Digital Image Processing Syllabus proposal for Autonomy

Stream: ECE

Subject Name: Digital Image Processing

Subject Code: EC 703 A

Contact hour: 3P

Total contact hour- 35

Credits: 3

Prepared by: Koushik Pal

Course Objective:

- To become familiar with digital image fundamentals To learn Transform of Digital Images and its applications.
- To get familiar with simple image enhancement techniques in both spatial and frequency domain.
- To become familiar with image compression and recognition methods
- To learn concepts of image restoration techniques and image segmentation and representation techniques.
- To study the Edge detection in Digital Image Processing.
- To become familiar with basics of Security in Digital Image Processing

Course Outcome:

Sem No.	Course Title (Code)	CO Codes	Course Outcomes
7th	Digital Image Processing (EC 703 A)	CO.EC703A.1	Have a clear idea on Digital Imaging fundamentals and Importance of Digital Image Transform.
		CO.EC703A.2	Understanding the importance of Digital Image enhancement in spatial and frequency domain and filtering techniques
		CO.EC703A.3	Explaining the requirements and types of Image Compression and its standards.
		CO.EC703A.4	Demonstrate the basic concepts of Digital Image Restoration and Segmentation of Digital Images
		CO.EC703A.5	Familiarize with Edge detection techniques and concepts on security in Digital Image Processing

Course Content:

Module No.	Topics	No. of Lectures Required
1	**Digital Imaging Fundamentals:** Basic idea of Digital image, Image formation in human eye, Pixel, Mathematical operation of Digital Image, Sampling, Quantization, application of digital Image Processing **Transform of Digital Images:** Importance of Digital Image Transform, Fourier Transform of Digital Image (DFT), Inverse Fourier Transform (IDFT), Fast Fourier Transform, Inverse Fast Fourier Transform, Application of Digital Image Transform in different area	3 4
2	**Digital Image Enhancement:** Importance of Digital Image enhancement, enhancement in spatial and frequency domain, Bit plane slicing, Histogram, Histogram Equalization , Mean and Median filtering in Digital Images, Frequency domain filtering in Digital Images – LPF, HPF and BPF	6
3	**Digital Image Compression:** Importance of Digital Image Compression, Types of Image Compression, example of lossless and lossy compression, Image compression standards, Compression in spatial domain, compression using Huffman coding, DCT and Wavelet based Digital image compression	6
4	**Digital Image Restoration :** Application and Importance of Digital Image Restoration, Reason for Image degradation, Inverse filtering **Segmentation of Digital Images:** Importance and applications of Digital Image Segmentation, Detection of discontinuities, Edge linking and Boundary detection, Thresholding, Segmentation based on Region Growing, Watershed algorithm,	3 5
5	**Edge detection in Digital Image Processing:** Importance of Edge detection in Digital Image Processing, Types of Edge Detection, Mathematical Equation of each operator. **Security in Digital Image Processing :** Importance of Digital Image Security, Watermarking, Image encryption in spatial and frequency domain, Steganography	4 4

TEXT BOOK:

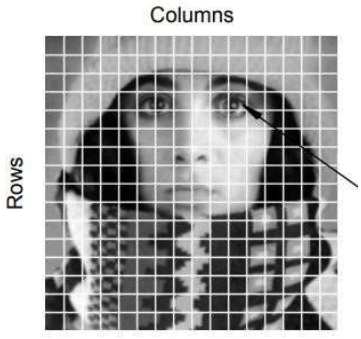
- Rafael C. Gonzales, Richard E. Woods, “Digital Image Processing”, Third Edition, Pearson Education, 2010.
- S. Annadurai,R. Shanmugalakshmi, “Fundamentals of Digital Image Processing”, Pearson Education, 2006

REFERENCES:

- Rafael C. Gonzalez, Richard E. Woods, Steven L. Eddins, “Digital Image Processing Using MATLAB”, Third Edition Tata Mc Graw Hill Pvt. Ltd., 2011. □ Anil Jain K. “Fundamentals of Digital Image Processing”, PHI Learning Pvt. Ltd., 2011.
- Willliam K Pratt, “Digital Image Processing”, John Willey, 2002.
- Malay K. Pakhira, “Digital Image Processing and Pattern Recognition”, First Edition, PHI Learning Pvt. Ltd., 2011.

Module 1: Digital Imaging Fundamentals:

Digital Image Definitions A digital image a[m,n] described in a 2D discrete space is derived from an analog image a(x,y) in a 2D continuous space through a sampling process that is frequently referred to as digitization. For now we will look at some basic definitions associated with the digital image. The effect of digitization is shown in Figure 1. The 2D continuous image a(x,y) is divided into N rows and M columns. The intersection of a row and a column is termed a pixel. The value assigned to the integer coordinates [m,n] with {m=0,1,2,…,M–1} and {n=0,1,2,…,N–1} is a[m,n]. In fact, in most cases a(x,y) – which we might consider to be the physical signal that impinges on the face of a 2D sensor – is actually a function of many variables including depth (z), color (λ), and time (t).

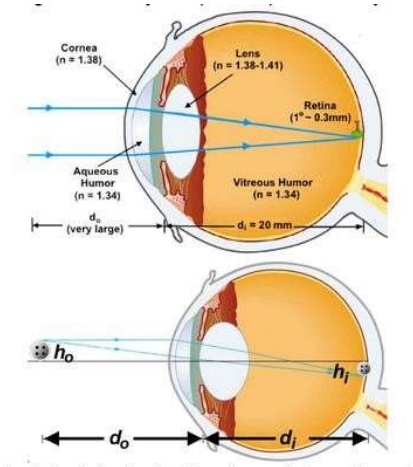


The image shown in Figure 1 has been divided into N = 16 rows and M = 16 columns. The value assigned to every pixel is the average brightness in the pixel rounded to the nearest integer value. The process of representing the amplitude of the 2D signal at a given coordinate as an integer value with L different gray levels is usually referred to as amplitude quantization or simply quantization.

Analysis of image production by the human eye :

While optical instruments can analyze the properties of light, such as interference and refraction with the interferometer and refractometer, respectively, a more familiar role of optical instruments is for image enhancement. There are two types of lens, convex and concave, but converging lens are much more prevalent since they can function as an image producer and magnifier.

The cornea and lens form a system that acts approximately like a single thin lens. Nonetheless, refraction primarily occurs at the cornea and then secondarily at the lens, so most of the power of the eye can be attributed to the cornea. For clear vision, a real image must be projected precisely onto the retina. Because the lens-to-retina distance does not change, the image distance remains the same at 2.0 cm for objects at all distances. The eye manages around this restriction by varying the power (and focal length) of the lens for objects at various distances, i.e. accommodation. It allows a person with normal vision to see objects clearly at distances ranging from 25 cm to essentially infinity.



(n = index of refraction, d_o = object distance, d_i = image distance)
Attribution: OpenStax College, CC-BY 3.0.

Fourier Transforms

The Fourier transform produces another representation of a signal, specifically a representation as a weighted sum of complex exponentials. Because of Euler's formula:

$$e^{jq} = \cos(q) + j\sin(q)$$

where $j^2 = -1$, we can say that the Fourier transform produces a representation of a (2D) signal as a weighted sum of sines and cosines. The defining formulas for the forward Fourier and the inverse Fourier transforms are as follows. Given an image a and its Fourier transform A, then the forward transform goes from the spatial domain (either continuous or discrete) to the frequency domain which is always continuous.

$$Forward - A = F\{a\}$$

The inverse Fourier transform goes from the frequency domain back to the spatial domain.

$$Inverse - a = F^{-1}\{A\}$$

The Fourier transform is a unique and invertible operation so that:

Substituting the above expression in (1.27) we get

$$a = F^{-1}\{F\{a\}\} \qquad and \qquad A = F\{F^{-1}\{A\}\}$$

The specific formulas for transforming back and forth between the spatial domain and the frequency domain are given below In 2D continuous space:

$$Forward: A(u,v) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} a(x,y)e^{-j(ux+vy)}dxdy$$

$$Inverse: a(x,y) = \frac{1}{4\pi^2}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} A(u,v)e^{+j(ux+vy)}dudv$$

In 2D Discrete space:

$$Forward: A(\Omega,\Psi) = \sum_{m=-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty} a[m,n]e^{-j(\Omega m,\Psi\pi)}$$

$$Inverse: a[m,n] = \frac{1}{4\pi^2}\int_{-\pi}^{+\pi}\int_{-\pi}^{+} A(\Omega,\Psi)e^{+j(\pi m+\Psi n)}d\Omega d\Psi$$

Properties of Fourier Transforms

- **Importance of phase and magnitude**
- **Circularly symmetric signals**
- **Examples of 2D signals and transforms**

There are a variety of properties associated with the Fourier transform and the inverse Fourier transform. The following are some of the most relevant for digital image processing.

* The Fourier transform is, in general, a complex function of the real frequency variables. As such the transform con be written in terms of its magnitude and phase.

$$A(u,v) = |A(u,v)|e^{j\phi(u,v)} \qquad A(\Omega,\Psi) = |A(\Omega,\Psi)|e^{j\phi(\Omega,\Psi)}$$

* A 2D signal can also be complex and thus written in terms of its magnitude and phase.

$$a(x,y) = |a(x,y)|e^{j\theta(x,y)} \;;\; a(m,n) = |a(m,n)|e^{j\theta(m,n)}$$

* If a 2D signal is real, then the Fourier transform has certain symmetries.

$$A(u,v) = A^*(-u,-v) \qquad A(\Omega,\Psi) = A^*(-\Omega,-\Psi)$$

The symbol (*) indicates complex conjugation. For real signals equation leads directly to,

$$|A(u,v)| = |A(-u,-v)| \qquad \phi(u,v) = -\phi(-u,-v)$$

$$|A(\Omega,\Psi)| = |A(-\Omega,-\Psi)| \qquad \phi(\Omega,\Psi) = -\phi(-\Omega,-\Psi)$$

* If a 2D signal is real and even, then the Fourier transform is real and even

$$A(u,v) = A(-u,-v) \qquad A(\Omega,\Psi) = A(-\Omega,-\Psi)$$

* The Fourier and the inverse Fourier transforms are linear operations

$$F\{w_1 a + w_2 b\} = F\{w_1 a\} + F\{w_2 b\} = w_1 a + w_2 b$$

$$F^{-1}\{w_1 A + w_2 B\} = F^{-1}\{w_1 A\} + F^{-1}\{w_2 B\} = w_1 a + w_2 b$$

where a and b are 2D signals(images) and w_1 and w_2 are arbitrary, complex constants.

* The Fourier transform in discrete space, $A(\Omega,\Psi)$,is periodic in both Ω and Ψ.Both periods are 2π

$$A(\Omega + 2\pi j, \Psi + 2\pi k) = A(\Omega,\Psi) \qquad j,k \text{ integers}$$

The energy, E, in a signal can be measured either in the spatial domain or the frequency domain. For a signal with finite energy:

Parseval's theorem (2D continuous space) is

$$E = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} |a(x,y)|^2 dxdy = \frac{1}{4\pi^2}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} |A(u,v)|^2 dudv$$

Parseval's theorem (2D discrete space):

$$E = \sum_{m=-\infty}^{+\infty}\sum_{\pi=-\infty}^{+\infty} |a[m,n]|^2 = \frac{1}{4\pi^2}\int_{-\pi}^{+\pi}\int_{-\pi}^{+\pi} |A(\Omega,\Psi)|^2 \, d\Omega d\Psi$$

This "signal energy' is not to be confused with the physical energy in the phenomenon that produced the signal. If, for example, the value *a[m,n]* represents a photon count, then the

physical energy is proportional to the amplitude 'a', and not the square of the amplitude. This is generally the case in video imaging.

*Given three, two dimensional signals *a, b,* and *c* and their Fourier transform *A, B,* and *C:*

$$c = a \otimes b \quad \xleftrightarrow{F} \quad C = A \bullet B$$

and

$$c = a \bullet b \quad \xleftrightarrow{F} \quad C = \frac{1}{4\pi^2} A \otimes B$$

In words,convolution in the spatial domain is equivalent to multiplication in the Fourier (frequency) domain and vice-versa. This is a central result which provide not only a methodology for the implementation of a convolution but also insight into how two signals interact with each other-under convolution - to produce a third signal. We shall make extensive use of this result later.

* If a two-dimensional signal $a(x,y)$ is scaled in its spatial coordinates then:

$$If \;\; a(x,y) \;\rightarrow\; a(M_x \bullet x, M_y \bullet y)$$
$$Then \;\; A(u,v) \;\rightarrow\; A(u/M_x, v/M_y)/\left|M_x \bullet M_y\right|$$

* If a two-dimensional signal $a(x,y)$ has Fourier spectrum then: $A(u,v)$

$$A(u=0, v=0) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} a(x,y)dxdy$$

$$a(x=0, y=0) = \frac{1}{4\pi^2}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} A(u,v)dxdy$$

· If a two-dimensional signal $a(x,y)$ has Fourier spectrum $A(u,v)$ then:

$$\frac{\partial a(x,y)}{\partial x} \overset{F}{\longleftrightarrow} juA(u,v) \qquad \frac{\partial a(x,y)}{\partial y} \overset{F}{\longleftrightarrow} jvA(u,v)$$

$$\frac{\partial^2 a(x,y)}{\partial x^2} \overset{F}{\longleftrightarrow} -u^2\ A(u,v) \qquad \frac{\partial^2 a(x,y)}{\partial y^2} \overset{F}{\longleftrightarrow} -v^2\ A(u,v)$$

Importance of phase and magnitude

The definition indicates that the Fourier transform of an image can be complex. This is illustrated below in Figure (1 a-c).

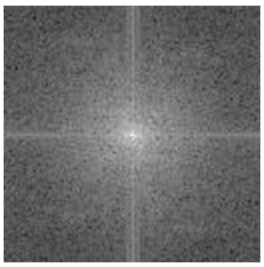
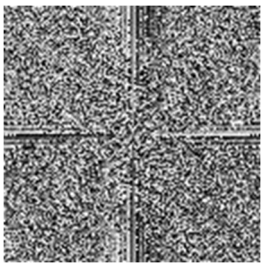
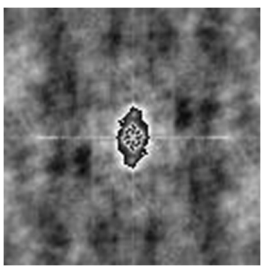




Figure (1a) **Figure(1b)** **Figure(1c)**

Figure (1.4a) shows the original image $a[m,n]$, Figure (1.4b) the magnitude in a scaled form as $\log(|A(\Omega,\Psi)|)$ and Figure (1.4c) the phase $\varphi(\Omega,\Psi)$.

Both the magnitude and the phase functions are necessary for the complete reconstruction of an image from its Fourier transform. Figure(2 a) shows what happens when Figure (1a) is restored solely on the basis of the magnitude information and Figure (2b) shows what happens when Figure (1a) is restored solely on the basis of the phase information.




Figure(2a) **Figure (2b)**

$\varphi(\Omega,\Psi)=0 \quad |A(\Omega,\Psi)|=$ constant

Figure (2a) figure(2b) shows

Neither the magnitude information nor the phase information is sufficient to restore the image. The magnitude-only image Figure (2a) is unrecognizable and has severe dynamic range problems. The phase-only image Figure (2b) is barely recognizable, that is, severely degraded in quality.

Circularly symmetric signals

An arbitrary 2D signal $a(x,y)$ can always be written in a polar coordinate system as $a(r,\theta)$ .When the 2D signal exhibits a circular symmetry this means that:

$$a(x,y)=a(r,\theta)=a(r)$$

where $r^2=x^2+y^2$ and $\tan\theta=y/x$. As a number of physical systems such as lenses exhibit circular symmetry, it is useful to be able to compute an appropriate Fourier representation.

The Fourier transform $A(u,v)$ can be written in polar coordinates $A(\omega_r,\xi)$ and then, for a circularly symmetric signal, rewritten as a *Hankel transform*:

$$A(u,v)=F\{a(x,y)\}=2\pi\int_0^\infty a(r)J_0(w_r r)rdr=A(\omega_r)$$

where $\omega_r{}^2=u^2+v^2$ and $\tan\xi=v/u$ *and* $j_0(*)$ is a Bessel function of the first kind of order zero.

The inverse Hankel transform is given by:

$$a(r)=\frac{1}{2}\int_0^\infty A(w_r)J_o(w_r r)w_r dw_r$$

The Fourier transform of a circularly symmetric 2D signal is a function of only the radial frequency w_r. The dependence on the angular frequency ξ has vanished. Further if $a(x,y)=a(r)$ is real, then it is automatically even due to the circular $A(w_r)$

symmetry. According to equ (1.2), will then be real and even.

2-D Discrete cosine transforms

One disadvantage of the DFT for some applications is that the transform is complex valued, even for real data. A related transform, the discrete cosine transform (DCT), does not have this problem. The DCT is a separate transform and not the real part of the DFT. It is widely used in image and video compression applications, e.g., JPEG and MPEG. It is also possible to use DCT for filtering using a slightly different form of convolution called symmetric convolution.

Definition (2-D DCT)

Assume that the data array has finite rectangular support on $[0, N_1, -1]x[0, N_2 - 1],$ then the 2-D DCT is given as

$$X_C(k_1, k_2) \triangleq \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} 4x(n_1, n_2) \cos\frac{\pi k_1}{2N_1}(2n_1 + 1) \cos\frac{\pi k_2}{2N_2}(2n_2 + 1),$$

for

$$(k_1, k_2) \in [0, N_1 - 1]x[0, N_2 - 1]; Otherwise, X_C(k_1, k_2) \triangleq 0.$$

The DCT basis functions for size 8 x 8 are shown in Figure ( ). The mapping between the mathematical values and the colors (gray levels) is the same as in the DFT case. Each basis function occupies a small square; the squares are then arranged into as 8 x 8 mosaic. Note that unlike the DFT, where the highest frequencies occur near , the $(N_1/2, N_2/2)$, the highest frequencies of the DCT occur at the highest indices $(k_1, k_2) = (7, 7)$.

The inverse DCT exists and is given for $(n_1, n_2) \in [0, N_1 - 1]x[0, N_2 - 1]$ as,

$$x(n_1, n_2) = \frac{1}{N_1 N_2} \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} w(k_1)w(k_2)X_C(k_1, k_2) \cos\frac{\pi k_1}{2N_1}(2n_1 + 1) \cos\frac{\pi k_2}{2N_2}(2n_2 + 1),$$

where the weighting function *w(k)* is given just as in the case of 1-D DCT by

$$w(k) \triangleq \begin{cases} 1/2 & k = 0 \\ 1, & k \neq 0. \end{cases}$$

From eqn, we see that the 2-D DCT is a separable operator. As such it can be applied to the rows and then the columns, or vice versa. Thus the 2-D theory can be developed by repeated application of the 1-D theory. In the following subsections we relate the 1-D DCT to 1-D DFT of a symmetrically extended sequence. This not only provides an understanding of the DCT but also enables its fast calculation. We also present a fast DCT calculation that can avoid the use of complex arithmetic in the usual case where x is a real-valued signal, e.g., an image. (Note: the next two subsections can be skipped by the reader familiar with the 1-D DCT)

Reference : NPTEL

Question :

1. Why Image transformation is required in digital image processing ?
2. What is the advantage of FFT over DFT ?
3. Why DCT is used in image processing ?

