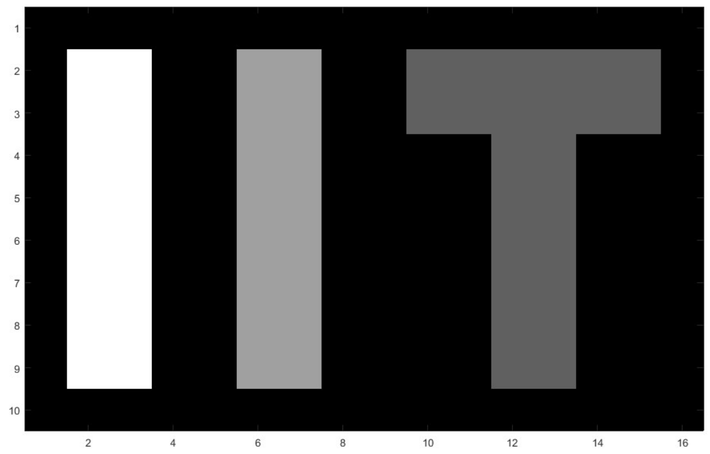

Figure 2.1:

Color image Color images can be be represented using various formats; the **RGB** format stores images as three components, which represent their intensities on red, green and blue scales. A pure red color is represented by intensity values $(1, 0, 0)$ while values $(1, 1, 0)$ represent yellow and (0,0,1) represent blue; other colors can be obtained with different choice of intensities.Hence, to present a color image, we need three value three values per pixel. Let X is a Multidimensional array of dimension $9 \times 16 \times 3$ define as

$$X(:, :, 1) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$X(:, :, 2) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$X(:, :, 3) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

We can display this image, in color, with the command `imagesc(X)`,obtaining the second picture shown in Figure 2.2

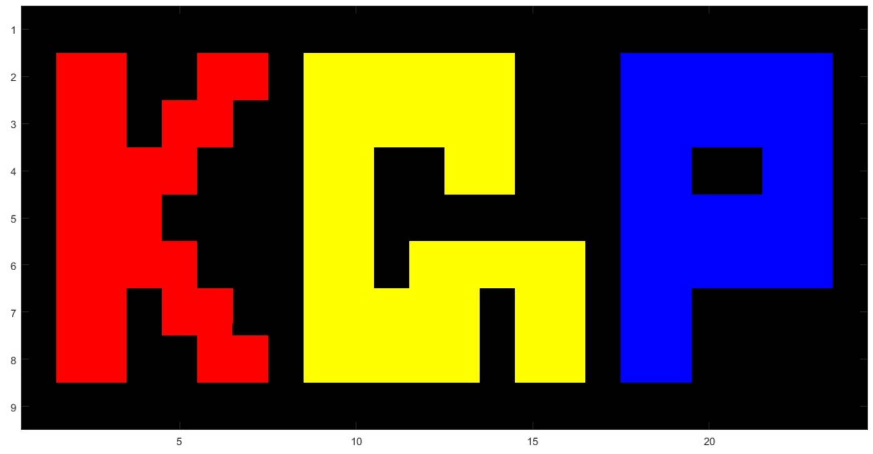

Figure 2.2:

2.2 Simple Linear Model

We have a mathematical model that relates the given blurred image to unknown true image.Consider the the example shown in the Figure 2.3. The left is the "true" scene and right id the blurred version of the same image. The blurred image is precisely what would be record in the camera if photographer forgot to focus the lens.

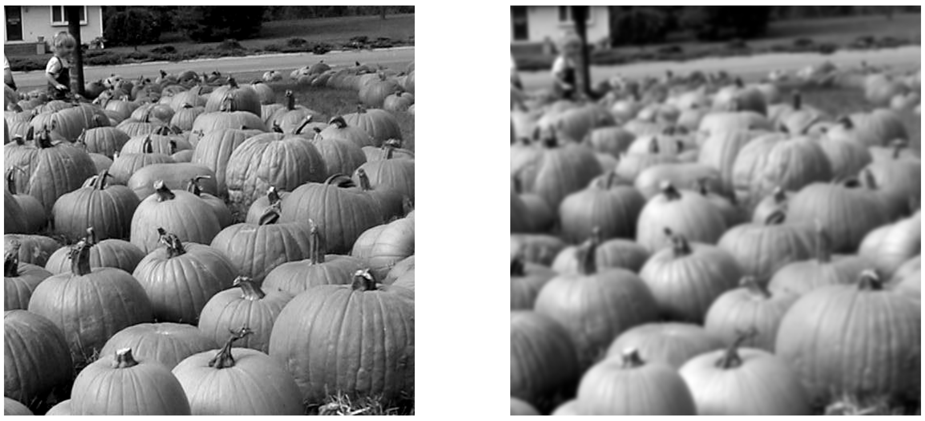

Figure 2.3: A sharp image(left) and the corresponding blurred image (right)

Grayscale images are typically recorded by means of a CCD(charge-coupled device), which is array of tiny detectors, arrange in the rectangular grid, able to record the amount, or intensity, of light that hits each detector. Thus we can think of grayscale digital image as a rectangular $m \times n$ array, whose entries represent light intensity captured by the detectors. Let

$$\mathbf{X} = \mathbb{R}^{m\times n} \; represent\ the\ desired\ sharp\ image, and$$

$$\mathbf{B} = \mathbb{R}^{m\times n} \textit{ represent the recorded blurred image.}$$

Let us first consider a simple case where the blurring of columns in the image is independent of blurring of rows. When this is the case , then there exit two matrices $\mathbf{A}_c \in \mathbf{R}^{m\times m}$ and $\mathbf{A}_r \in \mathbb{R}^{n\times n}$, such that we can express the relation between the sharp and blurred images as

$$\mathbf{A}_c\ \mathbf{X}\ \mathbf{A}_r^T \ = \ \mathbf{B} \tag{2.1}$$

Where matrix $\mathbf{A_c}$ applies the same vertical blurring operation to all n columns $\mathbf{x}_j$ of $\mathbf{X}$, because

$$\mathbf{A}_c\mathbf{X} = \mathbf{A}_c[x_1\ x_2\ \ldots\ x_n] = [\mathbf{A}_c x_1\ \mathbf{A}_c x_2\ \ldots\ \mathbf{A}_c x_n]$$

similarly, the right multiplication with $\mathbf{A}_r^T$ applies the same horizontal blurring to all m rows of $\mathbf{X}$.Since matrix multiplication is associative i.e. $(\mathbf{A}_c\mathbf{X})\mathbf{A}_r^T = \mathbf{A}_c(\mathbf{X}\mathbf{A}_r^T)$, it does not matter in which order we perform the blurring operation.

2.3 Deblurring using simple linear model

If the image blurring model is the simple form $\mathbf{A}_c\ \mathbf{X}\ \mathbf{A}_r^T \ = \ \mathbf{B}$, then one might think that the naive solution is

$$\mathbf{X}_{naive} \ = \ \mathbf{A}_c^{-1}\ \mathbf{B}\ \mathbf{A}_r^{-T}$$

will yield the desired reconstruction, where $\mathbf{A}_r^{-T} = (\mathbf{A}_r^T)^{-1} = (\mathbf{A}_r^{-1})^T$

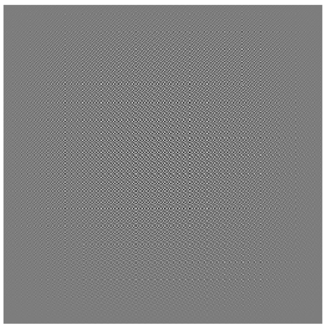

Figure 2.4: The naive reconstruction of pumpkin image in 2.3, obtained by computing $\mathbf{X} \ = \ \mathbf{A}_c^{-1}\ \mathbf{B}\ \mathbf{A}_r^{-T}$ via Gaussian elimination on the both $\mathbf{A}_c$ and $\mathbf{A}_r$. Both matrices are ill-conditioned, and the image $\mathbf{X}_{naive}$ is dominated by the influence from rounding error as well as error in the blurred image $\mathbf{B}$.

To understand the why this naive approach fails, we must realize that the blurring model in (2.1) is not quite correct, because we have ignored several type of errors.

The image represented by $\mathbf{B}$ is the combination of $\mathbf{B}_{exact} = \mathbf{A}_c\ \mathbf{X}\ \mathbf{A}_r^T$ represented the ideal image and all kind of errors.Because the blurred image is collected by a mechanical device, it is inevitable that small random error(noise) will be present in the recorded data. Thus the recorded blurred image $\mathbf{B}$ is really given by

$$\mathbf{B}\ =\ \mathbf{B}_{exact} + \mathbf{E}\ =\ \mathbf{A}_c\ \mathbf{X}\ \mathbf{A}_r^T\ + \mathbf{E} \tag{2.2}$$

where the noise image $\mathbf{E}$ (of the dimensions as B) represents the noise and the quantization errors in the recorded image.Consequently the naive reconstruction is given by

$$\mathbf{X}_{naive}\ =\ \mathbf{A}_c^{-1}\ \mathbf{B}\ \mathbf{A}_r^{-T}\ + \mathbf{A}_c^{-1}\ \mathbf{E}\ \mathbf{A}_r^{-T}$$

and therefore

$$\mathbf{X}_{naive}\ =\ \mathbf{X}\ +\ \mathbf{A}_c^{-1}\ \mathbf{E}\ \mathbf{A}_r^{-T} \tag{2.3}$$

where the term $\mathbf{A}_c^{-1}\mathbf{E}\mathbf{A}_r^{-T}$, which is call **inverted noise**, represent the contribution to the reconstruction from additive noise. This inverted noise will dominate the solution if the second term $\mathbf{A}_c^{-1}\mathbf{E}\mathbf{A}_r^{-T}$ in the (2.3) has larger elements than the first term $\mathbf{X}$. Unfortunately, in many situations, as in Figure 2.4, the inverted noise indeed dominates.

2.4 Deblurring Using General Linear Model

Our basic assumption is that the operation of going from sharp image to the blurred image is linear that is if $\mathbf{B}_1$ and $\mathbf{B}_2$ are the blurred images of the exact images $\mathbf{X}_1$ and $\mathbf{X}_2$ respectively. then blurred image $\mathbf{B} = \alpha\mathbf{B}_1 + \beta\mathbf{B}_2$ is the exact image of $\mathbf{X} = \alpha\mathbf{X}_1 + \beta\mathbf{X}_2$. As usual in the physical science this assumption is made because in many situations the blur is indeed linear or at least well approximated by linear model.