Module: 2: IMAGE ENHANCEMENT

2.1 **Digital Image Enhancement:** Importance of Digital Image enhancement

Importance and necessity of digital image processing stems from two principal application areas: the rst being the Improvement of pictorial information for human interpretation and the second being the Processing of a scene data for an autonomous machine perception. Digital image processing has a broad range of applications such as remote sensing, image and data storage for transmission in business applications, medical imaging, acoustic imaging, Forensic sciences and industrial automation. Images acquired by satellites are useful in tracking of earth resources, geographical mapping, and prediction of agricultural

crops, urban population, weather forecasting and re control. Space imaging applications include recognition and analyzation of objects contained in images obtained from deep space-probe missions. There are also medical applications such as processing of X-Rays, Ultrasonic scanning, Electron micrographs, Magnetic Resonance Imaging, Nuclear Magnetic Resonance Imaging, etc.

In addition to the above mentioned applications, digital image processing is now being used to solve a wide variety of problems. Though unrelated, these problems commonly require methods capable of enhancing information for human visual interpretation and analysis. The Image processing Procedures such as Image enhancement and restoration are used to process degraded or blurred images. Successful applications of image processing concepts are found in astronomy, defense, biology, medical and industrial applications. As per Medical Imaging is concerned most of the images may be used in the detection of tumors or for screening the patients. The current major area of application of digital image processing (DIP) techniques is in solving the problem of machine vision so as to attain good results.

The principle objective of image enhancement technique is to process an image so that the resultant image is more suitable than the original for a particular application. Most of the enhancement techniques are very much problem oriented and hence enhancement for one application may turn out to be degradation for the other. Enhancement approaches may be classi ed especially into two broad categories.

1. Spatial domain enhancement techniques
2. Frequency domain enhancement techniques.

The former technique refers to process the image in the image plane (pixels) itself while the latter techniques are based on modifying the transform (Fourier or any other) of an image. In most of the general enhancement techniques for problems involve various combinations of methods from both the categories. Some examples of enhancement operations are edge enhancement, pseudo- coloring, histogram equalization(HE), contrast stretching, noise ltering, un-sharp masking, sharpening, magnifying, etc. Usually the enhancement process does not increase the inherent information content present in the image but only tries to present it in a suitable manner for easy assessment. These image enhancement operations may be either local or global. Global operations work on the entire image at a time while local operations de ne spatial masks i.e., on small sub- images over which the operation is to be performed.

2.2 enhancement in spatial and frequency domain

2.2.1 enhancement in spatial domain

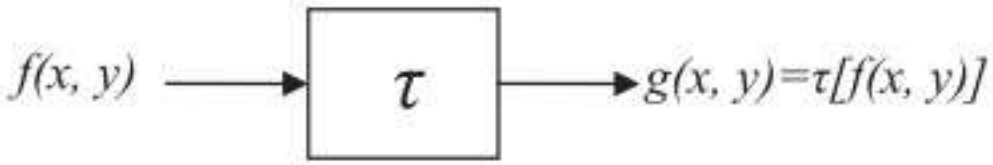


Fig. 2.1 The original image *f*(*x, y*) with pixel intensity *r* is processed through "*τ*" to get the enhanced image *g*(*x, y*) with pixel intensity *s*

Broadly, the image enhancement in the spatial domain is divided into four categories:

1. Contrast manipulation/intensity transformation 2. Image smoothing
3. Image sharpening
4. Image resampling

In the current chapter, processing of images through histogram, the intensity distribution is presented after a brief discussion on basic gray-level transformation. Histogram is presented here in terms of probability distribution function (PDF). For histogram-based image enhancement, the process of linearizing the cumulative density function (CDF) is described in the process of histogram equalization. Next, the process of image filtering in spatial domain is introduced with the help of convolution and correlation. Finally, resampling of images in order to achieve visually enhanced image after geometric transformations is discussed. First, the basic interpolation techniques are discussed in 1D in terms of B-splines, then the same concepts are extended to 2D for image interpolation/resampling (Fig. 2.1).

2.1.1.1 Intensity Transformations

Spatial domain refers to an aggregate of pixels consisting in an image. Spatial do- main image processing is processing over the pixels directly as expressed in the following equation:

$$g(x, y)=\tau[f(x, y)] \qquad (2.1)$$

where $f(x, y)$ and $g(x, y)$ are the input and the processed image through the mathematical mapping "τ" defined over (x, y). When this mapping/operator is applied on any arbitrary point of coordinate (x, y) to get the processed point at the same coordinate (x, y), this mathematical mapping "τ" is called as intensity operator or intensity mapping or gray-level transformation. If r and s be the intensity of the arbitrary point before and after transformation, Eq. 2.1 can be rewritten as

$$s =\tau[r]$$

(2.2)

Patterns/shapes of different intensity transformations are discussed with their effect on the images. In Fig. 2.2, different gray-level transformation characteristics are shown in terms of graphs (Fig. 2.2b, d, f, h).

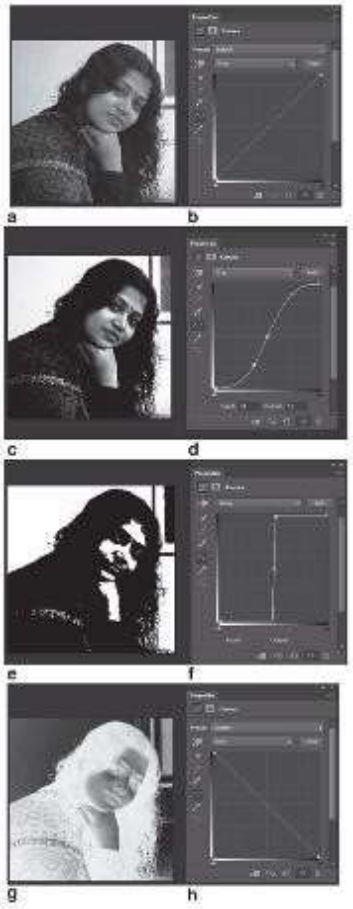


Fig. 2.2 Intensity (*gray-level*) transformations: **a** and **b** linear transformation, **c** and **d** contrast stretching, **e** and **f** thresholding for binarization, **g** and **h** negative transform

2.1.1.1.1. Linear Transformation

In the transformation characteristics, the *x*-axes and *y*-axes of the graphs represent gray levels (intensity levels) of input original image and transformed image, respec- tively, ranging from 0 to 255. Hence, one linear characteristic (unit ramp function) just maps the intensity of the input image to the same intensity of the transformed image. It maintains all the intensities unmodified as shown in Fig. 2.2a and b.

2.1.1.1.2 Contrast Stretching and Thresholding

Referring to the graph of Fig. 2.2d, the transformation clearly signifies that the contrast of the image is increased by mapping all the intensities higher than 128 to a compressed narrow range of intensities near white, and all the intensities less than 128 to a compressed narrow range of intensities near black, as shown in Fig. 2.2c. This transformation is therefore called as contrast stretching. In the limiting case where the higher intensities (

r>128) are mapped to the highest intensity (s=255) and lower intensities (r < 128) are mapped to the lowest intensity (s = 0), the trans- formation (Fig. 2.2f) takes the form of thresholding for binarization, as the output image would have only two intensities (binary image)—white and black (Fig. 2.2e).

2.1.1.1.3 Negative Intensity Transform

In this case, the slope of the transformation is made −1 instead of + 1 with respect to linear transformation, as shown in Fig. 2.2h. This *negative transformation* maps the intensities with the rule:

$$s = 255 - r \quad (2.3)$$

This transformation generates a complete negative image (as shown in Fig. 2.2g) by mapping the higher intensities to lower intensity and vice versa. Negative intensity transformation is very useful in specialized applications where the intensity details are embedded/hidden in dark regions of an image. Digi- tal mammogram is a method of analyzing lesions inside breast tissues. To analyze the breast tissue in digital mammogram, negative intensity transformation is very useful [4]. From Fig. 2.3, it is clearly observed that the tissues can be analyzed from the negative image even by visual attack. Note that here the content of information is exactly the same in both the original and transformed image, the representation of information has been changed for the ease of analysis.

2.1.1.1.4 Logarithmic Intensity Transformation

The logarithmic intensity transformation is defined by the following equation:

$$s = c \log(1 + r) \quad (2.4)$$

In the equation, c is an arbitrary positive constant, r and s are the intensities of the original and transformed images, respectively, with intensity profile 0 through 255. This intensity transformation maps a narrow range of lower intensity value in the original input image to a wider range of output levels and a wider range of higher intensity values to a narrower range of output gray levels. Hence, this transformation would be useful where expansion of dark pixel and compression of brighter pixels are required. The behavior of inverse logarithmic intensity transformation is exactly opposite. Any transformation characteristic curve with the same pattern will behave exactly as the log transformation does, i.e., expanding one part of the intensity profile and compressing the other. Power-law transformation, which is discussed in the next subsection, is a more generalized transformation to achieve this kind of behavior in transformation

Fig. 2.3 Digital mammogram: **a** original image, **b** negative intensity transformed image

2.1.1.1.5 Power-Law Intensity Transform and Gamma Correction

The power-law intensity transformation [10] is defined by the following equation:

$$s = cr^{\gamma} \quad (2.5)$$

In the equation, c and γ are the arbitrary positive constants, r and s are the intensi- ties of the original and transformed images, respectively, with the intensity profile 0 through 255. This intensity transformation characteristic is shown in Fig. 2.4 with varied γ . Like logarithmic intensity transformation, the power-law transformation with fractional γ value maps a narrow range of dark values (low intensity) to a wider range of output values and

wider range of lighter values (high intensity) to a narrower range of output. Moreover, in the power-law transformation, we can tune the characteristic curve by tuning γ , and hence we can change the narrowness of darker intensities and wideness of the lighter intensities of the input image. As understood, with γ = 1 unit ramp characteristic would be realized which is identity transformation and γ > 1 will have exactly an opposite effect with respect to frac- tional (γ < 1) values of γ .

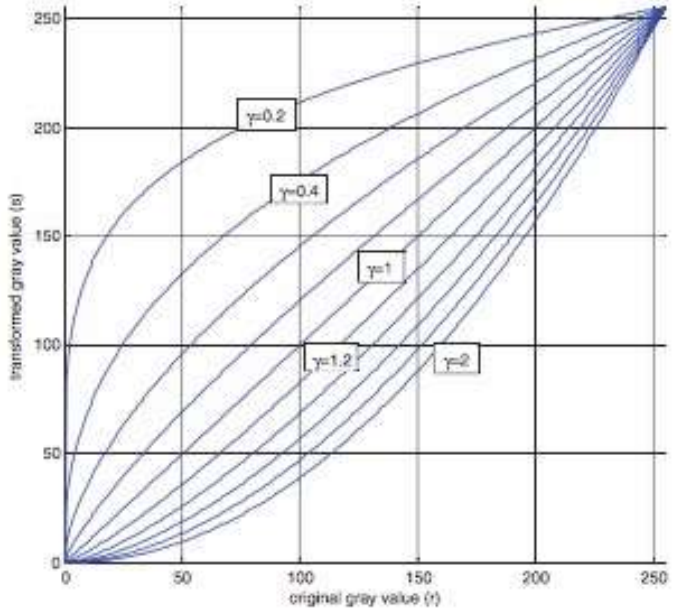


Fig. 2.4 Transfer characteristics for power-law intensity transformation

It is to be noted that, simply by applying the power-law equation, the dynamic intensity range of the transformed image would not be exactly [0 255]. To achieve this, the transformed intensities need to be normalized with respect to maximum intensity as depicted in the following equation.

$$s = s[]. (255 / \max(s[])); \quad (2.6)$$

A number of devices used for image acquisition, printing, and display respond ac- cording to power law. Hence, based on the exponent or the tuning parameter γ (gamma), the procedure of correcting the power-law phenomena is called as gamma correction. The cathode ray tube (CRT) works depending on the intensity to voltage response which is a power-law relationship. For CRT, the exponent gamma varies from 1.8 to 2.5. Considering an arbitrary value of γ in this range (say, γ = 2.5), we can understand the system behavior. From the power-law characteristics (Fig. 2.4), we can understand that the wider band of lower intensity values is mapped to a nar- rower band of lower intensity, which generates a darker image with respect to the

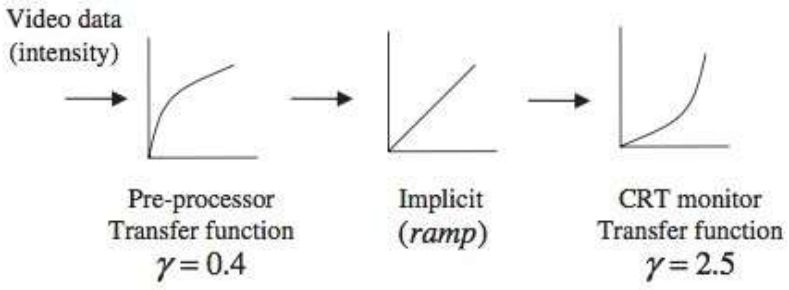


Fig. 2.5 Gamma correction for CRT monitor

original one. To handle this, we need to add a preprocessor transfer function of inverse characteristics (i.e., s = r 2.5 r 0.4=) before sending the signal to the CRT. The combination of these two transfer functions (transfer functions of preprocessor unit and that of the CRT) will realize a unit ramp function, which ensures the realization of the original gray intensity values as depicted in Fig. 2.5.

2.3 Bit plane slicing:

The gray level of each pixel in a digital image is stored as one or more bytes in a computer. For an 8-bit image, 0 is encoded as 00000000 and 255 is encoded as 11111111. Any number between 0 t0 255 is encoded as one byte. The bit in the far left side is referred as the most significant bit (MSB) because a change in that bit would significantly change the value encoded by the byte. The bit in the far right is referred as the least significant bit (LSB), because a change in this bit does not change the encoded gray value much. The bit plane representation of an eight-bit digital image is given by:

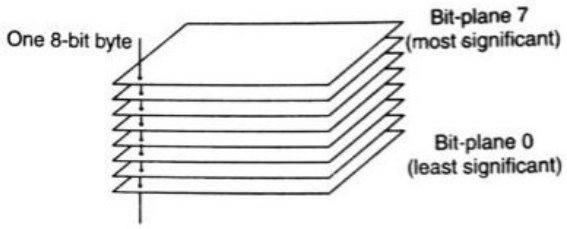


Fig. 2.6 Bit plane slicing

Bit plane slicing is a method of representing an image with one or more bits of the byte used for each pixel. One can use only MSB to represent the pixel, which reduces the original gray level to a binary image. The three main goals of bit plane slicing is:

- Converting a gray level image to a binary image.
- Representing an image with fewer bits and corresponding the image to a smaller size
- Enhancing the image by focussing.

6	7	6	6	7
0	0	0	1	2
1	1	1	2	3
4	5	5	4	2
6	6	6	7	7

Fig. 2.7 Original gray level intensity.

110	111	110	110	111
000	000	000	001	010
001	001	001	010	011
100	101	101	100	010
110	110	110	111	111

Fig. 2.8 3-bit binary coding of fig 2.6

1	1	1	1	1
0	0	0	0	0
0	0	0	0	0
1	1	1	1	0
1	1	1	1	1

MSB plane

1	1	1	1	1
0	0	0	0	1
0	0	0	1	1
0	0	0	0	1
1	1	1	1	1

Centre bit plane

0	1	0	0	1
0	0	0	1	0
1	1	1	0	1
0	1	1	0	0
0	0	0	1	1

LSB plane

Fig 2.9 Each plane matrix

2.4 Histogram, Histogram Equalization :

Histogram equalization is used to enhance contrast. It is not necessary that contrast will always be increase in this. There may be some cases were histogram equalization can be worse. In that cases the contrast is decreased.Lets start histogram equalization by taking this image below as a simple image.

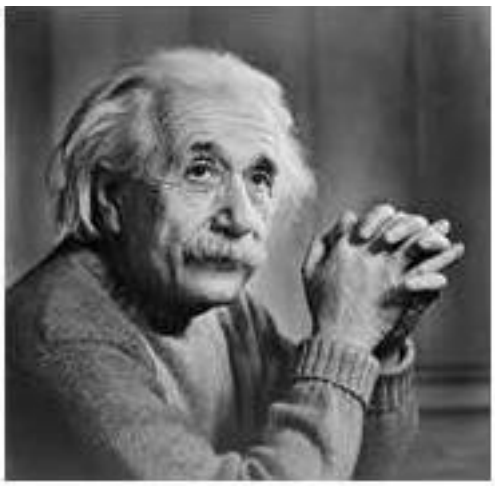


Fig 2.10 Original Image

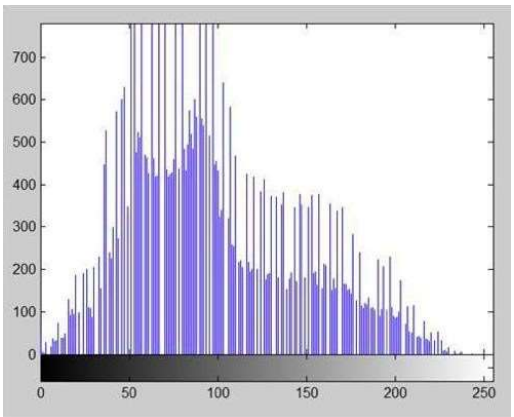


Fig. 2-.11 Histogram of fig

2.4.1 Calculate CDF Calculate according to gray levels

Lets for instance consider this , that the CDF(Cumulative Distribution Function) calculated in the second step looks like this.

Gray Level Value	CDF
0	0.11
1	0.22
2	0.55
3	0.66
4	0.77
5	0.88
6	0.99
7	1

Table 2.1

Then in this step you will multiply the CDF value with (Gray levels (minus) 1) .Considering we have an 3 bpp image. Then number of levels we have are 8. And 1 subtracts 8 is 7. So we multiply CDF by 7. Here what we got after multiplying.

Gray Level Value	CDF	CDF * (Levels-1)
0	0.11	0
1	0.22	1
2	0.55	3
3	0.66	4
4	0.77	5
5	0.88	6
6	0.99	6
7	1	7

Table 2.2

Now we have is the last step, in which we have to map the new gray level values into number of pixels.Lets assume our old gray levels values has these number of pixels.

Table 3

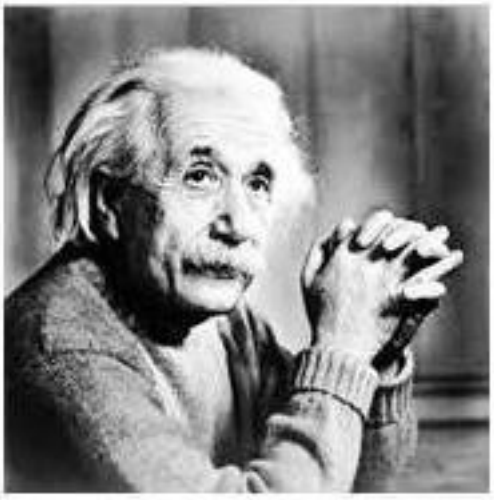
Gray Level Value	Frequency
0	2
1	4
2	6
3	8
4	10
5	12
6	14
7	16

Now if we map our new values to , then this is what we got.