In this model we rearrange the elements of the images $\mathbf{X} \in \mathbb{R}^{m\times n}$ and $\mathbf{B} \in \mathbb{R}^{m\times n}$ into column vectors by stacking the columns of these images into two long vector $\mathbf{x}$ and $\mathbf{b}$, both of length $N = mn$. The mathematical notation for this operation is vec,i.e

$$\mathbf{x} = vec(\mathbf{X}) = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_n \end{bmatrix} \in \mathbb{R}^N,\ \mathbf{b} = vec(\mathbf{B}) = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_n \end{bmatrix} \in \mathbb{R}^N$$

Since the blurring is assumed to be linear operation, there must exist a large **blurring matrix** $\mathbf{A} \in \mathbb{R}^{N\times N}$ such that $\mathbf{x}$ and $\mathbf{b}$ are related by linear model

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{2.4}$$

and this is fundamental image blurring model and we will explain how it can be constructed from the imaging system and what is precise structure of the matrix in next chapter.

For our linear model, the naive approach to image deblurring is simply to solve the linear algebraic system in (2.4) and equation (2.1) is the special case of equation (2.4), and from previous section, solution of equation (2.4), we expected failure. because

$$\mathbf{b} = \mathbf{b}_{exact} + \mathbf{e}$$

where $\mathbf{b} = vec(\mathbf{B})$ is the blurred image with noise, $\mathbf{b}_{exact} = vec(\mathbf{B}_{exact})$ is the noise free blurred image and $\mathbf{e} = vec(\mathbf{E})$ is the noise image and the naive solution is given by

$$\mathbf{x}_{naive} = \mathbf{A}^{-1}\mathbf{b} = \mathbf{A}^{-1}\mathbf{b}_{exact} + \mathbf{A}^{-1}\mathbf{e} = \mathbf{x} + \mathbf{A}^{-1}\mathbf{e} \tag{2.5}$$

where the term $\mathbf{A}^{-1}\mathbf{e}$ is the inverted noise. The important observation here is that the deblurred image consists of two components: the first component is the exact image, and the second component is the inverted noise. If the deblurred image looks unacceptable, it is because the inverted noise term contaminates the reconstructed image.

Important insight about the inverted noise term can be gained using the singular value decomposition (SVD), which is the tool-of-the-trade in matrix computations for analyzing linear systems of equations. The SVD of a square matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ is essentially unique and is defined as the decomposition

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$$

where $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N]$ and $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N]$, Satisfying $\mathbf{U}^T\mathbf{U} = \mathbf{I}_N$ and $\mathbf{V}^T\mathbf{V} = \mathbf{I}_N$and $\Sigma = diag(\sigma_1, \sigma_2, \ldots, \sigma_N)$ is a diagonal matrix whose elements σ_i are nonnegative and appear in nonincreasing order,

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_N \geq 0$$

The quantities σ_i are called the singular values, and the rank of A is equal to the number of positive singular values.The columns $\mathbf{u}_i$ of $\mathbf{U}$ are called the left singular vectors, while the columns $\mathbf{v}_i$ of $\mathbf{V}$ are the right singular vectors. Since $\mathbf{V}$ and $\mathbf{U}$ are are orthogonal matrix therefore $\mathbf{u}_i^T\mathbf{u}_j = 0$ if $i \neq j$ and $\mathbf{v}_i^T\mathbf{v}_j = 0$ if $i \neq j$

If all singular values are strictly positive,then inverse of A is given by

$$\mathbf{A}^{-1} = (\mathbf{U}\Sigma\mathbf{V}^T)^{-1}$$

$$= (\mathbf{V}^T)^{-1}\Sigma^{-1}\mathbf{U}^{-1}$$

$$\because\ (\mathbf{V}^T)^{-1} = (\mathbf{V}^{-1})^T \ and\ \mathbf{V}^{-1} = \mathbf{V}^T,\ \mathbf{U}^{-1} = \mathbf{U}^T$$

$$\implies \mathbf{A}^{-1} = (\mathbf{V}^{-1})^T\Sigma^{-1}\mathbf{U}^{-1}$$

$$\implies \mathbf{A}^{-1} = \mathbf{V}\Sigma^{-1}\mathbf{U}^T$$

Since Σ is a diagonal matrix, its inverse Σ is also diagonal, with entries $\frac{1}{\sigma_i}$, for$i = 1, 2, \ldots, N$ Another representation of $\mathbf{A}$ and $\mathbf{A}^{-1}$ is

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$$

$$= \begin{bmatrix}\mathbf{u}_1 & \mathbf{u}_2 & \ldots & \mathbf{u}_N\end{bmatrix}\begin{bmatrix}\sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_N\end{bmatrix}\begin{bmatrix}\mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_N^T\end{bmatrix}$$

$$= \mathbf{u}_1\sigma_1\mathbf{v}_1^T + \mathbf{u}_2\sigma_2\mathbf{v}_2^T + \ldots + \mathbf{u}_N\sigma_N\mathbf{v}_N^T$$

$$= \sum_{i=1}^{N} \sigma_i\mathbf{u}_i\mathbf{v}_i^T$$

and

$$\mathbf{A}^{-1} = \mathbf{V}\Sigma^{-1}\mathbf{U}^T$$

$$= \begin{bmatrix}\mathbf{v}_1 & \mathbf{v}_2 & \ldots & \mathbf{v}_N\end{bmatrix}\begin{bmatrix}\frac{1}{\sigma_1} & & & \\ & \frac{1}{\sigma_2} & & \\ & & \ddots & \\ & & & \frac{1}{\sigma_N}\end{bmatrix}\begin{bmatrix}\mathbf{u}_1^T \\ \mathbf{u}_2^T \\ \vdots \\ \mathbf{u}_N^T\end{bmatrix}$$

$$= \mathbf{v}_1\frac{1}{\sigma_1}\mathbf{u}_1^T + \mathbf{v}_2\frac{1}{\sigma_2}\mathbf{u}_2^T + \ldots + \mathbf{v}_N\frac{1}{\sigma_N}\mathbf{u}_N^T$$

$$= \sum_{i=1}^{N} \frac{1}{\sigma_i}\mathbf{v}_i\mathbf{u}_i^T$$

Now reconstruction given by in equation (2.5) ac be written as

$$\mathbf{x}_{naive} = \mathbf{A}^{-1}\mathbf{b} = \mathbf{V}\Sigma^{-1}\mathbf{U}^T\mathbf{b} = \sum_{i=1}^{N}\left(\frac{1}{\sigma_i}\mathbf{v}_i\mathbf{u}_i^T\right)\mathbf{b} = \sum_{i=1}^{N}\frac{\mathbf{u}_i^T\mathbf{b}}{\sigma_i}\mathbf{v}_i \tag{2.6}$$

$$\because\ \mathbf{b} = \mathbf{b}_{exact} + \mathbf{e}$$

$$\therefore \mathbf{x}_{naive} = \mathbf{A}^{-1}\mathbf{b} + \mathbf{A}^{-1}\mathbf{e} = \mathbf{V}\Sigma^{-1}\mathbf{U}^T\mathbf{b} + \mathbf{V}\Sigma^{-1}\mathbf{U}^T\mathbf{e} = \sum_{i=1}^{N}\frac{\mathbf{u}_i^T\mathbf{b}}{\sigma_i}\mathbf{v}_i + \sum_{i=1}^{N}\frac{\mathbf{u}_i^T\mathbf{e}}{\sigma_i}\mathbf{v}_i$$

and inverted noise contribution to the solution is given by

$$\mathbf{A}^{-1}\mathbf{e} = \mathbf{V}\Sigma^{-1}\mathbf{U}^T\mathbf{e} = \sum_{i=1}^{N} \frac{\mathbf{u}_i^T\mathbf{e}}{\sigma_i}\mathbf{v}_i \tag{2.7}$$

To understand when this error term dominates the solution, for that we need to know following (SVD) properties generally hold for image deblurring problem:
(i) The error components $|\mathbf{u}_i^T e|$ are small and typically of roughly the same order of magnitude for all i.
(ii) The singular values decay to a value very close to zero. As a consequence the condition number

$$cond(\mathbf{A}) = \frac{\sigma_1}{\sigma_N}$$

is very large, indicating that the solution is very sensitive to perturbation and rounding errors.
(iii) The singular vectors corresponding to the smaller singular values typically represent higher-frequency information. *i.e.* as i increases, the vectors $\mathbf{u}_i$, and $\mathbf{v}_i$,tend to have more sign changes.
The consequence of the (iii) property is that the SVD provides us with basis vectors $\mathbf{v}_i$, for an expansion where each basis vector represents a certain "frequency," approximated by the number of times the entries in the vector change signs.

Let reshaped the each singular vectors $\mathbf{v}_i$ into an $m \times n$ array $\mathbf{V}_i$ *i.e.*

$$\mathbf{v}_i = \begin{bmatrix} v_{i1} \\ v_{i2} \\ v_{i3} \\ v_{i4} \\ \vdots \\ \vdots \\ v_{iN} \end{bmatrix}_{N\times 1} = \begin{bmatrix} v_{i1} & v_{i(m+1)} & \cdots & v_{i((n-1)m+1)} \\ v_{i2} & v_{i(m+2)} & \cdots & v_{i((n-1)m+2)} \\ \vdots & \vdots & \ddots & \vdots \\ v_{im} & v_{i(2m)} & \cdots & v_{i(nm)} \end{bmatrix}_{m\times n} = \mathbf{V}_i$$

Now naive Solution of equation (2.6) can be written as

$$\mathbf{x}_{naive} = \sum_{i=1}^{N} \frac{\mathbf{u}_i^T\mathbf{b}}{\sigma_i}\mathbf{V}_i \tag{2.8}$$

All the $\mathbf{V}_i$ arrays (except the first) have negative elements and therefore, strictly speaking,they are not images and the spatial frequencies in $\mathbf{V}_i$ increase with the index i.

When we encounter an expansion of the form $\sum_{i=1}^{N} \xi_i \mathbf{v}_i$, where $\xi_i = \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i}$. then the i^{th} expansion coefficient ξ_i, measures the contribution of $\mathbf{v}_i$ to the result. And since each vector $\mathbf{v_i}$, can be associated with some "frequency," the i^{th} coefficient measures the amount of information of that frequency in the image.

For $\mathbf{A}^{-1}\mathbf{e}$ we see that the quantities $\frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i}$ are the expansion coefficients for the basis vectors $\mathbf{v}_i$. When these quantities are small in magnitude, the solution has very little contribution from $\mathbf{v}_i$, but when we divide by a small singular value such as σ_N, we greatly magnify the corresponding error component, $\mathbf{u}_N^T \mathbf{e}$, which in turn contributes a large multiple of the high-frequency information contained in $\mathbf{v}_N$ to the computed solution. This is precisely why a naive reconstruction, such as the one in Figure 2.4, appears as a random image dominated by high frequencies.