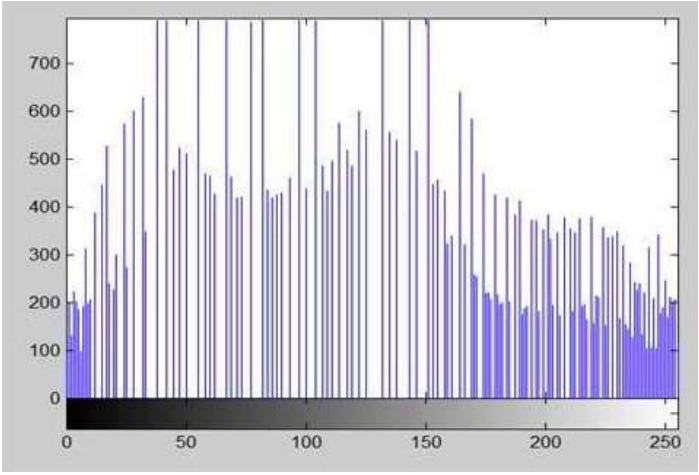
Table 4

Gray Level Value	New Gray Level Value	Frequency
0	0	2
1	1	4
2	3	6
3	4	8
4	5	10
5	6	12
6	6	14
7	7	16

Now map these new values you are onto histogram, and you are done.Lets apply this technique to our original image. After applying we got the following image and its following histogram.



Fig



2.12 Histogram Equalized Image.

Fig 2.13 Histogram of fig 2.12

As you can clearly see from the images that the new image contrast has been enhanced and its histogram has also been equalized. There is also one important thing to be note here that during histogram equalization the overall shape of the histogram changes, where as in histogram stretching the overall shape of histogram remains same.

2.5. Mean and Median filtering in Digital Images:

2.5.1 Mean filter

The mean filter is a simple sliding-window spatial filter that replaces the center value in the window with the average (mean) of all the pixel values in the window. The window, or kernel, is usually square but can be any shape. An example of mean filtering of a single 3x3 window of values is shown below.

5	3	6
2	1	9
8	4	7

Fig. 2.14

5+3+ 6 + 2 + 1 + 9 + 8 + 4 + 7 = 45
45 / 9 = 5

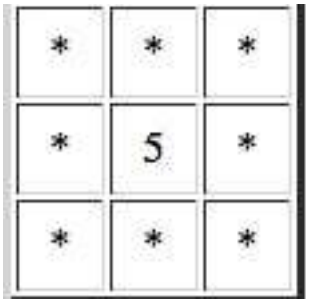


Fig. 2.15

Center value (previously 1) is replaced by the mean of all nine values (5).

2.5.2 Median filter

The median filter is also a sliding-window spatial filter, but it replaces the center value in the window with the median of all the pixel values in the window. As for the mean filter, the kernel is usually square but can be any shape. An example of median filtering of a single 3x3 window of values is shown below.

Note that for the first (top) example, the median filter would also return a value of 5, since the ordered values are 1, 2, 3, 4, **5**, 6, 7, 8, 9. For the second (bottom) example, though, the mean filter returns the value 16 since the sum of the nine values in the window is 144 and 144 / 9 = 16. This illustrates one of the celebrated features of the median filter: its ability to remove 'impulse' noise (outlying values, either high or low). The median filter is also widely claimed to be 'edge-preserving' since it theoretically preserves step edges without blurring. However, in the presence of noise it does blur edges in images slightly.

Unfiltered values

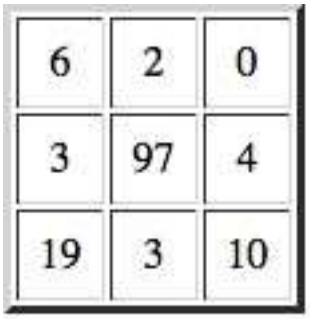


Fig. 2.16

in order:

0, 2, 3, 3, **4**, 6, 10, 15, 97

median filtered

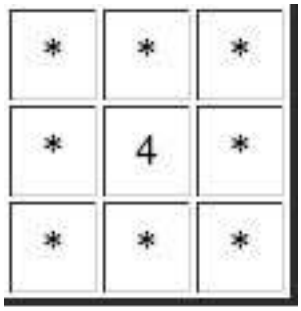


Fig. 2.17

Center value (previously 97) is replaced by the median of all nine values (4).

2.6 Frequency domain filtering in Digital Images :

Frequency domain is basically a space defined by Fourier transform. Fourier transforms has wide applications in image processing, such as image analysis, image filtering, image reconstruction and image compression [9]. In frequency domain analysis, it indicates that how signal energy is distributed over a range of frequencies. The basic principle of the frequency domain image filtering consists of computing a 2-D discrete Fourier transform of the image, for instance the 2-D DFT, manipulating the transform coefficients by an operator i.e. filter function and then performing the inverse discrete Fourier transform.

2.6.1 LPF, HPF and BPF

2.6.1.1 Low pass filter perform smoothing or blurring of images. It is achieved in frequency domain by attenuation of high frequency or high intensity transition. The output of low pass filter contains small or less intensity transition in to a particular group of pixels. We implement Ideal low pass filter(ILPF), Butterworth low pass filter (BLPF) and Gaussian low pass filter (GLPF) on MATLAB platform and analyze it.

Ideal low pass filter (ILPF)-ILPF gives ideal response for image filtering. It passes all the frequency within a circle of radius D0 which is cutoff frequency of ILPF, while attenuates all the frequency lies outside this circle. It is specified by the function in equation 2.7

2.7

$$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \leq D0 \\ 0 & \text{if } D(u,v) > D0 \end{cases}$$

Where D(u,v) is the distance between center of the frequency rectangle and a point (u,v) in the frequency domain image calculated by equation 2.8.

2.8 $$D(u,v) = \sqrt{\left(u - \frac{p}{2}\right)^2 + \left(v - \frac{Q}{2}\right)^2}$$

$\left[\frac{P}{2}, \frac{Q}{2}\right]$

Defines the center of circle of radius D0 .

2.6.1.2 **High pass filter** High pass filter performs sharpening of images. It is achieved in frequency domain by attenuation of low frequency or low intensity transition. The output of high pass filter contains high intensity transition in to a particular group of pixels. We implement Ideal high pass filter (IHPF), Butterworth high pass filter (BHPF) and Gaussian high pass filter (GHPF) on MATLAB platform and analyze it.

Ideal high pass filter (IHPF)-The filter function of IHPF is given by equation 2.9

2.9 $$H(u,v) = \begin{cases} 0 & \text{if } D(u,v) \leq D0 \\ 1 & \text{if } D(u,v) > D0 \end{cases}$$

It passes all the frequency outside the circle of radius D0 which is cutoff frequency of IHPF, while attenuates all the frequency lies within this circle. D(u,v) and D0 has same meaning as defined .

Module 3: IMAGE COMPRESSION

Image compression is an application of data compression that encodes the original image with few bits. The objective of image compression is to reduce the redundancy of the image and to store or transmit data in an efficient form. Fig 1.1 shows the block diagram of the general image storage system. The main goal of such system is to reduce the storage quantity as much as possible, and the decoded image displayed in the monitor can be similar to the original image as much as can be. The essence of each block will be introduced in the following sections.

3.1 $$Compression\ ratio = \frac{Uncompressed\ size}{Compressed\ size}$$

3.1 Importance of Digital Image Compression

The objective of image compression is to reduce the redundancy of the image and to store or transmit data in an efficient form.Cost for transmitting an image as data reduces at much extent as cost depends upon duration for which data is being transmitted.It saves computing

power as execution of image transmission takes very less time if the size is lesser.It reduces the transmission errors since fewer bits are transferred.Secure level of transmission is possible due to encoding and compressing the image.

3.2. Types of Image Compression, example of lossless and lossy compression,

1.Lossy techniques
2. Lossless techniques

3.2.1. Lossy techniques-Lossy compression methods have larger compression ratios as compared to the lossless compression techniques. Lossy methods are used for most applications. By this the output image that is reconstructed image is not exact copy but somehow resembles it at larger portion.

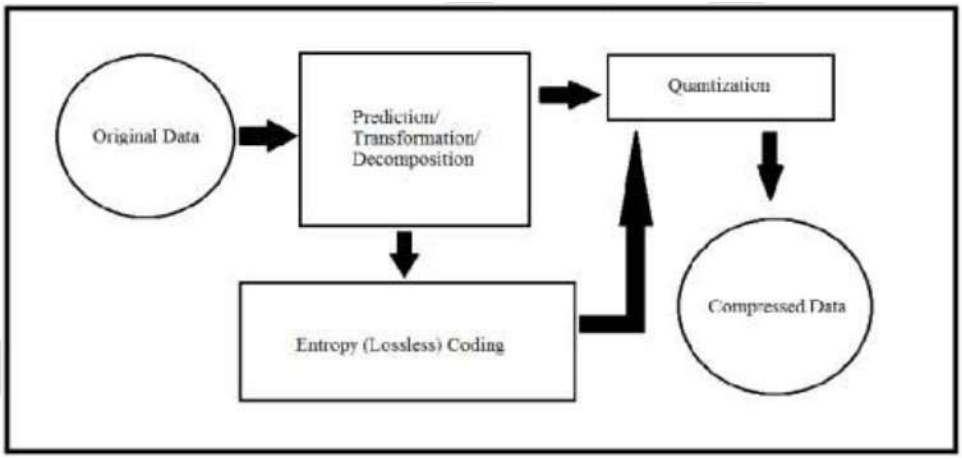


Fig. 3.1 Block diagram of Lossy Image Compression

As shown in Fig.3, this prediction – transformation – decomposition process is completely reversible. There is loss of information due to process of quantization. The entropy coding after the quantizing, is lossless. When decoder has input data, entropy decoding is applied to compressed signal values to get the quantized signal values. Then, de-quantization is used on it and the image is recovered which resembles to the original [5].
Lossy compression methods include some basic consideration (performance wise):
1. Speed of encoding and decoding 2.
Compression ratio
3. SNR ratio.
Lossy compression includes following methods:
1. Block truncation coding
2. Code Vector quantization
3. Fractal coding
4. Transform coding
5. Sub-band coding

3.2.2 Lossless techniques

It is also known as entropy coding as it uses decomposition techniques to minimize loopholes. The original image can be perfectly recovered from the compressed image, in lossless compression techniques. These do not add noise to the signal. It is also known as entropy coding as it uses decomposition techniques to minimize redundancy. Following techniques are included in lossless compression:

1. Huffman encoding
2. Run length encoding
3. LZW coding
4. Area coding

3.3. Image compression standards,

Images are very important documents nowadays; to work with them in some applications they need to be compressed, more or less depending on the purpose of the application. There are some algorithms that perform this compression in different ways; some are lossless and keep the same information as the original image, some others loss information when compressing the image. Some of these compression methods are designed for specific kinds of images, so they will not be so good for other kinds of images. Some algorithms even let you change parameters they use to adjust the compression better to the image.

Lossless image representation formats:
BMP (bitmap) is a bitmapped graphics format used internally by the Microsoft Windows graphics subsystem (GDI), and used commonly as a simple graphics file format on that platform. It is an uncompressed format.
PNG (Portable Network Graphics) (1996) is a bitmap image format that employs lossless data compression. PNG was created to both improve upon and replace the GIF format with an image file format that does not require a patent license to use. It uses the DEFLATE compression algorithm, that uses a combination of the LZ77 algorithm and Huffman coding.
PNG supports palette based (with a palette defined in terms of the 24 bit RGB colors), greyscale and RGB images. PNG was designed for distribution of images on the internet not for professional graphics and as such other color spaces.

TIFF (Tagged Image File Format) (last review 1992) is a file format for mainly storing images, including photographs and line art. It is one of the most popular and flexible of the current public domain raster file formats. Originally created by the company Aldus, jointly with Microsoft, for use with PostScript printing, TIFF is a popular format for high color depth images, along with JPEG and PNG. TIFF format is widely supported by image-

manipulation applications, and by scanning, faxing, word processing, optical character recognition, and other applications.

Lossy image compression formats:
JPEG (Joint Photographic Experts Group) (1992) is an algorithm designed to compress images with 24 bits depth or greyscale images. It is a lossy compression algorithm. One of the characteristics that make the algorithm very flexible is that the compression rate can be adjusted. If we compress a lot, more information will be lost, but the result image size will be smaller. With a smaller compression rate we obtain a better quality, but the size of the resulting image will be bigger. This compression consists in making the coefficients in the quantization matrix bigger when we want more compression, and smaller when we want less compression.
The algorithm is based in two visual effects of the human visual system. First, humans are more sensitive to the luminance than to the chrominance. Second, humans are more sensitive to changes in homogeneous areas, than in areas where there is morevariation (higher frequencies). JPEG is the most used format for storing and transmitting images in Internet.
JPEG 2000 (Joint Photographic Experts Group 2000) is a wavelet-based image compression standard. It was created by the Joint Photographic Experts Group committee with the intention of superseding their original discrete cosine transform- based JPEG standard.
JPEG 2000 has higher compression ratios than JPEG. It does not suffer from the uniform blocks, so characteristics of JPEG images with very high compression rates. But it usually makes the image more blurred that JPEG.

3.4. Compression in spatial domain:

Bit Plane Slicing-Discussed in 2.3

3.4.1. compression using Huffman coding:

Huffman coding is one of the basic compression methods, that have proven useful in image and video compression standards. When applying Huffman encoding technique on an Image, the source symbols can be either pixel intensities of the Image, or the output of an intensity mapping function.Huffman coding[24] is based on the frequency of occurrence of a data item (pixel in images). The principle is to use a lower number of bits to encode the data that occurs more frequently. Codes are stored in a Code Book which may be constructed for each image or a set of images. In all cases the code book plus encoded data must be transmitted to enable decoding.

Example of Huffman Coding:

Input: Six symbols and their respective probability / frequency values

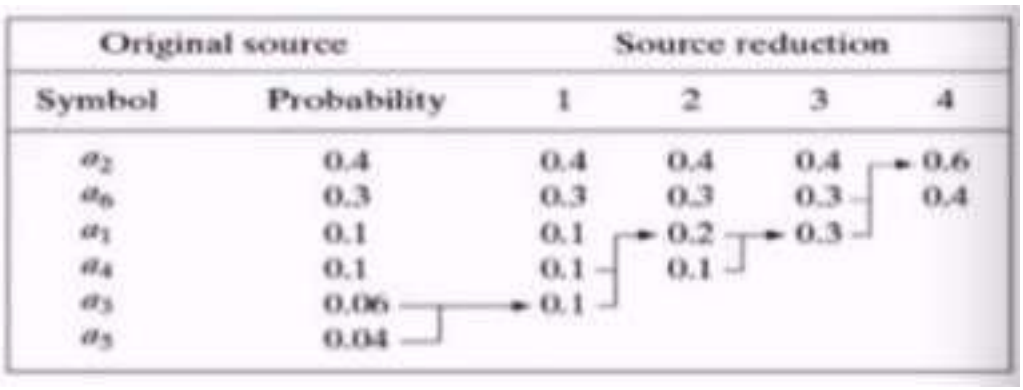


Fig 3.2 Huffman example

Steps:

1. Arrange the symbols in descending order of frequency values.
2. Add the two lowest values and remove the symbols
3. At the end we will be left with only two values

3.4.2. DCT (Discrete Cosine Transform)

Discrete cosine transform (DCT) developed by Ahmed, Natarajan, and Rao [1974], the DCT is a close rela- tive of the discrete Fourier transform (DFT). Its application to image compression was pioneered by Chen and Pratt [1984]. In this article, I will develop some simple functions to compute the DCT and show how it is used for image com- pression. We have used these functions in our laboratory to explore methods of optimizing image compression for the human viewer, using information about the human visual system [Watson 1993]. The goal of this paper is to illustrate the use of Mathematica in image processing and to provide the reader with the basic tools for further exploration of this subject. DCT separates images into parts of different frequencies where less important frequencies are discarded through quantization and important frequencies are used to retrieve the image during decompression. Compared to other input dependent transforms, DCT has many advantages: (1) It has been implemented in single integrated circuit; (2) It has the ability to pack most information in fewest coefficients; (3) It minimizes the block like appearance called blocking artifact that results when boundaries between sub-images become visible[11].
The forward 2D_DCT transformation is given by the following equation:

$$C(u,v)=D(u)D(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos[(2x+1)u\pi/2N]\cos[(2y+1)v\pi/2N]$$

Where,u,v=0,1,2,3,……,N-1

The inverse 2D-DCT transformation is given by the following equation:

$$f(x,y)=\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} D(u)D(v)D(u,v)\cos[(2x+1)u\pi/2N]x\cos(2y+1)v\pi/2N]$$

where

D(u)=(1/N) ^1/2 for u=0

D(u)=2(/N)^1/2 for u=1,2,3……..,(N-1)

3.4.3 Wavelet based Digital image compression

Wavelets are mathematical functions that scratch up data into altered occurrence components, and then study each section with a pledge matched to its scale. Wavelets were developed autonomously in the arenas of mathematics, quantum physics, electrical engineering, Image processing and so on. Wavelets are functions defined over a limited interval and having a standard value of nil. The basic idea of the wavelet transform is to signify any arbitrary function (t) as a superposition of a rest of such wavelets or basis functions. These origin functions or infant wavelets are obtained from a single archetype wavelet called the mother wavelet, by dilations or else contractions and translations. The DWT of a controlled length signal x(n) having N components, for example, is uttered by an N x N matrix. Wavelet transform has become an important method for image compression. Using a wavelet transform, the wavelet compression methods are satisfactory for representing transients, such as percussion sounds in audio, or highfrequency gears in two-dimensional images. Wavelet-coding schemes are particularly suitable for applications where scalability and tolerable humiliation are important. Recently the JPEG group has released its fresh image coding standard, JPEG-2000, which has been based upon DWT. Wavelet transform provide substantial improvement in picture quality at higher compression ratio. It also known as discrete wavelet transforms (DWT). This transform organize the image information into a uninterrupted wave, typically with many peaks and dips and centers it on zero. The Discrete Wavelet Transform of a limited extent signal x(n) having N components, is uttered by an N x N matrix. The DWT provides sufficient information for the analysis and synthesis of a signal, but is advantageously;

much more efficient. Discrete Wavelet analysis is computed using the concept of filter banks. Filters of different cut-off frequencies analyse the signal at different scales. Resolution is changed by the filtering; the scale is transformed by up sampling and down sampling. If a signal is put through two filters: High-pass filter, high frequency information is kept, low frequency information is lost. Low pass filter, low frequency information is kept, high frequency information is lost. Then the signal is effectively decomposed into two parts, a detailed part (high frequency), and an approximation part (low frequency). The filtering is done for each column of the transitional data. The resulting two- dimensional collection of coefficients contains four bands (sub bands) of data, each labelled as LL (low-low): low sub bands for row and column filtering, HL(high-low): high sub bands for row filtering and low sub bands for column filtering, LH (low-high): low sub bands for row filtering and highsubbands for columns filtering and HH (high- high): high sub bands for row and column filtering.

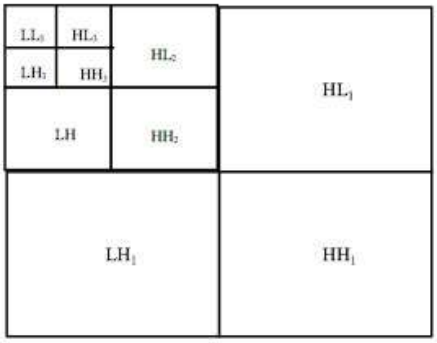


Fig. 3.3 Subband decomposition

MCQ:

1.Compressed image can be recovered back by
a. image enhancement
b.image decompression
c.image contrast
d.image equalization Answer: (b).
image decompression

2. Image compression comprised of
a. encoder
b. decoder
c. frames
d. Both a and b Answer: (d).

3. Coding redundancy works on b.matrix
a.pixels c.intensity

d.coordinates

Answer: (c).

4. Sequence of code assigned is called
a. code word
b. word
c. byte
d. nibble Answer: (a)

5. Sample Questions:

What is image compression?
What is the need for Compression?
What are different Compression Methods?
Define is coding redundancy?
What is JPEG?

Module 4:Digital Image Restoration

Restoration: A process that uses a prior knowledge to re- construct the image that has been degraded.

Restoration is a process that attempts to reconstruct or recover an image that has been degraded by using a priori knowledge of the degradation phenomenon. Similar to the enhancement technique, the function of the restoration technique is to improve the image. Restoration techniques are based on modeling the degradation us- ing a priori knowledge and applying the inverse process in order to restore the original image.

Restoration techniques involve formulating certain criteria of goodness that gives the optimal estimate of the desired result. On the other hand, enhancement techniques are heuristic procedures to manipulate the given image to produce the most pleasing image, in order to take advantage of psycho-physical aspects of human visual system. For example, contrast stretching is considered as an enhancement technique because it is primarily based on the pleasing aspects it might present to the observer, whereas removal of image blur by using a de-blurring function is considered as a restoration technique.

DEGRADATION MODEL

Figure 1 shows the degradation process which is modeled as an operator H, with an additive noise term $\eta(x, y)$ operating on an in- put image $f(x, y)$ to produce a degraded image $g(x, y)$.

$$g(x, y) = H[f(x, y)] + \eta(x, y)$$
…..(1)

The input–output relationship of the model is given as in equation (1)

If $H[K_1 f_1(x, y) + K_2 f_2(x, y)] = K_1 H[f_1(x, y)] + K_2 H[f_2(x, y)]$ (7 …..(2) For simplicity assume η(x, y) = 0 such that g(x, y) = H[f(x, y)] then the operator H is said to be linear where k1 and k2 are con- stants and f1 (x, y) and f2 (x, y) are two input images. If K1 = K2 = 1 then equation (2) can be written as

$$H[f_1(x, y) + f_2(x, y)] = H[f_1(x, y)] + H[f_2(x, y)] \qquad (7.3)$$

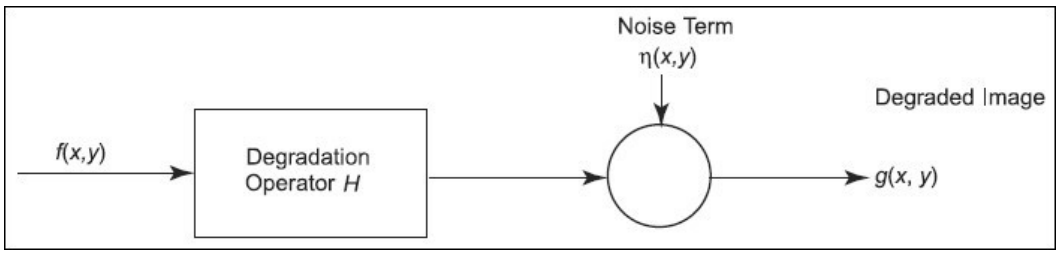


FIGURE 1 A model for image degradation process

From the equation 7.3 we infer that if H is a linear operator, then the response to a sum of two inputs is equal to the sum of two responses and this is called additivity property. If we assume f2 (x, y)= 0 equation (2) reduces to

$$H[K_1 f_1(x, y)] = K_1 H[f_1(x, y)]$$

…..(4) which is known as the property of homogenity.

Linear Operator: An operator is said to be linear if the response to a sum of two inputs is equal to the sum of two response.

constant. From this discussion one can conclude that the linear operator possesses both the property of additivity and homogeneity.

For any image f(x, y) and any α, β values, if

$$H[f(x - \alpha, y - \beta)] = g(x - \alpha, y - \beta) \qquad \text{...(5)}$$

then g(x, y) = H[f(x, y)] is said to be position invariant.

Position Invariant: An operator is said to be position in- variant if it satisfies both additivity and homogeneity properties.

Inverse Filtering

If we know of or can create a good model of the blurring function that corrupted an image, the quickest and easiest way to restore that is by inverse filtering. Unfortunately, since the inverse filter is a form of high pass filer, inverse filtering responds very badly to any noise

that is present in the image because noise tends to be high frequency. In this section, we explore two methods of inverse filtering - a thresholding method and an iterative method.

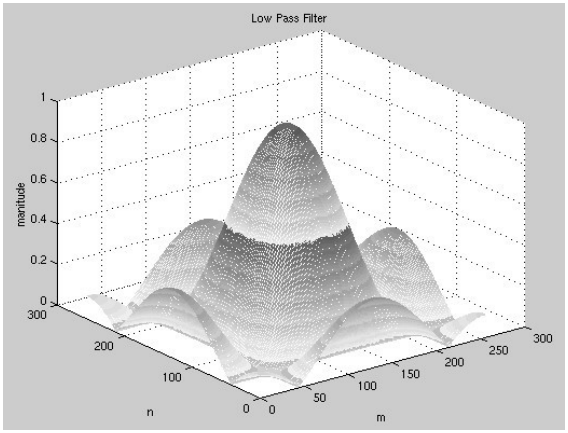
Method 1: Thresholding

We can model a blurred image by

$$f(n_1, n_2) = g(n_1, n_2) * *h(n_1, n_2)$$

where *f* is the original image, *b* is some kind of a low pass filter and *g* is our blurred image. So to get back the original image, we would just have to convolve our blurred function with some kind of a high pass filter

But how do we find h? If we take the DFT of *b* so that **B**=DFT2(*b*), we would get something that looks like this



In the ideal case, we would just invert all the elements of **B** to get a high pass filter. However, notice that a lot of the elements in **B** have values either at zero or very close to it. Inverting these elements would give us either infinities or some extremely high values. In order to avoid these values, we will need to set some sort of a threshold on the inverted element. So instead of making a full inverse out of **B**, we can an "almost" full inverse by

$$H(\omega_1, \omega_2) = \begin{cases} \frac{1}{B(\omega_1,\omega_2)} & \text{if } \frac{1}{B(\omega_1,\omega_2)} < \gamma \\ \gamma & \text{else} \end{cases}$$

So the higher we set γ, the closer **H** is to the full inverse filter.

Implementation and Results

Since Matlab does not deal well with infinity, we had to threshold **B** before we took the inverse. So we did the following:

$$B(\omega_1, \omega_2) = \begin{cases} B(\omega_1, \omega_2) & \text{if } B(\omega_1, \omega_2) > n \\ n & \text{else} \end{cases}$$

where n is essentially and is set arbitrarily close to zero for noiseless cases. The following images shows our results for n=0.0001.

We see that the image is almost exactly like the original. The MSE is 2.5847.

Because an inverse filter is a high pass filter, it does not perform well in the presence of noise. There is a definite tradeoff between de-blurring and de-noising. In the following image, the blurred image is corrupted by AWGN with variance 10. n=0.2.

The MSE for the restored image is 1964.5. We can see that the sharpness of the edges improved but we also have a lot of noise specs in the image. We can get rid of more specs (thereby getting a smoother image) by increasing *n*. In general, the more noise we have in the image, the higher we set *n*. The higher the *n*, the less high pass the filter is, which means that it amplifies noise less. It also means, however, that the edges will not be as sharp as they could be.

Method 2: Iterative Procedure

The idea behind the iterative procedure is to make some initial guess of *f* based on *g* and to update that guess after every iteration. The procedure is

$$\hat{f}_0(n_1, n_2) = \lambda g(n_1, n_2)$$
$$\hat{f}_{k+1}(n_1, n_2) = \hat{f}_k(n_1, n_2) + \lambda(g(n_1, n_2) - \hat{f}_k(n_1, n_2) * *b(n_1, n_2))$$

where $\hat{f}_0$ is an initial guess based on *g*. If our $\hat{f}_k$ is a good guess, eventually $\hat{f}_k$ convolved with *b* will be close to *g*. When that happens the second term in the $\hat{f}_{k+1}$ equation will disappear and $\hat{f}_k$ and $\hat{f}_{k+1}$ will converge. λ is our convergence factor and it lets us determine how fast $\hat{f}_k$ and $\hat{f}_{k+1}$ converge.

If we take both of the above equations to the frequency domain, we get

$$\hat{F}_0 = \lambda G(\omega_1, \omega_2)$$
$$\hat{F}_{k+1}(\omega_1, \omega_2) = \hat{F}_k(\omega_1, \omega_2) + \lambda(G(\omega_1, \omega_2) - \hat{F}_k(\omega_1, \omega_2)B(\omega_1, \omega_2))$$

Solving for $\hat{F}_k$ recursively, we get

$(1-\lambda B(\omega_1,\omega_2))^{k+1}$ goes to zero as k goes to So

$$\begin{aligned}\hat{F}_k(\omega_1,\omega_2) &= \lambda G(\omega_1,\omega_2)[1+(1-\lambda B(\omega_1,\omega_2))+\ldots+(1-\lambda B(\omega_1,\omega_2))^k] \text{ if}\\ &= \frac{G(\omega_1,\omega_2)}{B(\omega_1,\omega_2)}[1-(1-\lambda B(\omega_1,\omega_2))^{k+1}]\end{aligned}$$

infi
nity, we would get the result as obtained by the inverse filter. In general, this method will not give the exact same results as inverse filtering, but can be less sensitive to noise in some cases.

Implementation and Results

The first thing we have to do is pick a λ λ must satisfy the following

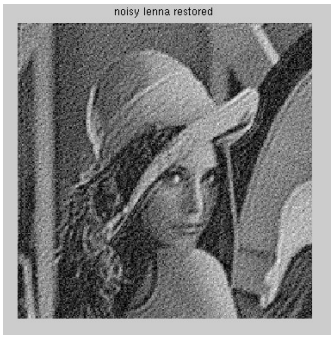
$$|1-\lambda B(\omega_1,\omega_2)| < 1$$

and thus will be a positive integer in the range of 0 to 1. The bigger λ is, the faster and will converge. However, picking too large a λ may also make and diverge instead of converge. Imagine that we're walking along a path and the end of the path is a cliff. λ is the size of of the steps we take. We want to go to the edge of the path as fast as possible without falling off. Taking large steps will ensure that we will get there fast but we'd probably first. Taking small will ensure that we get there without falling off but it could take an infinite amount of time. So the compromise would be to take big steps at the start and decrease our step size as we get close to our destination.

The following is the noiseless image after 150 iterations. λ starts off at 0.1 and decreases by 10% every 25 iterations.

The MSE is 364.6897. The image is sharper than the blurred image although the MSE is high. But the image restored using the direct inverse filter is much better.

The following is the blurred image corrupted with AWGN with a variance of 10. The number of iterations is 150.



The MSE for the restored image is 1247.3. We see the same noise specs as we had seen with the inverse filter. But the image is in general better than the the noisy image restored using the inverse filtering method and has a lower MSE. So we can conclude that the direct inverse filtering method is better for a noiseless case and the iterative method is better when noise is present.

Segmentation of Digital Images:

Image analysis is an area used for extracting the information from an image. Before we extract the information, the image has to be subdivided into constituent parts or objects. The process of subdividing the given image into its constituent parts or objects is called *image segmentation.* Image segmentation is the first step in image analysis. The level at which the subdivision is carried out depends on the problem being solved. For example,

let us consider the image of a basket containing fruits like apples, oranges, and grapes. To know the size of the orange, the image is subdivided into its constituent parts until we get a sub image of the orange. Further subdivision is then not required.
One of the most difficult task in image processing is segmentation process. It plays a vital role in any application and its success is based on the effective implementation of the segmentation technique.

The segmentation algorithms can be divided into two broad categories based on the two important properties, namely, ● Discontinuity and ● Similarity.
The various segmentation techniques based on (1) gray level discontinuity and (2) gray level similarity are well depicted in a graph as shown in Figure 1.

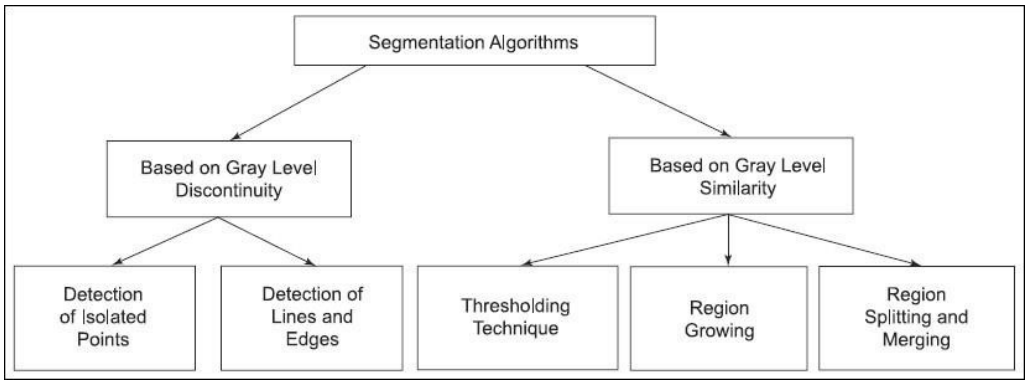


FIGURE 1 Segmentation techniques

DETECTION OF ISOLATED POINTS

In order to detect the isolated points due to noise or any other interference, the general mask employed is shown in Figure ~~2(a) and~~ the typical values of the weights 'W' are shown in Figure 2(b). This mask co~~nsists~~ of coefficients '−1' everywhere except at the center which is 8. The sum of these coefficients is 0. When we place the mask over an image it covers 9 pixels in the image. The average response to the mask is computed as

$$R = \frac{1}{9}\{W_1Z_1 + W_2Z_2 + \cdots + W_9Z_9\} = \frac{1}{9}\sum_{i=1}^{9} W_iZ_i \quad \text{---1}$$

where Wi is the coefficient in the mask and Zi denotes the gray level values of the pixel in the image under the mask. Now the mask is placed at the top left corner of the image and the response to the mask is computed using equation (6.1).

W_1	W_2	W_3
W_4	W_5	W_6
W_7	W_8	W_9

(a)

−1	−1	−1
−1	8	−1
−1	−1	−1

(b)

FIGURE 2 (a) The general representation of the mask (b) The mask with coefficient values

If the mask is over a uniform intensity area, the response due to this mask is equal to 0. This means there are no isolated pixels with different gray level values. On the other hand, if the mask is placed over the area having an isolated point with different gray levels, the response to the mask will be a nonzero value. The aver- age response will be maximum when the isolated points are just below the center of the mask. Therefore, from the mask response it is possible to locate the isolated points resulting due to noise.

LINE DETECTION

The various masks used for detecting horizontal, vertical, +45°, and −45° slanting line are shown in Figure 3.

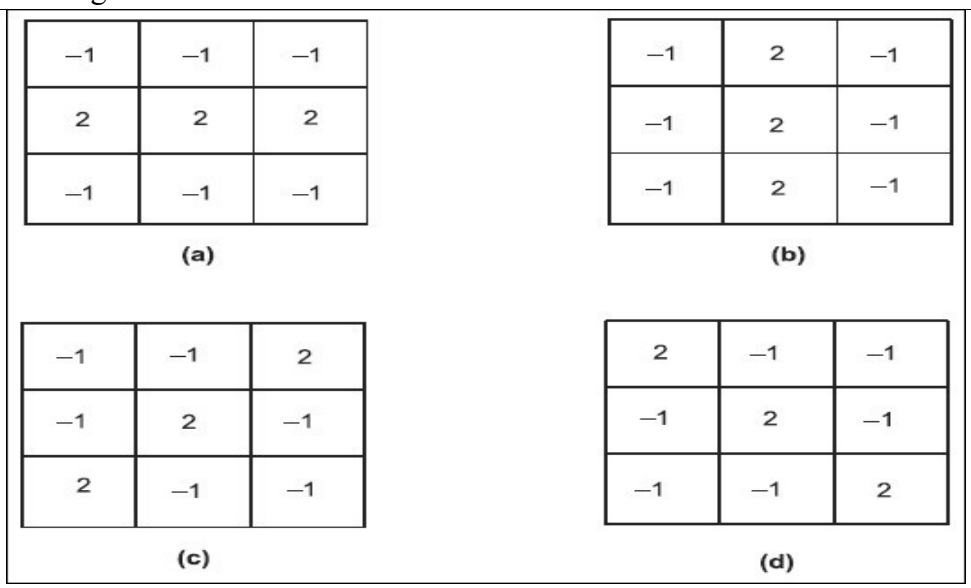


FIGURE 3 Masks for detecting lines. (a) Mask for horizontal line detection (b) Mask for + 45° slanting line detection (c) Mask for vertical line detection (d) Mask for − 45° slanting line detection

If the first mask sh~~own in F~~igure 3(a) is moved around an image, its response will be a large value to lines oriented horizontally. The response will be maximum when the line passes through the mid- dle row of the mask with constant background. For example, when we move the mask over an image consisting of all '1's as back- ground and with a line of different gray level '10's, then the re- sponse due to the first mask is computed as pixel gray level values. Similar experiments with second mask re- sults in high response

$$(1 * -1) + (1 * -1) + (1 * -1) + (2 * 10) + (2 * 10) + (2 * 10)$$
$$+ (+1 * -1) + (+1 * -1) + (+1 * -1) = 54$$

to vertical lines and the third mask to the lines +45° and the fourth mask to the lines in the −45° direction.

Suppose all the masks are applied to an image and the responses computed are denoted as R1, R2, R3, and R4. If at a certain point in the image $|R_i| > |R_j|$ for all $j \neq i$, then the point is more likely to be associated with the line in the direction of mask i. For example, if a

point in the image where | R1| > | Rj | for j = 2, 3, and 4 then that particular point is more likely to be associated with a horizontal line.

EDGE DETECTION

In image processing, the points and line detection are seldom used. In most of the practical image applications edge detection plays a vital role and the concept involved in the edge detection is illustrated in this section with the help of the image shown in Figure 4.