Because of this, It is better to chop out of the high-frequency components altogether, since they are dominated by error. For example, for some choice of $k \leq N$ we can compute the truncated expansion

$$\mathbf{x}_k = \sum_{i=1}^{k} \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i} \mathbf{v}_i \tag{2.9}$$

we can rewrite the equation above equation in form of

$$\mathbf{x}_k = \mathbf{A}_k^{\dagger} \mathbf{b}$$

where $\mathbf{A}_k^{\dagger} = \sum\limits_{i=1}^{k} \frac{1}{\sigma_i} \mathbf{v}_i \mathbf{u}_i^T$ and $\mathbf{A}_k$ is the rank-k matrix

$$\mathbf{A}_k^{\dagger} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \dots & \mathbf{v}_k \end{bmatrix} \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_k \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{u}_1^T \\ \mathbf{u}_2^T \\ \vdots \\ \mathbf{u}_k^T \end{bmatrix}$$

The truncated SVD expansion for x_k involves the computation of the SVD of the large $N \times N$ matrix $\mathbf{A}$ and is therefore computationally feasible only if we can find fast algorithms to compute the decomposition.

We model the blurring of images as a linear process characterized by a blurring matrix $\mathbf{A}$ and an observed image $\mathbf{B}$, which, in vector form, is $\mathbf{b}$. The reason $\mathbf{A}^{-1}\mathbf{b}$ cannot be used to deblur images is the amplification of high-frequency components of the noise in the data, caused by the inversion of very small singular values of $\mathbf{A}$. Practical methods for image deblurring need to avoid this pitfall.

Chapter 3

Point Spread Function (PSF)

In the linear model, there exists a large matrix $\mathbf{A}$ of dimensions $N \times N$, with $N = mn$,such that $\mathbf{b}$ and $\mathbf{x}$ are related by the equation

$$\mathbf{Ax} = \mathbf{b}$$

The matrix $\mathbf{A}$ represents the blurring that is taking place in the process of going from the exact to the blurred image.

At this stage we know there is such a matrix, but how do we get it? Imagine the following experiment. Suppose we take the exact image to be all black, except for a single bright pixel. If we take a picture of this image, then the blurring operation will cause the single bright pixel to be spread over its neighboring pixels, as illustrated in figure 3.1. For obvious reasons, the single bright pixel is called a **point source**, and the function that describes the blurring and the resulting image of the point source is called the **point spread function (PSF)**.

Mathematically, the point source is equivalent to defining an array of all zeros, except a single pixel whose value is 1. That is, we set $\mathbf{x} = \mathbf{e}_i$, to be the i^{th} unit vector which consists of all zeros except the i^{th} entry, which is 1. The process of taking a picture of this true image is equivalent to computing

$$\mathbf{Ae}_i = \mathbf{A}(:, i) = column\ i\ of\ \mathbf{A}$$

Clearly, if we repeat this process for all unit vectors $\mathbf{e}_i$,for $i = 1, 2, ..., N$, then in principle we have obtained complete information about the matrix $\mathbf{A}$.

The blurring matrix A is determined from two ingredients: the PSF, which defines how each pixel is blurred, and the boundary conditions, which specify our assumptions on the scene just outside our image.

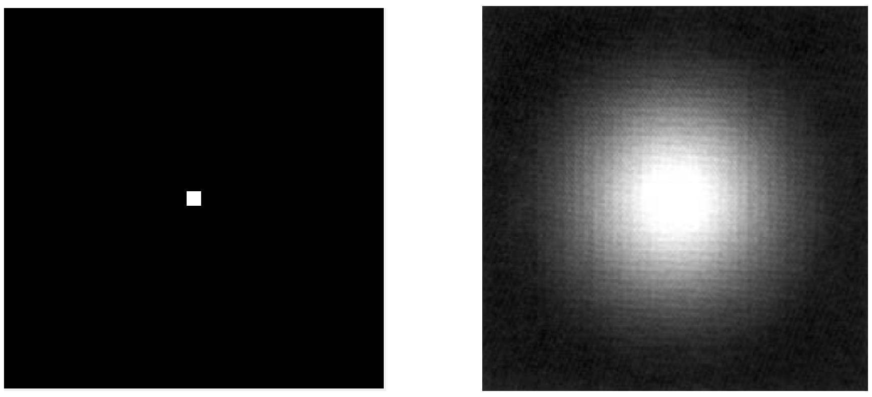
Figure 3.1: Left: a single bright pixel, called point source. Right: the blurred point source, called a point spread function

3.1 Obtaining the PSF

The light intensity of the PSF is confined to a small area around the center of the PSF (the pixel location of the point source), and outside a certain radius, the intensity is essentially zero. In other words, the blurring is a local phenomenon. If we assume that the imaging process captures all light, then the pixel values in the PSF must sum to 1

. The PSF is the same regardless of the location of the point source. When this is the case, we say that the blurring is spatially invariant. As a consequence of this linear and local nature of the blurring, to conserve storage we can often represent the PSF using an array P of much smaller dimension than the blurred image

Many of our deblurring algorithms require that the PSF array be the same size as the blurred image. In this case the small PSF array is embedded in a larger array of zeros; this process is often referred to as "zero padding".