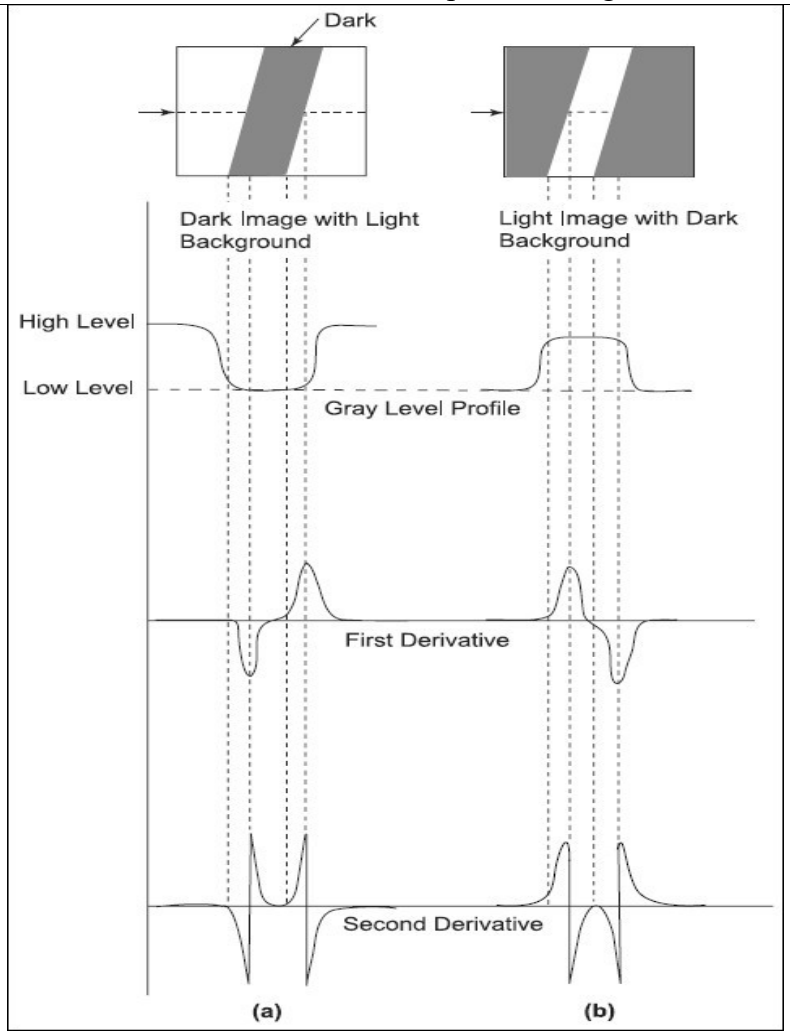


FIGURE 4 Edge detection. (a) The dark object on a light back- ground with its derivatives (b) The bright object on the dark back- ground with its derivatives

An edge is a boundary between two regions with relatively distinct gray level properties. Consider the image shown in the Figure 4(a) consisting of a dark object in a light background. The gray level profile along the horizontal line of the image corresponding to the location shown by the arrow line is also given in the Figure 4(a).

Edge: An edge is a boundary between two regions with relatively distinct gray level properties.

The first derivative of the gray level profile is negative at the leading edge of the transition, positive at the trailing edge, and zero in the areas of constant gray levels. The second derivative is negative for that part of transition associated with the light side of the edge, positive for that part of the transition associated with the dark side of the edge, and zero for pixels lying exactly on edges. By analyzing the first derivative and second derivative of the image profile cor- responding to a horizontal line, the following inference is obtained.
The magnitude of the first derivative is used to detect the presence of an edge in the image and the sign of the second derivative is used to determine whether the edge pixel lies on the dark side or light side of an edge. For example, if the second derivative is positive it shows that the corresponding pixel lies on the dark side of the edge and vice-versa.

The second derivative has a zero crossing at the midpoint of the transition in gray level. The first derivative and the second derivative at any point in an image are obtained by using the magnitude of the gradient at that point and Laplacian operator, respectively. The detailed discussion of the gradient and Laplacian operator

The gradient of an image f(x, y) at the location (x, y) is given by the vector

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \qquad \text{-------- 2}$$

The gradient vector points in the direction of the maximum rate of change of fat (x, y). In the edge detection we employ the magnitude of the gradient vector and it is denoted as

$$\nabla f = \text{mag}(\nabla f) = [G_x^2 + G_y^2]^{\frac{1}{2}}$$

-----3

To reduce the computational complexity the magnitude of the gradient vector can be approximated as given in equation (6.4).

$$\Delta f = |G_x| + |G_y|$$

----4

The direction of the gradient vector is another important quantity and is given in equation (6.5).

$$\alpha(x, y) = \tan^{-1}\left[\frac{G_y}{G_x}\right]$$

-----5

The computation of the gradient of an image is obtained from the partial derivatives $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$ at every pixel in the image. It is always possible to implement the derivatives in digital form in different ways. One of the equivalent digital forms for the gradient is given by Sobel operators and they are given by the following equations.

$$G_x = (P_7 + 2P_8 + P_9) - (P_1 + 2P_2 + P_3)$$

----------6

$$G_y = (P_3 + 2P_6 + P_9) - (P_1 + 2P_4 + P_7) \quad \text{-------7}$$

where P1 to P9 are pixel values in a subimage as shown in Figure 5(a).

The equations (6) and (7) can be represented by two 3 × 3 masks as given in Figure 5(b) and (c).

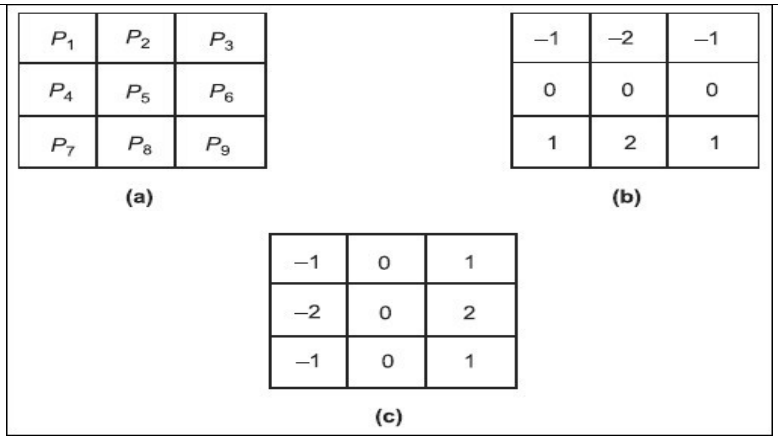


FIGURE 5 Sobel masks. (a) Sub image (b) Sobel mask for horizontal direction (c) Sobel mask for vertical direction

The mask in Figure 5(b) is used to compute Gx at the center point of the 3 × 3 region and mask in Figure 5(c) is used ~~to comp~~ute Gy. The other mask called Prewitts can also be used to compute the gradient Gx and Gy as shown in Figure 6.

The following two equations give the computations of Gx and Gy components.

$$G_x = (P_7 + P_8 + P_9) - (P_1 + P_2 + P_3) \quad \text{------8}$$

$$G_y = (P_3 + P_6 + P_9) - (P_1 + P_4 + P_7) \quad \text{------9}$$

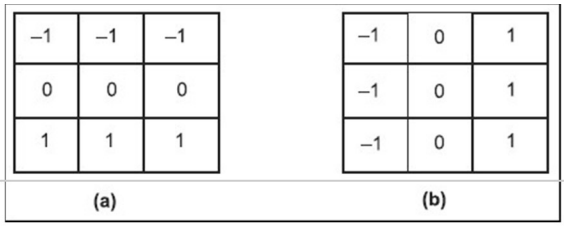


FIGURE 6 Prewitt's masks for horizontal and vertical components. (a) Mask to compute Gx (b) Mask to compute Gy

The simplest possible way to implement the partial derivative at the center of the 3 × 3 mask is to use the Roberts, cross gradient operators.

$$G_x = P_9 - P_5 \quad \text{-----10}$$

$$G_y = P_8 - P_6 \quad \text{------11}$$

The gradient image computed using Sobel operators is given in Figure 7.

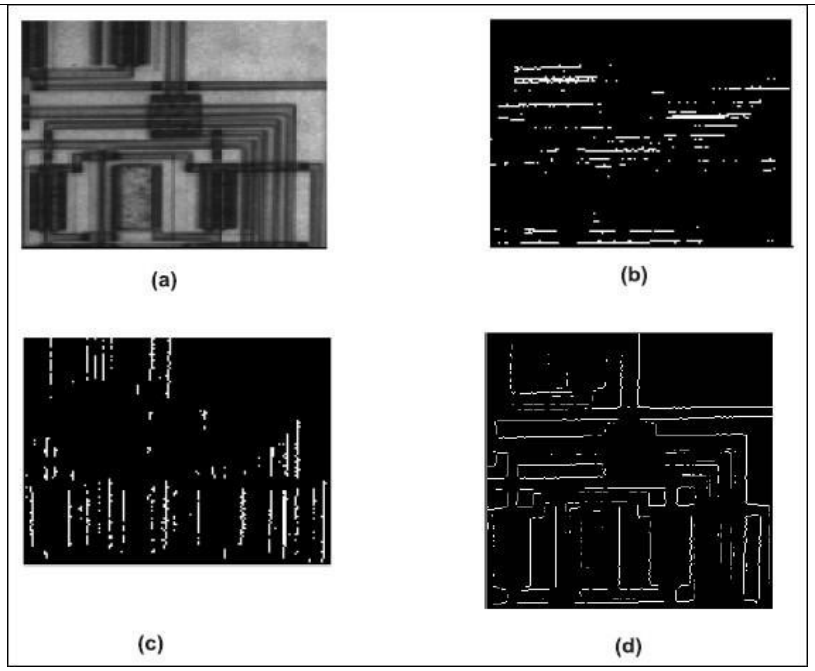


FIGURE 7 The gradient images using Sobel operators. (a) Original image (b) Image obtained using gradient Gx (c) Image obtained using Gy (d) Complete gradient image (Gx + Gy)

Figure 7(a) shows the original image and Figure 7(b) shows the result of computing the modulus of Gx. This result gives the horizontal edges, which is perpendicular to the x-axis. Figure 7(c) gives the computation of gradient |Gy |, for vertical edges, which is perpendicular to the y-direction. Combining the above two components results in Figure 7(d), which is nothing but the gradient

The Laplacian of the two-dimensional image f(x, y) is the second- order derivative and is defined as

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \quad \text{-------12}$$

For a 3 × 3 subimage the digital form equivalent to the Laplacian operator is given as

$$\nabla^2 f = 4P_5 - (P_2 + P_4 + P_6 + P_8) \quad \text{------13}$$

From equation (13) it is possible to define the digital Laplacian mask so that the coefficient associated with the center pixels should be positive and that associated with the outer pixels should be negative. Moreover, the sum of the coefficients should be zero. Such a spatial mask [corresponding to equation (13)] is shown in Figure 8.

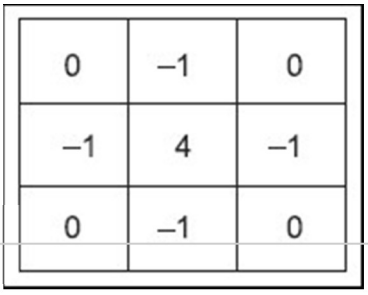


FIGURE 8 The mask used to compute Laplacian operator

Basic Rules for Segmentation

Let R be the entire image region. Then by using the segmentation algorithm the image region R is subdivided into 'n' subregions $R_1, R_2,\ldots, R_n$ such that

$$\bigcup_{i=1}^{n} R_i = R$$

1. This expression indicates that the segmentation process is complete in all respects.
2. R_i is a connected region for $i = 1, 2,\ldots, n$. This condition indicates that the points in the R_i must be connected.
3. $\square$ R_i $R_j = \Phi$ for all i and j, where $i \neq j$. This condition indicates that region must be disjoint.
4. $P(R_i)$ = True for $i = 1,2,\ldots, n$. This means all the pixels in the region Ri have the same intensity.
5. $P(R_i \cup R_i)$ = False for $i \neq j$. This condition indicates that the regions R_i and R_j are different in the sense of the predicate P.

EDGE LINKING AND BOUNDARY DETECTION

In practice, the set of pixels detected by the gradient operators do not form a complete boundary due to noise, non-uniform illumination, and other effects. Thus a linking and other boundary detection procedures to assemble the edge pixels into meaningful boundary follow the edge detection algorithm. There are a number of techniques available for this purpose and they are

1. Local processing
2. Global processing using Hough transform
3. Graph theoretic approach
4. Thresholding
5. Region growing and

6. Region splitting and merging.

SEGMENTATION USING THRESHOLD

Thresholding is one of the most important techniques used for image segmentation. In this section we discuss the various thresholding techniques for image segmentation. The merits and demerits of various techniques are also discussed.

Fundamental Concepts: The histogram of an image f(x, y) consisting of a light object on the dark background is shown in Figure 9(a). This histogram con-sists of two dominant regions, one for the object and the other for the background. For such an image it is easy to select a threshold T that separates the object and background region. Then for any point (x, y), f(x, y) > T is called an object point, otherwise the point is called a background point. A general case of this approach is shown in Figure 9(b).

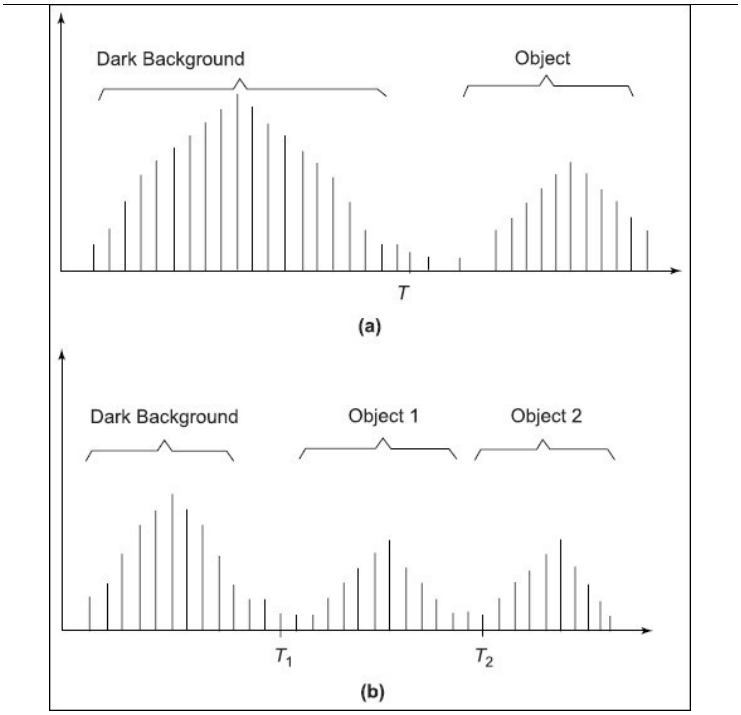


FIGURE 9 (a) Histogram of an image consisting of dark back- ground and a light object (b) Histogram for two objects in a dark background

Figure 9(b) has three dominant regions that characterize the histogram of the given image. This histogram corresponds to two different light objects on a dark background. From the histogram it is possible to select two different threshold values T1 and T2, respectively. Then a point (x, y) belongs to the first object if T 1 < f(x, y)≤ T2, or it belongs to the second object if f(x, y) > T2 and to the background if f(x, y) ≤ T1. Usually this kind of thresholding is called multilevel thresholding and is less reliable. single

The reason for this is, that it is difficult to locate multiple thresholds in a given histogram for a real image. The thresholding technique can be put into three different types based on the function T and its associated parameters as given in equation (14).

$$T = T[(x, y), p(x, y), f(x, y)] \qquad \text{-------14}$$

where f(x, y) is the gray level at the point (x, y) and p(x, y) denotes some local property at that point (e.g. the average gray level of neighborhood center on (x, y)). The threshold image g(x, y) is giv- en in equation (15).

$$G(x, y) = \begin{cases} 1 & \text{if } f(x, y) > T \\ 0 & \text{if } f(x, y) \leq T \end{cases} \qquad \text{-------15}$$

In the thresholded image the pixel labeled 1 corresponds to the object, whereas pixels labeled 0 corresponds to the background.

When the threshold value T depends only on f(x, y) the threshold technique is called global. If T depends on both f(x, y) and p(x, y), the threshold is called local. If T depends on all the three parameters, that is, the coordinates (x, y), local property p(x, y), and f(x,y)then the threshold is called dynamic.

Region Growing by Pixel Aggregation

Region growing is a technique that groups the pixels or subregions into larger regions. The simplest of these approaches is pixel aggregation. In this approach a set of seed points are used as starting points and from this, regions grow by appending to each seed point those neighboring pixels that have similar properties.

Pixel Aggregation: A procedure used to group pixels with similar gray level properties in an image or a region in the image.

Let the pixel 2 and 7 with the co-ordinate (4, 2) and (3, 4), respectively be the two seeds. Using the two seeds as starting point, the regions are grown by applying the predicate property P. The predicate P to be used to include a pixel in either region is that the absolute difference between the gray level of that pixel and the gray level of the seed be less than a threshold value T.

Any pixel that satisfies this property simultaneously for both seeds is arbitrarily assigned to region R1.

Figure 10(b) shows the result obtained using the threshold value T = 3. The segmentation results show two regions namely, R1 and R2 and they are denoted by a′s and b′s. It is also noted that the two regions formed are independent of the starting point. However, if we choose T = 8, the resulting region is shown in Figure 10(c).

The pixel aggregation technique used to grow the region is simple and suffers from two difficulties. The first one is the selection of initial seeds that properly represent the region of interest and the second is the selection of suitable properties for including points in the various regions during the region growing processes. The number of seed points selected can be based on the nature of the problem. Figure 11(a) shows an image with a single seed point. The threshold used for the region growing is the absolute difference in the gray level between the

seed and a candidate point, not exceeding 10% of the difference between maximum and minimum gray level in the entire image.

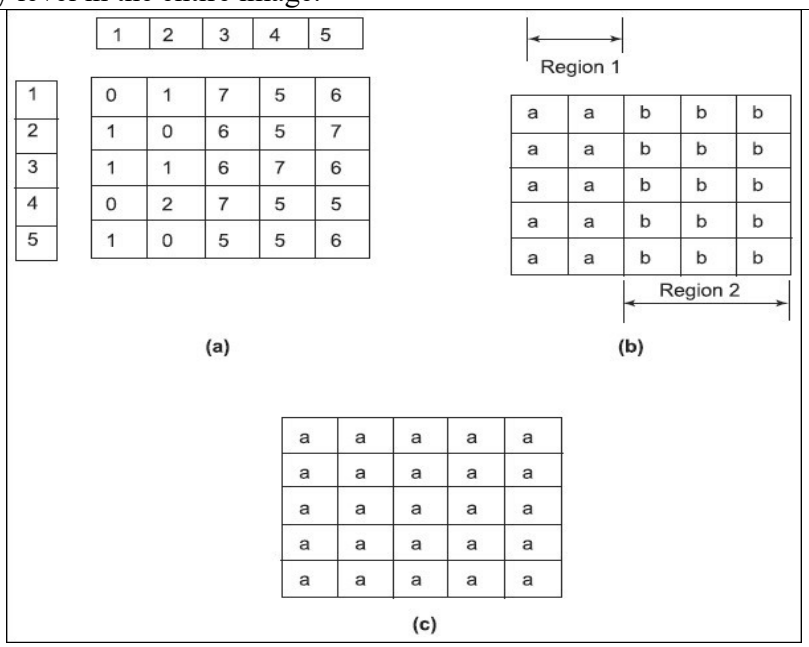


FIGURE 10 (a) A subimage with co-ordinates and gray values (b) Region growing with T = 3 (c) Region growing with T = 8

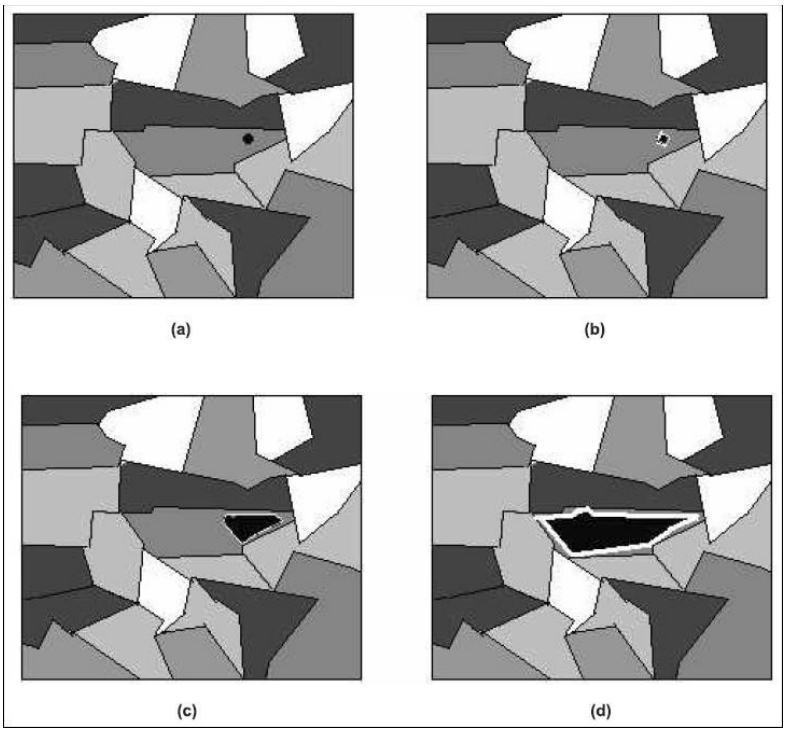


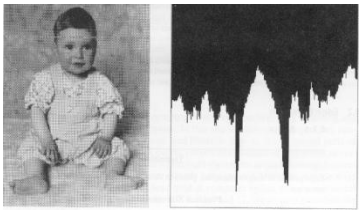
FIGURE 11 (a) An image with seed point (given as dark point), (b)Region growing after few iterations, (c) Intermediate stage of region growing (d) Final growth

The property used to add a pixel into the region is an 8-connected one. Figure 11(b) shows the region in the early stages of region growing. Figure 11(c) shows the region in an intermediate stage of the region growing and Figure 11(d) shows the complete region grown by using this technique.

Module 5: Edge detection

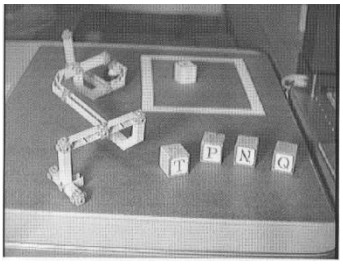
Edges are significant local changes of intensity in an image.

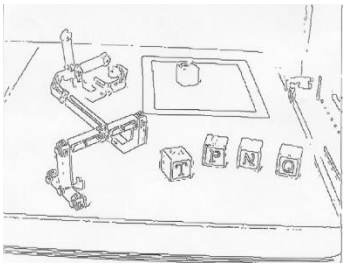
Edges typically occur on the boundary between two different regions in an image.



- **Goal of edge detection**

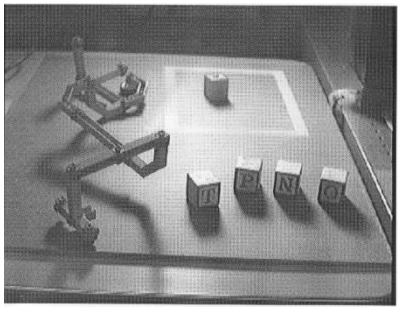
- Produce a line drawing of a scene from an image of that scene.
- Important features can be extracted from the edges of an image (e.g., corners, lines, curves). - These features are used by higher-level computer vision algorithms (e.g., recogni- tion)

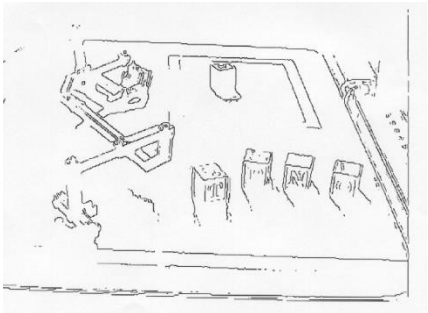




What causes intensity changes?

- Various physical events cause intensity changes.
- Geometric events
  - * object boundary (discontinuity in depth and/or surface color and texture)
  - * surface boundary (discontinuity in surface orientation and/or surface color and texture)
- Non-geometric events
  - * specularity (direct reflection of light, such as a mirror)
  - * shadows (from other objects or from the same object) * inter-reflections



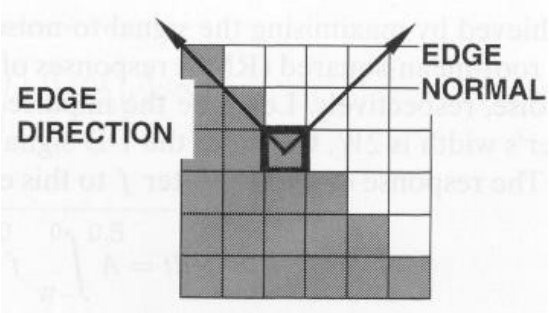


- **Edge descriptors**

Edge normal: unit vector in the direction of maximum intensity change.

Edge direction: unit vector to perpendicular to the edge normal.

Edge position or center: the image position at which the edge is located.

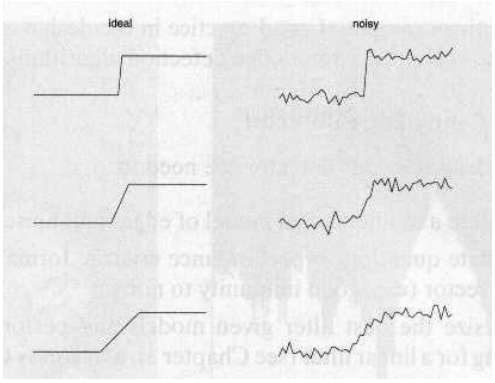


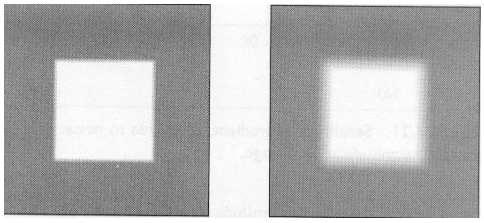
Edge strength: related to the local image contrast along the normal.

- **Modeling intensity changes**

- Edges can be modeled according to their intensity profiles.

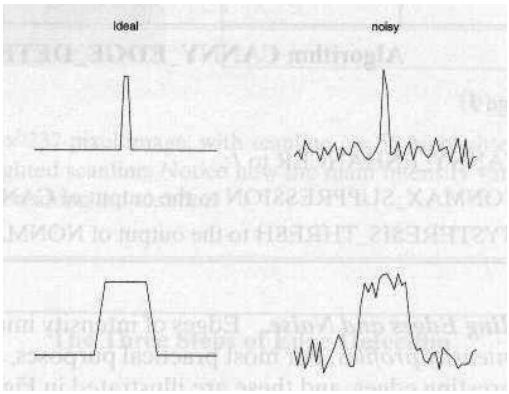
Step edge: the image intensity abruptly changes from one value to one side of the discontinuity to a different value on the opposite side.



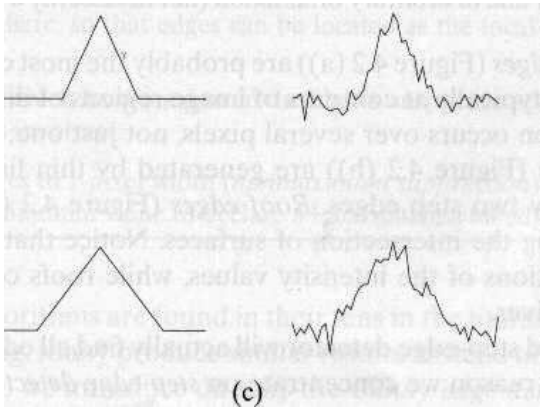


Ramp edge: a step edge where the intensity change is not instantaneous but occur over a finite distance.

Ridge edge: the image intensity abruptly changes value but then returns to the starting value within some short distance (generated usually by lines).



Roof edge: a ridge edge where the intensity change is not instantaneous but occur over a finite distance (generated usually by the intersection of surfaces).



- **The four steps of edge detection**

(1) Smoothing: suppress as much noise as possible, without destroying the true edges.

(2) Enhancement: apply a filter to enhance the quality of the edges in the image (sharpening).

(3) Detection: determine which edge pixels should be discarded as noise and which should be retained (usually, thresholding provides the criterion used for detection).

(4) Localization: determine the exact location of an edge (*sub-pixel* resolution might be required for some applications, that is, estimate the location of an edge to better than the spacing between pixels). Edge thinning and linking are usually required in this step.

- **Edge detection using derivatives**

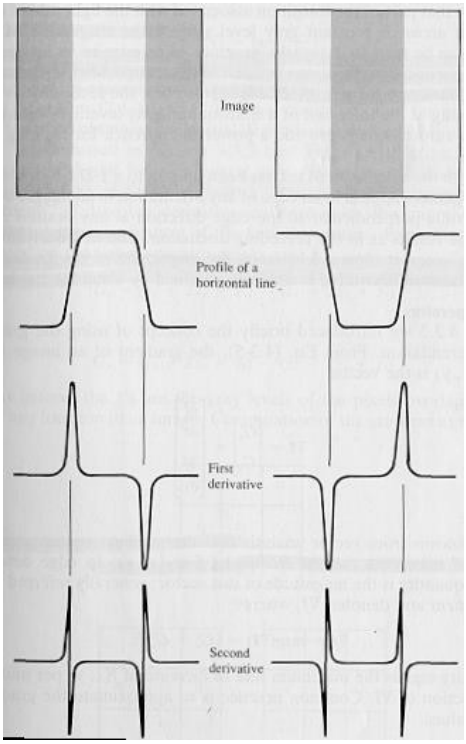
- Calculus describes changes of continuous functions using *derivatives*.

- An image is a 2D function, so operators describing edges are expressed using *partial derivatives*.

- Points which lie on an edge can be detected by:

(1) detecting local maxima or minima of the first derivative

(2) detecting the zero-crossing of the second derivative



- **Differencing 1D signals** (see also Trucco, Appendix A.2)

- To compute the derivative of a signal, we approximate the derivative by finite dif- ferences:

mask M = [−1, 0, 1]

S_1			12	12	12	12	12	24	24	24	24	24
S_1	⊗	M	0	0	0	0	12	12	0	0	0	0

(a) S_1 is an upward step edge

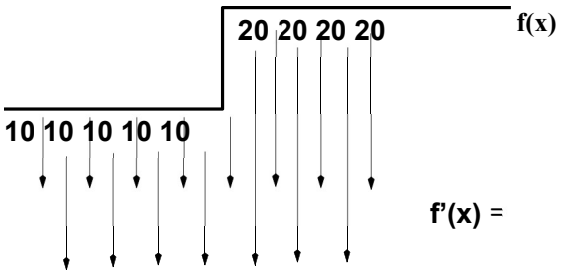
S_2			24	24	24	24	24	12	12	12	12	12
S_2	⊗	M	0	0	0	0	-12	-12	0	0	0	0

(b) S_2 is a downward step edge

S_3			12	12	12	12	15	18	21	24	24	24
S_3	⊗	M	0	0	0	3	6	6	6	3	0	0

(c) S_3 is an upward ramp

S_4			12	12	12	12	24	12	12	12	12	12
S_4	⊗	M	0	0	0	12	0	-12	0	0	0	0



f(x+1) - f(x)

(approximates f'() at x+1/2)

0 0 0 10 -10 0 0 **f''(x)** = f(x-1)-2f(x)+f(x+1) (approximates f''() at x)

zero-crossing

- Examples using the edge models:

mask M = [−1, 2, −1]

S_1			12	12	12	12	12	24	24	24	24	24
S_1	⊗	M	0	0	0	0	-12	12	0	0	0	0

(a) S_1 is an upward step edge

S_2			24	24	24	24	24	12	12	12	12	12
S_2	⊗	M	0	0	0	0	12	-12	0	0	0	0

(b) S_2 is a downward step edge

S_3			12	12	12	12	15	18	21	24	24	24
S_3	⊗	M	0	0	0	-3	0	0	0	3	0	0

(c) S_3 is an upward ramp

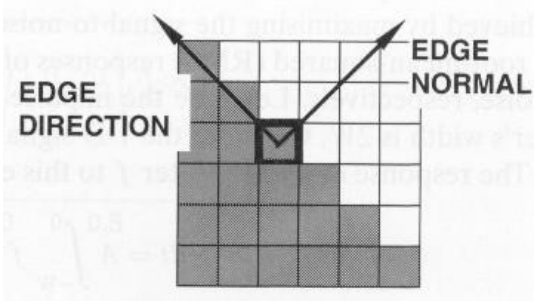
S_4			12	12	12	12	24	12	12	12	12	12
S_4	⊗	M	0	0	0	-12	24	-12	0	0	0	0

Edge detection using the gradient

- **Properties of the gradient**

- The magnitude of gradient provides information about the strength of the edge.

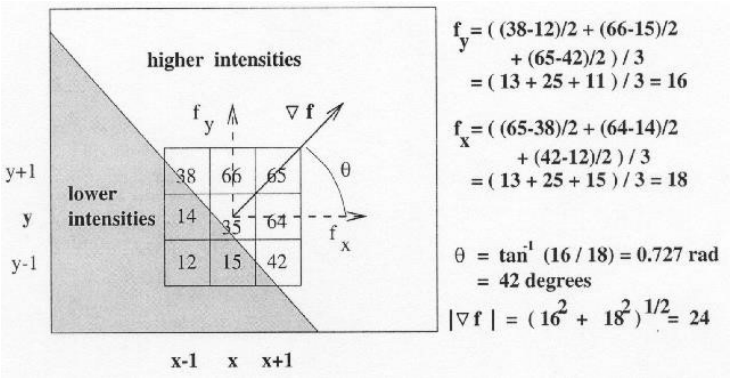
- The direction of gradient is always perpendicular to the direction of the edge (the edge direction is rotated with respect to the gradient direction by -90 degrees).



- **The Roberts edge detector**

- **The Sobel edge detector**

- **Main steps in edge detection using masks**



(1) Smooth the input image ($\hat{f}(x, y) = f(x, y) * G(x, y)$)

(2) $\hat{f}_x = \hat{f}(x, y) * M_x(x, y)$

(3) $\hat{f}_y = \hat{f}(x, y) * M_y(x, y)$

(4) $magn(x, y) = |\hat{f}_x| + |\hat{f}_y|$

(5) $dir(x, y) = \tan^{-1}(\hat{f}_y / \hat{f}_x)$

(6) If $magn(x, y) > T$, then possible edge point

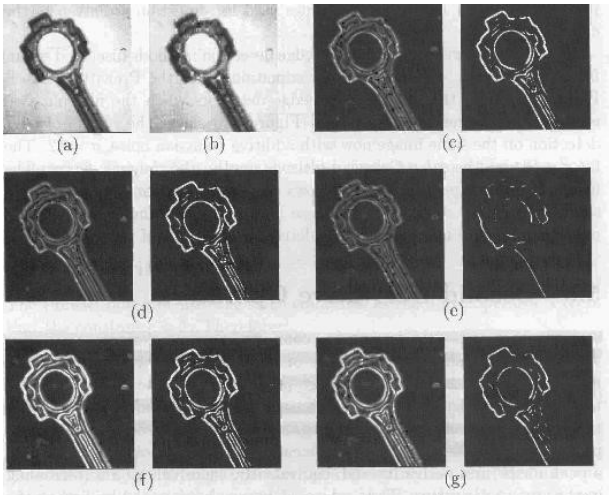


Figure 5.4: A comparison of various edge detectors. (a) Original image. (b) Filtered image. (c) Simple gradient using 1×2 and 2×1 masks, $T = 32$. (d) Gradient using 2×2 masks, $T = 64$. (e) Roberts cross operator, $T = 64$. (f) Sobel operator, $T = 225$. (g) Prewitt operator, $T = 225$.

(with noise filtering)

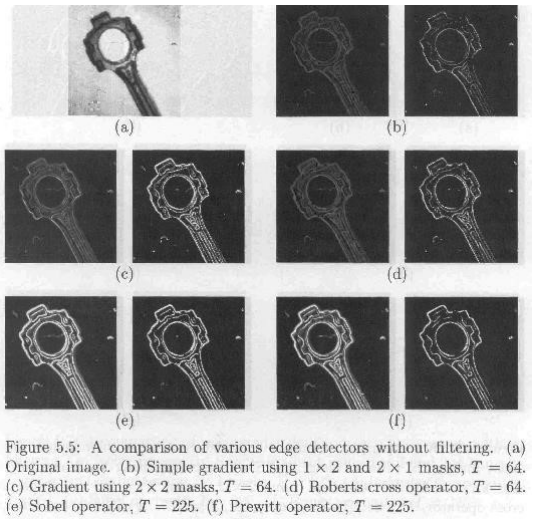
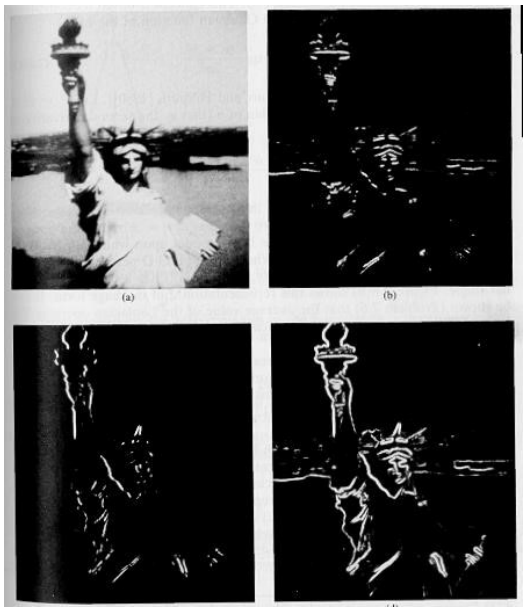


Figure 5.5: A comparison of various edge detectors without filtering. (a) Original image. (b) Simple gradient using 1×2 and 2×1 masks, $T = 64$. (c) Gradient using 2×2 masks, $T = 64$. (d) Roberts cross operator, $T = 64$. (e) Sobel operator, $T = 225$. (f) Prewitt operator, $T = 225$.

(without noise filtering)

- **Isotropic property of gradient magnitude**



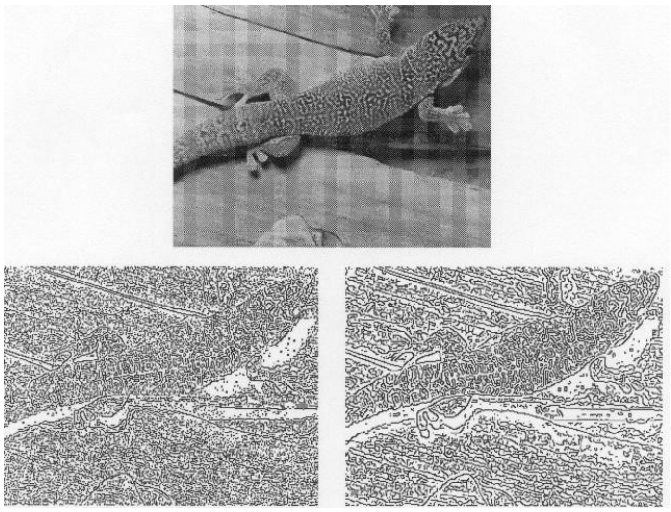
- The magnitude of gradient is an *isotropical* operator (it detects edges in any direction !!)