In some cases the PSF can be described analytically, and thus **P** can be constructed from a function, rather than through experimentation. Consider, for example, horizontal motion blur, which smears a point source into a line. If the line covers r pixels—over which the light is distributed—then the magnitude of each nonzero element in the PSF array is r^{-1}.The same is true for vertical motion blur. An example of the PSF array for horizontal motion is shown in Figure 3.2

In other cases, knowledge of the physical process that causes the blur provides an explicit formulation of the PSF. When this is the case, the elements of the PSF array are given by a precise mathematical expression. For example, the elements p_{ij} of the PSF

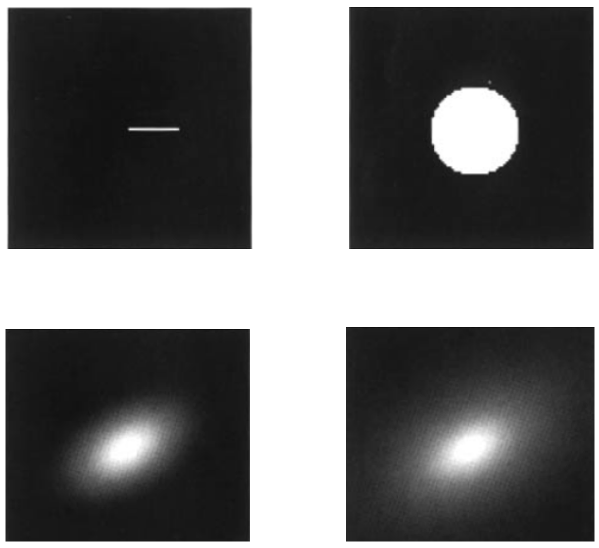

Figure 3.2: Top left: Horizontal motion blur,Top right: Out-of-focus blur, Bottom left: Atmospheric turbulence blur, Bottom right: Moffat blur

array for out-of-focus blur are given by

$$p_{ij} = \begin{cases} \frac{1}{\pi r^2} & \text{if } (i-k)^2 + (j-l)^2 \leq r^2 \\ 0 & \text{otherwise} \end{cases}$$

where (k, l) is the center of $\mathbf{P}$, and r is the radius of the blur. The PSF for blurring caused by atmospheric turbulence can be described as a twodimensional Gaussian function, and the elements of the unsealed PSF array are given by

$$p_{ij} = \frac{1}{2\pi s_1 s_2} exp\left(-\frac{1}{2}\left(\frac{i-k}{s_1}\right)^2 - \frac{1}{2}\left(\frac{j-l}{s_2}\right)^2\right)$$

The PSF of an astronomical telescope is often modeled by the so-called Moffat function and for this PSF the elements of the unsealed PSF array are given by

$$p_{ij} = \left(1 + \left(\frac{i-k}{s_1}\right)^2 + \left(\frac{j-l}{s_2}\right)^2\right)^{-\beta}$$

The PSF array P is the image of a single white pixel, and its dimensions are usually much smaller than those of **B** and **X**. If the blurring is local and spatially invariant,then **P** contains all information about the blurring throughout the image.

Once the PSF array is specified, we can always construct the big blurring matrix **A** one column at a time by simply placing the elements of P in the appropriate positions,leaving zeros elsewhere in the column.

Chapter 4

Basic Structure of blur Matrix A

4.1 One-Dimensional Problems

Convolution is a mathematical operation that can be described as follows. If $p(s)$ and $x(s)$ are continuous functions, then the convolution of p and x is a function b having the form

$$b(s) = \int_{\infty}^{\infty} x(t)p(s-t)dt$$

That is, each value of the function $b(s)$ is essentially a weighted average of the values of $x(t)$, where the weights are given by the function p. In order to perform the integration, we must first "flip" the function $p(t)$ to obtain $p(-t)$, and then "shift" to obtain $p(s-t)$.

The discrete version of convolution is a summation over a finite number of terms. That is, pixels of the blurred image are obtained from a weighted sum of the corresponding pixel and its neighbors in the true image. The weights are given by the elements in the PSF array. It is perhaps easier to illustrate the discrete operation for one-dimensional convolution using a small example. Suppose a true image scene and PSF array are given, respectively, by the one-dimensional arrays

$$\mathbf{x} = \left[\begin{array}{c} w_1 \\ w_2 \\ \hline x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ \hline y_1 \\ y_2 \end{array}\right] \quad and\ \mathbf{p} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{bmatrix}$$

where w_i and y_i represent pixels in the original scene that are actually outside the field of view. The basic idea of computing the convolution **b**, of $\mathbf{x}x$ and **p**, can be summarized as follows:

- Rotate (i.e., flip) the PSF array, **p**, by 180 degrees, writing its elements from bottom to top.
- Match coefficients of the rotated PSF array with those in **x** by placing (i.e., shifting) the center of the PSF array (i.e., the entry corresponding to a shift of zero) over the i^{th} entry in **x**.
- Multiply corresponding components and sum to get the i^{th} entry in **b**.