- **Some practical issues**

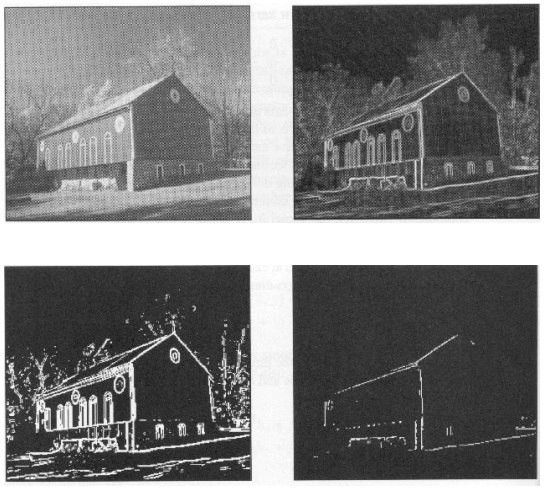
- The differential masks act as high-pass filters which tend to amplify noise.

- To reduce the effects of noise, the image needs to be smoothed first with a low- pass filter.

(1) The noise suppression-localization tradeoff: a lar ger filter reduces noise, but worsens localization (i.e., it adds uncertainty to the location of the edge) and vice versa.

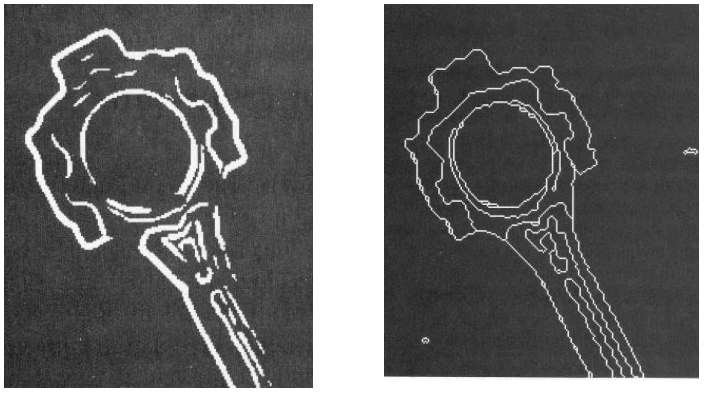


- (2) How should we choose the threshold?



- Edge thinning and linking are required to obtain good contours.

- **Criteria for optimal edge detection**



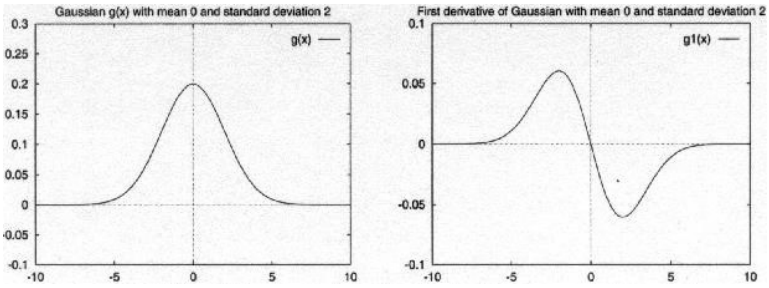
Good detection: the optimal detector must minimize the probability of false positives (detecting spurious edges caused by noise), as well as that of false neg- atives (missing real edges).

The Canny edge detector

- This is probably the most widely used edge detector in computer vision.

- Canny has shown that the first derivative of the Gaussian closely approximates the operator that optimizes the product of *signal-to-noise* ratio and localization.

- His analysis is based on "step-edges" corrupted by "additive Gaussian noise".



Algorithm

1. Compute f_x and f_y

$$f_x = \frac{\partial}{\partial x}(f * G) = f * \frac{\partial}{\partial x}G = f * G_x$$

$$f_y = \frac{\partial}{\partial y}(f * G) = f * \frac{\partial}{\partial y}G = f * G_y$$

$G(x, y)$ is the Gaussian function

$G_x(x, y)$ is the derivate of $G(x, y)$ with respect to x: $G_x(x, y) = \frac{-x}{\sigma^2} G(x, y)$

$G_y(x, y)$ is the derivate of $G(x, y)$ with respect to y: $G_y(x, y) = \frac{-y}{\sigma^2} G(x, y)$

2. Compute the gradient magnitude

$$magn(i, j) = \sqrt{f_x^2 + f_y^2}$$

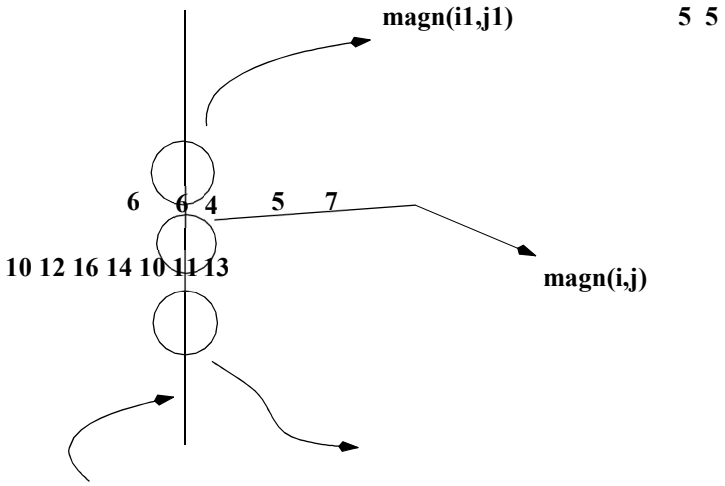
3. Apply non-maxima suppression.

- **Non-maxima suppression**

- To find the edge points, we need to find the local maxima of the gradient magni- tude.

- Broad ridges must be thinned so that only the magnitudes at the points of greatest local change remain.

- All values along the direction of the gradient that are not peak values of a ridge are suppressed.


magn(i1,j1)
5 5
6 6 4 5 7
10 12 16 14 10 11 13
magn(i,j)

Algorithm

For each pixel (x,y) do:

if $magn(i, j) < magn(i_1, j_1)$ or $magn(i, j) < magn(i_2, j_2)$ then
$I_N(i, j) \leftarrow 0$ else $I_N(i, j) \leftarrow magn(i, j)$

- **Hysteresis thresholding/Edge Linking**

- The output of non-maxima suppression still contains the local maxima created by noise.

- Can we get rid of them just by using a single threshold?

  * if we set a low threshold, some noisy maxima will be accepted too.

  * if we set a high threshold, true maxima might be missed (the value of true maxima will fluctuate above and below the threshold, fragmenting the edge).

- A more effective scheme is to use two thresholds:

  * a low threshold t_l

  * a high threshold t_h

  * usually, $t_h \approx 2t_l$

Algorithm

1. Produce two thresholded images $I_1(i,j)$ and $I_2(i,j)$.

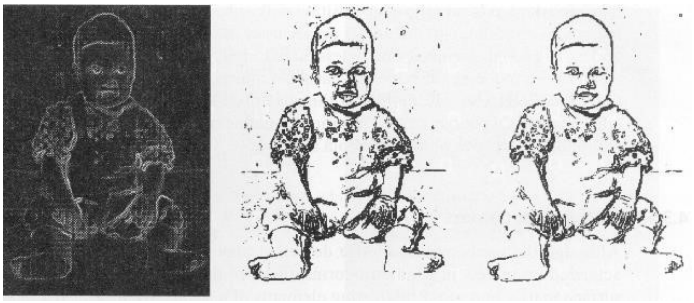
(note: since $I_2(i,j)$ was formed with a high threshold, it will contain fewer false edges but there might be gaps in the contours)

2. Link the edges in $I_2(i,j)$ into contours

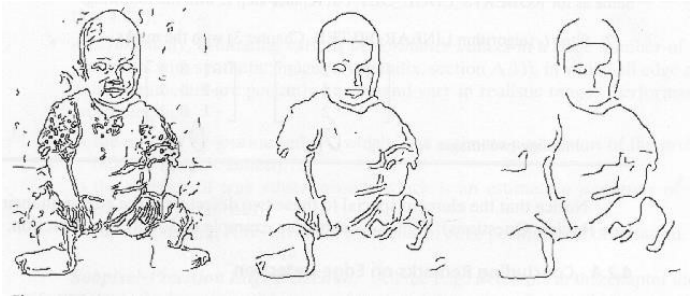
2.1 Look in $I_1(i,j)$ when a gap is found.

2.2 By examining the 8 neighbors in $I_1(i,j)$, gather edge points from $I_1(i,j)$ until the gap has been bridged to an edge in $I_2(i,j)$.

- The algorithm performs edge linking as a by-product of double-thresholding !!

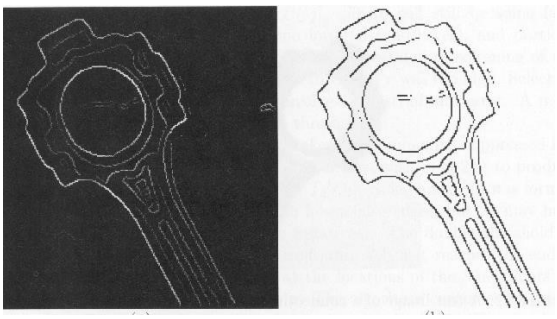


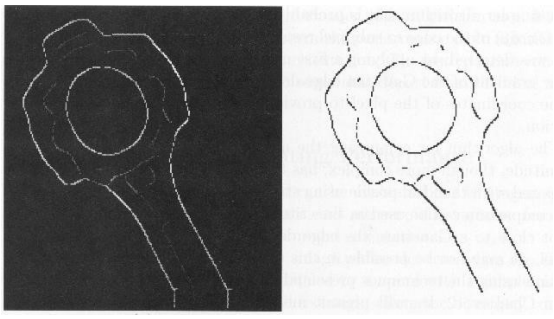
(left:Sobel, middle: thresh=35, right: thersh=50)



(Canny - left: =1, middle: =2, right: =3)

(Canny - 7x7 Gaussian, more details)

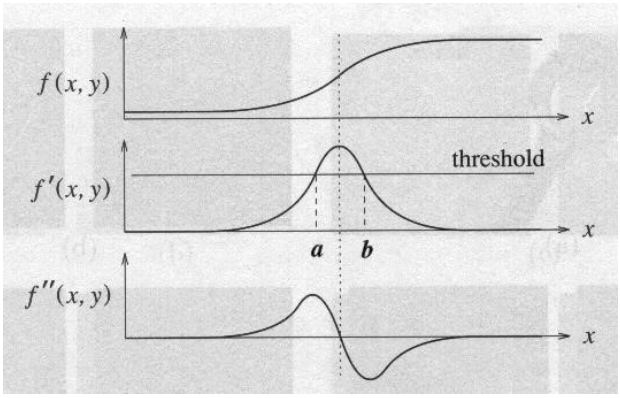




(Canny - 31x31 Gaussian, less details)

Edge detection using the second derivative

- Edge points can be detected by finding the zero-crossings of the second derivative.



- **Properties of the Laplacian**

- It is an isotropic operator.

- It is cheaper to implement (one mask only).

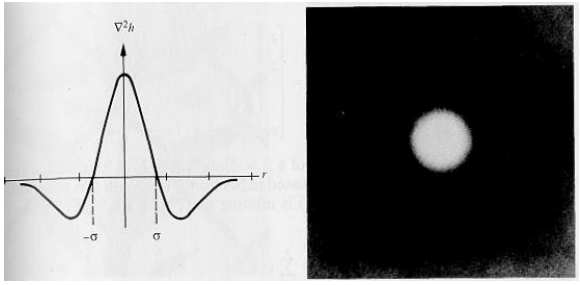
- It does not provide information about edge direction.

- It is more sensitive to noise (differentiates twice).

- **The Laplacian-of-Gaussian (LOG)**

- To reduce the noise effect, the image is first smoothed with a low-pass filter.

- In the case of the LOG, the low-pass filter is chosen to be a Gaussian.



5×5 Laplacian of Gaussian mask

0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

17×17 Laplacian of Gaussian mask

0	0	0	0	0	0	−1	−1	−1	−1	−1	0	0	0	0	0	0
0	0	0	0	−1	−1	−1	−1	−1	−1	−1	−1	−1	0	0	0	0
0	0	−1	−1	−1	−2	−3	−3	−3	−3	−3	−2	−1	−1	−1	0	0
0	0	−1	−1	−2	−3	−3	−3	−3	−3	−3	−3	−2	−1	−1	0	0
0	−1	−1	−2	−3	−3	−3	−2	−3	−2	−3	−3	−3	−2	−1	−1	0
0	−1	−2	−3	−3	−3	0	2	4	2	0	−3	−3	−3	−2	−1	0
−1	−1	−3	−3	−3	0	4	10	12	10	4	0	−3	−3	−3	−1	−1
−1	−1	−3	−3	−2	2	10	18	21	18	10	2	−2	−3	−3	−1	−1
−1	−1	−3	−3	−3	4	12	21	24	21	12	4	−3	−3	−3	−1	−1
−1	−1	−3	−3	−2	2	10	18	21	18	10	2	−2	−3	−3	−1	−1
−1	−1	−3	−3	−3	0	4	10	12	10	4	0	−3	−3	−3	−1	−1
0	−1	−2	−3	−3	−3	0	2	4	2	0	−3	−3	−3	−2	−1	0
0	−1	−1	−2	−3	−3	−3	−2	−3	−2	−3	−3	−3	−2	−1	−1	0
0	0	−1	−1	−2	−3	−3	−3	−3	−3	−3	−3	−2	−1	−1	0	0
0	0	−1	−1	−1	−2	−3	−3	−3	−3	−3	−2	−1	−1	−1	0	0
0	0	0	0	−1	−1	−1	−1	−1	−1	−1	−1	−1	0	0	0	0
0	0	0	0	0	0	−1	−1	−1	−1	−1	0	0	0	0	0	0

- **Gradient vs LOG: a comparison**

- Gradient works well when the image contains sharp intensity transitions and low noise

- Zero-crossings of LOG offer better localization, especially when the edges are not very sharp

The second directional derivative

2	2	2	2	2	8	8	8	8	8
2	2	2	2	2	8	8	8	8	8
2	2	2	2	2	8	8	8	8	8
2	2	2	2	2	8	8	8	8	8
2	2	2	2	2	8	8	8	8	8
2	2	2	2	2	8	8	8	8	8

A sample image containing a vertical step edge.

0	0	0	6	−6	0	0	0
0	0	0	6	−6	0	0	0
0	0	0	6	−6	0	0	0
0	0	0	6	−6	0	0	0

- This is the second derivative computed in the direction of the gradient.

2	2	2	2	2	5	8	8	8	8
2	2	2	2	2	5	8	8	8	8
2	2	2	2	2	5	8	8	8	8
2	2	2	2	2	5	8	8	8	8
2	2	2	2	2	5	8	8	8	8
2	2	2	2	2	5	8	8	8	8

A sample image containing a vertical ramp edge.

0	0	0	3	0	−3	0	0
0	0	0	3	0	−3	0	0
0	0	0	3	0	−3	0	0
0	0	0	3	0	−3	0	0

Multiscale processing (scale space)

- A serious practical problem with any edge detector is the matter of choosing the *scale* of smoothing (e.g.,the value of using a Gaussian). - For many applications, it is desirable to be able to process an image at multiple scales.
- We determine which edges are most significant in terms of the range of scales over which they are observed to occur.

STEGANOGRAPHY

Introduction

Though the fields of steganography and cryptography are associated with one another, there is a distinction to be made. Cryptography is the art of jumbling a message so that a wouldbe

eavesdropper cannot interpret the message. Steganography, on the other hand, is the art of hiding a message so that a would-be eavesdropper is unaware of the message's presence.

While steganography has been around for centuries, the Digital Revolution has sparked a renewed interest in the field. For instance, the mass media industry has shown increasing interest in steganography to fight piracy. It is even rumored that the terrorist organization, AlQaeda, has employed steganography to transmit orders to its operatives over the internet.

All digital file formats can be used to hide secret messages. This paper, however, focuses specifically on the techniques employed in hiding information in digital image files.

A Generic Steganographic System

As with any other science, steganography has its own set of terminology. The term *cover* is used to describe the original message in which we will hide our secret message. Once we embed our secret message into the cover, the new message is known as the *stego data*. The *stego data* is analogous the *cipher text* of cryptography.

A generic steganographic system, or *stegosystem*, works thusly. A secret message is embedded into the cover data using some sort of embedding algorithm. The cover data may be a single file, but that is not necessarily the case. The embedding algorithm then outputs the stego data. There is, however, a minor detail that needs to be added to the system. Recall Kerckhoff's principle, which states that the security of a system should not be based on the obscurity of the algorithm, but on the strength of its key. Therefore the embedding algorithm should require a key as an input. Additionally it is advisable to encrypt the secret message prior to embedding it.

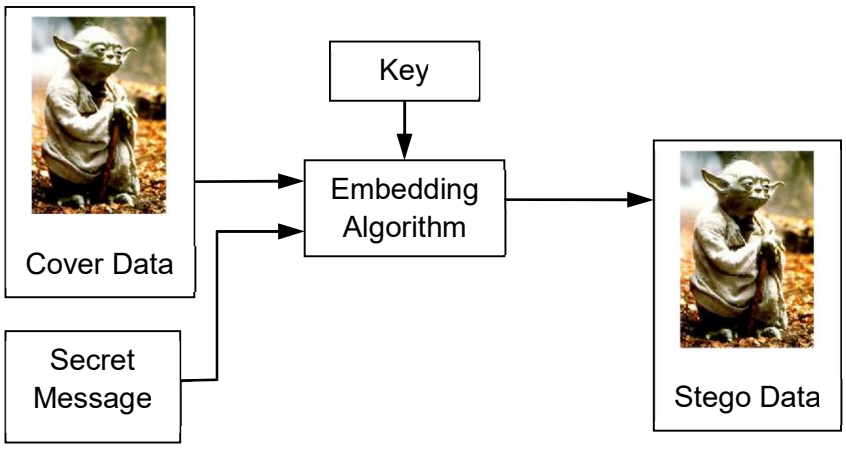


Figure 1 An overview of a generic steganographic system.

Though embedding algorithms may take many forms, there are some requirements that all embedding algorithms should meet. Firstly, the distortion of the cover data as a result of the embedding algorithm should be as imperceptible as possible. Secondly, no part of the secret message should be contained in the header of the stego data file. The message must become part of the cover data and should be immune to manipulation attacks such as re-sampling or filtering. Ideally, it would also be a good idea to include error correcting codes into the message so that if the stego data is damaged, the message can still be recovered. Finally, it is imperative that the original cover data never fall into the hands of an eavesdropper or be used twice. Since the embedding process is additive, the secret message can be recovered if an eavesdropper has different stego files which utilize the same cover data.

We will now explore some of the more popular techniques for embedding messages into cover.

LSB Modification

LSB modification is perhaps the most popular method to embed a message into cover data. As its name suggests, this method works by modifying the least significant bit of one of the RGB values of the pixel data. The secret message data is then scattered pseudo-randomly across the image. This technique is analogous to the spread spectrum communication technique of *frequency hopping*.

This method is quite effective against human detection because it is difficult for the human eye to discern an LSB modified pixel. Also, any modifications that are made could easily be attributed to "noise" that may already be contained in the image. However, computer generated images, such as those generated by vector drawing applications like Adobe Illustrator or

Macromedia Flash, do not contain much noise and would therefore make a poor choice as cover data.