For example, assuming that p_3 is the center of the PSF array, we have

$$b_1 = p_5 w_1 + P_4 w_2 + p_3 x_1 + p_2 x_2 + p_1 x_3$$

$$b_2 = p_5 w_2 + P_4 x_1 + p_3 x_2 + p_2 x_3 + p_1 x_4$$

$$b_3 = p_5 x_1 + P_4 x_2 + p_3 x_3 + p_2 x_4 + p_1 x_5$$

$$b_4 = p_5 x_2 + P_4 x_3 + p_3 x_4 + p_2 x_5 + p_1 y_1$$

$$b_5 = p_5 x_3 + P_4 x_4 + p_3 x_5 + p_2 y_1 + p_1 y_2$$

Thus, convolution can be written as a matrix-vector multiplication,

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \begin{bmatrix} p_5 & p_4 & P_3 & p_2 & p_1 & & & & \\ & p_5 & p_4 & P_3 & p_2 & p_1 & & & \\ & & p_5 & p_4 & P_3 & p_2 & p_1 & & \\ & & & p_5 & p_4 & P_3 & p_2 & p_1 & \\ & & & & p_5 & p_4 & P_3 & p_2 & p_1 \end{bmatrix} \left[\begin{array}{c} w_1 \\ w_2 \\ \hline x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ \hline y_1 \\ y_2 \end{array}\right] \tag{4.1}$$

It is important to keep in mind that the values w_i and y_i contribute to the observed pixels in the blurred image even though they are outside the field of view. Since we do not know these values, boundary conditions are used to relate them to the pixels x_i, within the field of view. We have the following Boundary conditions:

- **Zero Boundary Conditions:** In this case, $w_i = y_i = 0$, and thus equation (4.1) can be rewritten as

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \begin{bmatrix} P_3 & p_2 & p_1 & & \\ p_4 & P_3 & p_2 & p_1 & \\ p_5 & p_4 & P_3 & p_2 & p_1 \\ & p_5 & p_4 & P_3 & p_2 \\ & & p_5 & p_4 & P_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} \tag{4.2}$$

  A matrix whose entries are constant on each diagonal, such as in equation (4.2), is called a **Toeplitz matrix** with parameters **p**.

- **Periodic Boundary Conditions:** In this case we assume that the true scene is comprised of periodic copies of **x**, so $w_1 = x_4,\ w_2 = x_5,\ y_1 = x_1$ and $y_2 = x_2$ Thus euqation (4.1) can be rewritten as

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \left(\begin{bmatrix} P_3 & p_2 & p_1 & & \\ p_4 & P_3 & p_2 & p_1 & \\ p_5 & p_4 & P_3 & p_2 & p_1 \\ & p_5 & p_4 & P_3 & p_2 \\ & & p_5 & p_4 & P_3 \end{bmatrix} + \begin{bmatrix} & & & p_5 & p_4 \\ & & & & p_5 \\ & & & & \\ p_1 & & & & \\ p_2 & p_1 & & & \end{bmatrix} \right) \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}$$

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \begin{bmatrix} P_3 & p_2 & p_1 & p_5 & p_4 \\ p_4 & P_3 & p_2 & p_1 & p_5 \\ p_5 & p_4 & P_3 & p_2 & p_1 \\ p_1 & p_5 & p_4 & P_3 & p_2 \\ p_1 & p_2 & p_5 & p_4 & P_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} \tag{4.3}$$

  A Toeplitz matrix in which each row (and column) is a periodic shift of its previous row (column), such as in equation (4.3), is called a **circulant matrix**.

- **Reflexive Boundary Conditios:** In this case we assume that the true scene immediately outside the field of view is a mirror reflection of the scene within the field of view. Thus $w_1 = x_2, w_2 = x_1, y_1 = x_5$ and $y_2 = x_4$, and equation (4.1) can be rewritten as

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \left(\begin{bmatrix} P_3 & p_2 & p_1 & & \\ p_4 & P_3 & p_2 & p_1 & \\ p_5 & p_4 & P_3 & p_2 & p_1 \\ & p_5 & p_4 & P_3 & p_2 \\ & & p_5 & p_4 & P_3 \end{bmatrix} + \begin{bmatrix} P_4 & p_5 & & & \\ p_5 & & & & \\ & & & & \\ & & & & p_1 \\ & & & p_1 & P_2 \end{bmatrix} \right) \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}$$

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \begin{bmatrix} P_3 + p_4 & p_2 + p_5 & p_1 & & \\ p_4 + p_5 & P_3 & p_2 & p_1 & \\ p_5 & p_4 & P_3 & p_2 & p_1 \\ & p_5 & p_4 & P_3 & p_2 + p_1 \\ & & p_5 & p_4 + P_1 & P_3 + p_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} \tag{4.4}$$

A matrix whose entries are constant on each antidiagonal is called a **Hankel matrix**, so the matrix given in equation(4.4) is a **Toeplitz-plus-Hankel matrix**.

4.2 Two-Dimensional Problems

The convolution operation for two-dimensional images is very similar to the one-dimensional case. In particular, to compute the (i, j) pixel of the convolved image, $\mathbf{B}$, the PSF array is rotated by 180 degrees and matched with pixels in the true scene, $\mathbf{X}$, by placing the center of the rotated PSF array over the (i, j) pixel in $\mathbf{X}$. Corresponding components are multiplied, and the results summed to compute b_{ij}. For example, let

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix}, \; \mathbf{P} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix}, \; \mathbf{B} = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix}$$

with p_{22} the center of the PSF array. Then the element b_{22} in the center of $\mathbf{B}$ is given by

$$\begin{aligned} b_{22} &= p_{33}.x_{11} + p_{32}.x_{12} + p_{31}.x_{13} \\ &+ p_{23}.x_{21} + p_{22}.x_{22} + p_{21}.x_{23} \\ &+ p_{13}.x_{31} + p_{12}.x_{32} + p_{11}.x_{33} \end{aligned}$$

For all other elements in this 3×3 example, we must make use of the boundary conditions.In particular, if we assume zero boundary conditions, then the element b_{11} and b_{21} at the border of $\mathbf{B}$ is given by

$$\begin{aligned} b_{11} &= p_{33}.0 + p_{32}.0 + p_{31}.0 \\ &+ p_{23}.0 + p_{22}.x_{11} + p_{21}.x_{12} \\ &+ p_{13}.0 + p_{12}.x_{21} + p_{11}.x_{22} \end{aligned}$$

$$\begin{aligned} b_{21} &= p_{33}.0 + p_{32}.x_{11} + p_{31}.x_{12} \\ &+ p_{23}.0 + p_{22}.x_{21} + p_{21}.x_{22} \\ &+ p_{13}.0 + p_{12}.x_{31} + p_{11}.x_{32} \end{aligned}$$

sililarly we can compute the other element of matrix $\mathbf{B}$

- For **zero Boundary condition**, $\mathbf{b} = vec(\mathbf{B})$ and $\mathbf{x} = vec(\mathbf{X})$ are related by

$$\begin{bmatrix} b_{11} \\ b_{21} \\ b_{31} \\ b_{12} \\ b_{22} \\ b_{32} \\ b_{13} \\ b_{23} \\ b_{33} \end{bmatrix} = \left[\begin{array}{ccc|ccc|ccc} p_{22} & p_{12} & & p_{21} & b_{11} & & & & \\ p_{32} & p_{22} & p_{12} & p_{31} & b_{21} & p_{11} & & & \\ & p_{32} & p_{22} & & b_{31} & p_{21} & & & \\ \hline p_{23} & p_{13} & & p_{22} & b_{12} & & p_{21} & p_{11} & \\ p_{33} & p_{23} & p_{13} & b_{32} & p_{22} & p_{12} & p_{31} & p_{21} & p_{11} \\ & p_{33} & p_{23} & & p_{32} & b_{22} & & p_{31} & p_{21} \\ \hline & & & p_{23} & p_{13} & & p_{22} & b_{12} & \\ & & & p_{33} & p_{23} & b_{13} & p_{32} & b_{22} & b_{12} \\ & & & & p_{33} & b_{23} & & b_{32} & b_{22} \end{array}\right] \begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \\ x_{12} \\ x_{22} \\ x_{32} \\ x_{13} \\ x_{23} \\ x_{33} \end{bmatrix} \tag{4.5}$$

The matrix in equation(4.5) has a block Toeplitz structure (as indicated by the lines), and each block is itself a Toeplitz matrix. We call such a matrix **block Toeplitz with Toeplitz blocks (BTTB)**

- when periodic boundary conditions are employed, element b_{21} for example, is given by

$$\begin{aligned} b_{21} &= p_{33}.x_{13} + p_{32}.x_{11} + p_{31}.x_{12} \\ &+ p_{23}.x_{23} + p_{22}.x_{21} + p_{21}.x_{22} \\ &+ p_{13}.x_{33} + p_{12}.x_{31} + p_{11}.x_{32} \end{aligned}$$

Similarly we can compute other element of $\mathbf{B}$. IF it follow periodic boundary condition, the linear model takes the form

$$\begin{bmatrix} b_{11} \\ b_{21} \\ b_{31} \\ b_{12} \\ b_{22} \\ b_{32} \\ b_{13} \\ b_{23} \\ b_{33} \end{bmatrix} = \left[\begin{array}{ccc|ccc|ccc} p_{22} & p_{12} & p_{32} & p_{21} & b_{11} & p_{31} & p_{23} & p_{13} & p_{33} \\ p_{32} & p_{22} & p_{12} & p_{31} & b_{21} & p_{11} & p_{33} & p_{23} & p_{13} \\ p_{12} & p_{32} & p_{22} & p_{11} & b_{31} & p_{21} & p_{13} & p_{33} & p_{23} \\ \hline p_{23} & p_{13} & p_{33} & p_{22} & b_{12} & p_{32} & p_{21} & p_{11} & p_{31} \\ p_{33} & p_{23} & p_{13} & b_{32} & p_{22} & p_{12} & p_{31} & p_{21} & p_{11} \\ p_{13} & p_{33} & p_{23} & p_{12} & p_{32} & b_{22} & p_{11} & p_{31} & p_{21} \\ \hline p_{21} & p_{11} & p_{31} & p_{23} & p_{13} & p_{33} & p_{22} & b_{12} & p_{32} \\ p_{31} & p_{21} & p_{11} & p_{33} & p_{23} & b_{13} & p_{32} & b_{22} & b_{12} \\ p_{11} & p_{31} & p_{21} & p_{13} & p_{33} & b_{23} & p_{12} & b_{32} & b_{22} \end{array}\right] \begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \\ x_{12} \\ x_{22} \\ x_{32} \\ x_{13} \\ x_{23} \\ x_{33} \end{bmatrix} \tag{4.6}$$

Matrix used in equation (4.6) is called **block circulant with circulant block (BCCB)**

References

1. Biswa Nath Datta,Numerical linear algebra and applications,second edition, ISBN:978-0-89871-685-6,Society for Industrial and Applied Mathematics3600 University City Science Center Philadelphia, PA, United States, 20 January 2010.

2. Hansen P.C., Nagy, J. G., and O'Leary, D. P. Deburring Images: Matrices, Spectra, and Filtering, eISBN:978-0-89871-887-4, SIAM Philadelphia,2006.