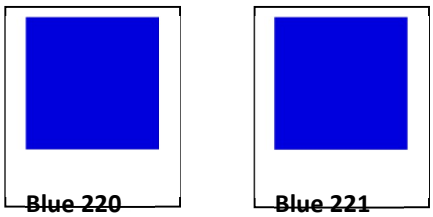


Figure 2 A comparison of an LSB altered color tone.

While 24-bit *true-color* RGB data formats are best suited for LSB modification, it is possible to use this method with 8-bit *color-index* data formats. This can be tricky, however, because the palette is much smaller and pixel luminescence variation may be much greater and more easily detected. Therefore, it is wise to attempt LSB modification with a grayscale or monochromatic cover image.

There are, however, problems with LSB modification. For one, this method will only work with raw image data. Lossy image compression formats, such as JPEG, do not store images in an RGB format and are therefore not as forgiving to simple bit manipulation. Another problem with LSB manipulation is that if the stego data is compressed with a lossy compression algorithm, the secret message may be destroyed.

JPEG Algorithms

While JPEG files are not as tolerant to bit manipulation as uncompressed image files, it is still possible to use them as cover data. The key is to know where to hide the information.

The JPEG algorithm works by dividing an image into several 8 x 8 pixel matrices. Discrete cosine transform (DCT) coefficients are then calculated for each matrix. The coefficients are then multiplied by a quantization matrix. The results are then rounded off to the nearest integer. The rounded numbers are then further compressed and the results are saved.

It is in these DCT values where we can hide our data. A typical approach involves slightly altering a set of the largest DCT coefficients. These larger values contain the most "energy" and would therefore produce the least amount of distortion in the image. Another approach is to choose DCT coefficients that fall into a particular range so as to avoid perception.

The popular JPEG steganography algorithms, F5 and JSteg, both use DCT modification to embed data. And while both algorithms generally escape human detection, they are both detectable through statistical analysis.

Patchwork

Patchwork, an early steganographic algorithm, works by hiding a single bit in pairs of pixels that are in different parts of the image. Here is a sketch of the Patchwork algorithm:

- Create a pseudo-random bit stream to choose pairs of pixels from the cover data.
- For each pair, let d be the difference between the two pixels.
- Encode a bit of information into the pair. Let $d < 0$ represent 0 and $d > 0$ represent 1. If the pixels are in the wrong order, switch them.
- If d happens to be greater than a predefined threshold or if is equal to 0, disregard the pair and move on to a new pair.

Because the algorithm involves swapping pixels, a fair amount of image distortion may occur. Therefore, this algorithm is only suited for embedding a small amount of data. A benefit of this algorithm can resilient against lossy compression algorithms if the message is redundantly embedded.

Steganalysis

While steganography is the art of hiding data, steganalysis is the art of finding a steganographic message. Attacks on a steganographic system can be summarized as follows:

- **Traffic analysis –** If an attacker suspects Alice and Bob are sending messages to each other, a simple attack would be to monitor the information that they send to each other.

- **Detection** – Stego data can be detected through visual or statistical attacks. Visual attacks work if the embedding algorithm causes noticeable artifacts in the stego data. Statistical attacks work by comparing the frequencies of a potential stego file with the theoretically expected frequencies of the file. While statistical attacks are quite effective in identifying stego, they cannot necessarily recover the secret message.

- **Brute force** - An attacker has received some stego data and is attempting to recover the message. Unless the sender of the stego data used the same cover data twice, the attacker has a copy of the cover data, the attacker has the key, or a poor key was used, this attack can be difficult.

- **Manipulation –** By altering the stego data, an attacker may be able to destroy the message. There are a number ways that this can be done: lossy compression, cropping (as with the Mosaic attack), rotating, or scaling (as with Anderson's Stirmark). This type of attack typically beats digital watermarking.

Conclusion

Although steganography is an interesting research field, it falls short of the mark. Because the secret message can be damaged with manipulation attacks, current steganographic algorithms are fairly useless for digital watermarking. Additionally, all known image based stegosystems can be detected with statistical analysis. Perhaps one day an undetectable or unalterable stegosystem will be developed. Or maybe the mythical "public-key" stegosystem will be devised.

Digital Watermarking

Introduction

Digital watermarking is the act of hiding a message related to a digital signal (i.e. an image, song, video) within the signal itself. It is a concept closely related to steganography, in that they both hide a message inside a digital signal. However, what separates them is their goal. Watermarking tries to hide a message related to the actual content of the digital signal, while in steganography the digital signal has no relation to the message, and it is merely used as a

cover to hide its existence. Watermarking has been around for several centuries, in the form of watermarks found initially in plain paper and subsequently in paper bills. However, the field of digital watermarking was only developed during the last 15 years and it is now being used for many different applications.

Watermarking applications

The increasing amount of research on watermarking over the past decade has been largely driven by its important applications in digital copyrights management and protection. One of the first applications for watermarking was broadcast monitoring. It is often crucially important that we are able to track when a specific video is being broadcast by a TV station. This is important to advertising agencies that want to ensure that their commercials are getting the air time they paid for. Watermarking can be used for this purpose. Information used to identify individual videos could be embedded in the videos themselves using watermarking, making broadcast monitoring easier. Another very important application is owner identification. Being able to identify the owner of a specific digital work of art, such as a video or image can be quite difficult. Nevertheless, it is a very important task, especially in cases related to copyright infringement. So, instead of including copyright notices with every image or song, we could use watermarking to embed the copyright in the image or the song itself. Transaction tracking is another interesting application of watermarking. In this case the watermark embedded in a digital work can be used to record one or more transactions taking place in the history of a copy of this work. For example, watermarking could be used to record the recipient of every legal copy of a movie by embedding a different watermark in each copy. If the movie is then leaked to the Internet, the movie producers could identify which recipient of the movie was the source of the leak.

Finally, copy control is a very promising application for watermarking. In this application, watermarking can be used to prevent the illegal copying of songs, images of movies, by embedding a watermark in them that would instruct a watermarking- compatible DVD or CD writer to not write the song or movie because it is an illegal copy.

Watermarking properties

Every watermarking system has some very important desirable properties. Some of these properties are often conflicting and we are often forced to accept some trade- offs between these properties depending on the application of the watermarking system. The first and perhaps most important property is effectiveness. This is the probability that the message in a watermarked image will be correctly detected. We ideally need this probability to be 1. Another important property is the image fidelity. Watermarking is a process that alters an original image to add a message to it, therefore it inevitably affects the image's quality. We want to keep this degradation of the image's quality to a minimum, so no obvious difference in the image's fidelity can be noticed. The third property is the payload size. Every watermarked work is used to carry a message. The size of this message is often important as many systems require a relatively big payload to be embedded in a cover work. There are of course applications that only need a single bit to be embedded. The false positive rate is also very important to watermarking systems. This is the number of digital works that are identified to have a watermark embedded when in fact they have no watermark embedded. This should be kept very low for watermarking systems. Lastly, robustness is crucial for most watermarking systems. There are many cases in which a watermarked work is altered during its lifetime, either by transmission over a lossy channel or several malicious attacks that try to remove the watermark or make it undetectable. A robust watermark should be able to withstand additive Gaussian noise, compression, printing and scanning, rotation, scaling, cropping and many other operations.

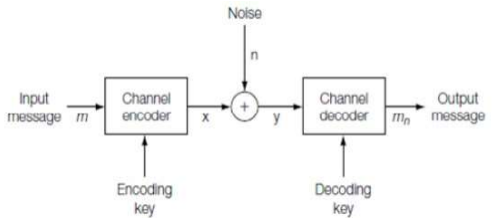
Watermarking models

There are several ways in which we can model a watermarking process. These can be broadly classified in one of two groups. The first group contains models which are based on a communication-based view of watermarking and the second group contains models based on a geometric view of watermarking.

Communication-based models

Communication-based models describe watermarking in a way very similar to the traditional models of communication systems. Watermarking is in fact a process of communicating a message from the watermarking embedder to the watermarking receiver. Therefore, it makes sense to use the models of secure communication to model this process.

In a general secure communication model we would have the sender on one side, which would encode a message using some kind of encoding key to prevent eavesdroppers to decode the message if the message was intercepted during transmission. Then the message would be transmitted on a communications channel, which would add some noise to the noise to the encoded message. The resulting noisy message would be received at the other end of the transmission by the receiver, which would try to decode it using a decoding key, to get the original message back. This process can be seen in Figure.



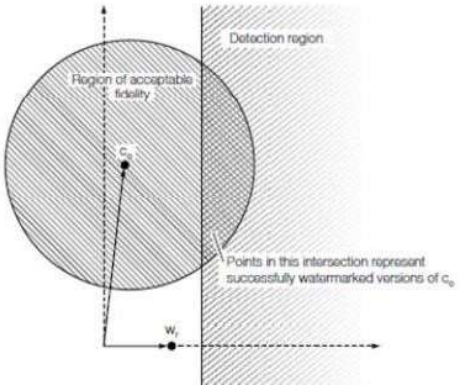
Standard model of a communications channel with key-based encoding

In general, communication-based watermarking models can be further divided into two subcategories. The first uses side-information to enhance the process of watermarking and the second does not use side-information at all. The term side- information refers to any auxiliary information except the input message itself, that can be used to better encode or decode it. The best example of this is the image used to carry the message, which can be used to provide useful information to enhance the correct detection of the message at the receiver.

Geometric models

It is often useful to think of watermarking in geometric terms. In this type of model, images, watermarked and unwatermarked, can be viewed as high-dimensional vectors, in what is called the media space. This is also a high-dimensional space that contains all possible images of all dimensions. For example a 512 X 512 image would be described as a 262144 elements vector in a 262144-dimensional space. Geometric models can be very useful to better visualize the watermarking process using a number of regions based on the desirable properties of watermarking. One of these regions is the embedding region, which is the region that contains all the possible images resulting from the embedding of a message inside an unwatermarked image using some watermark embedding algorithm. Another very important region is the detection region, which is the region containing all the possible images from which a watermark can be successfully extracted using a watermark detection algorithm. Lastly, the region of acceptable fidelity contains all the possible images resulting from the embedding of a message into an

unwatermarked image, which essentially look identical to the original image. The embedding region for a given watermarking system should ideally lie inside the intersection of the detection region and the region of acceptable fidelity, in order to produce successfully detected watermarks that do not alter the image quality very much. An example of a geometric model can be seen in Figure . Here we can see that if mean square error (MSE) is used as a measure of fidelity, the region of acceptable fidelity would be an n-dimensional sphere centred on the original unwatermarked image (co), with a radius defined by the largest MSE we are willing to accept for images with acceptable fidelity. The detection region for a detection algorithm based on linear correlation would be defined as a half space, based on the threshold used to decide whether an image has a watermark embedded or not. Note that the diagram is merely a projection of an ndimensional space into a 2d space.



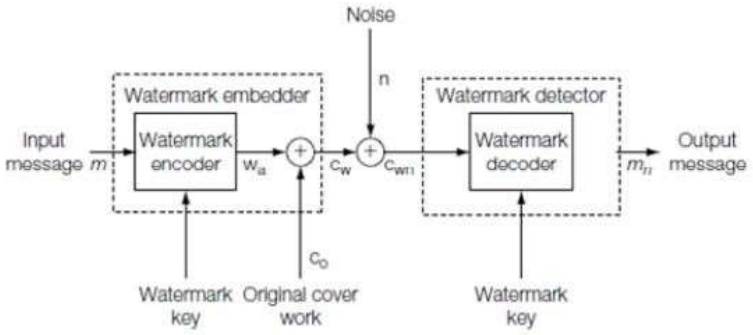
The region of acceptable fidelity (defined by MSE) and the detection region (defined by linear correlation)

When thinking about complex watermarking systems, it is sometimes more useful to consider a projection of the media space into a possibly lower-dimension marking space in which the watermarking then takes place as usual. This projection can be handled more easily by computers because of the smaller number of vector elements and can be possibly expressed by block-based watermarking algorithms which separate images into blocks instead of operating on a pixel basis

Watermarking without side-information

As described earlier, some communication-based watermarking models do not take advantage of the channel side-information. In this kind of models, the image is simply considered as another form of channel noise that distorts the message during its transmission. This can be seen in Figure

3. The watermark embedder encodes a message using a watermark encoder and a key. This is then added to the original image and transmitted over the communication channel which adds some noise. The watermark detector at the other end receives the noisy watermarked image and tries to decode the original image using a key.



Standard model for watermarking with no side-information

Blind embedding and linear correlation detection

This system is an example of blind embedding, which does not exploit the original image statistics to embed a message in an image. The detection is done using linear correlation. This system is a 1-bit watermarking system, in other words it only embeds one bit (a 1 or 0) inside the cover image. The algorithm for the embedder and the detector is as follows:

Embedder:

1. Choose a random reference pattern. This is simply an array with the same dimensions as the original image, whose elements are drawn from a random Gaussian distribution in the interval [-1, 1]. The watermarking key is the seed that is used to initiate the pseudorandom number generator that creates the random reference pattern.

2. Calculate a message pattern depending on whether we are embedding a 1 or a 0. For a 1, leave the random reference pattern as it is. For a 0, take its negative to get the message pattern.

3. Scale the message pattern by a constant α which is used to control the embedding strength. For higher values of α we have more robust embedding, at the expense of loosing image quality.

The value used at the initial experiment was $\alpha = 1$.

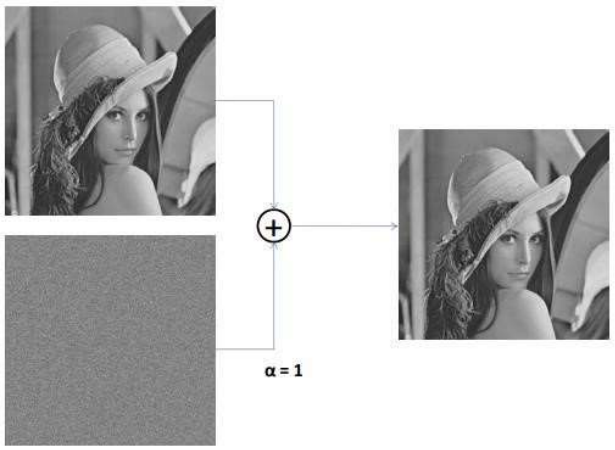
4. Add the scaled message pattern to the original image to get the watermarked image.

Detector:

1. Calculate the linear correlation between the watermarked image that was received and the initial reference pattern that can be recreated using the initial seed which acted as the watermarking key.

2. Decide what the watermark message was, according to the result of the correlation. If the linear correlation value was above a threshold, we say that the message was a 1. If the linear correlation was below the negative of the threshold we say that the message was a 0. If the linear correlation was between the negative and the positive threshold we say that no message was embedded. An example of the embedding process can be seen in figure .

The top left image is the original image, the bottom left image is the reference pattern and the watermarked image resulting from embedding a 1, with α=1, is seen on the right. As we can see, there is no perceptual difference between the original and the watermarked image.



To test the effectiveness of this system, I used 400 small images (112 X 92 pixels). I ran the embedding algorithm with α = 1 and tried to embed a 1 and a 0 in each of these images, resulting in 800 watermarked images. I then calculated the linear correlation (as in the detector) for each one of those images, as well as for the original, unwatermarked images and plotted the result. This can be seen in Figure 5. The x-axis is the linear correlation value, while the y axis is the percentage of images that had a specific linear correlation value. As we can see, most of the images that had a 0 embedded had a value centred on -1. Those that had a 1 embedded had a value centred on 1,

while unwatermarked images had a value centred on 0. The problem is that the three graphs overlap at points between -1 and 0, as well as between 0 and 1. This means that choosing a specific threshold can be difficult because no matter what threshold we chose, there will always be some unwatermarked images classified as having a watermark (false positives) while some others that had a watermark will be classified as unwatermarked (false negatives). Ideally we want these three graphs to be non-overlapping and far from each other so we can choose a threshold that gives no false positives or false negatives.

IMAGE STEGANOGRAPHY

Steganography is the art of hiding the fact that communication is taking place, by hiding information in other information. Many different carrier file formats can be used, but digital images are the most popular because of their frequency on the Internet. For hiding secret information in images, there exists a large variety of steganographic techniques some are more complex than others and all of them have respective strong and weak points. Different applications have different requirements of the steganography technique used. For example, some applications may require absolute invisibility of the secret information, while others require a larger secret message to be hidden.

Since the rise of the Internet one of the most important factors of information technology and communication has been the security of information. Cryptography was created as a technique for securing the secrecy of communication and many different methods have been developed to encrypt and decrypt data in order to keep the message secret. Unfortunately it is sometimes not enough to keep the contents of a message secret, it may also be necessary to keep the existence of the message secret. The technique used to implement this, is called steganography. Steganography is the art and science of invisible communication. This is accomplished through hiding information in other information, thus hiding the existence of the communicated information. The word steganography is derived from the Greek words “stegos” meaning

“cover” and “grafia” meaning “writing” defining it as “covered writing”. In image steganography the information is hidden exclusively in images. The idea and practice of hiding information has a long history. In Histories the Greek historian Herodotus writes of a nobleman, Histaeus, who

needed to communicate with his son-in-law in Greece. He shaved the head of one of his most trusted slaves and tattooed the message onto the slave's scalp. When the slave's hair grew back the slave was dispatched with the hidden message.. In the Second World War the Microdot technique was developed by the Germans. Information, especially photographs, was reduced in size until it was the size of a typed period. Extremely difficult to detect, a normal cover message was sent over an insecure channel with one of the periods on the paper containing hidden information .Today steganography is mostly used on computers with digital data being the carriers and networks being the high speed delivery channels. Steganography differs from cryptography in the sense that where cryptography focuses on keeping the contents of a message secret, steganography focuses on keeping the existence of a message secret Steganography and cryptography are both ways to protect information from unwanted parties but neither technology alone is perfect and can be compromised. Once the presence of hidden information is revealed or even suspected, the purpose of steganography is partly defeated . The strength of steganography can thus be amplified by combining it with cryptography.

images are the most popular cover objects used for steganography. In the domain of digital images many different image file formats exist, most of them for specific applications. For these different image file formats, different steganographic algorithms exist.

JPEG steganography

Originally it was thought that steganography would not be possible to use with JPEG images, since they use lossy compression which results in parts of the image data being altered. One of the major characteristics of steganography is the fact that information is hidden in the redundant bits of an object and since redundant bits are left out when using JPEG it was feared that the hidden message would be destroyed. Even if one could somehow keep the message intact it would be difficult to embed the message without the changes being noticeable because of the harsh compression applied. However, properties of the compression algorithm have been exploited in order to develop a steganographic algorithm for JPEGs. One of these properties of JPEG is exploited to make the changes to the image invisible to the human eye. During the DCT transformation phase of the compression algorithm, rounding errors occur in the coefficient data that are not noticeable. Although this property is what classifies the algorithm as being lossy, this property can also be used to hide messages. It is neither feasible nor possible to embed information in an image that uses lossy compression, since the compression would destroy all information in the process. Thus

it is important to recognize that the JPEG compression algorithm is actually divided into lossy and lossless stages. The DCT and the quantization phase form part of the lossy stage, while the Huffman encoding used to further compress the data is lossless. Steganography can take place between these two stages. Using the same principles of LSB insertion the message can be embedded into the least significant bits of the coefficients before applying the Huffman encoding. By embedding the information at this stage, in the transform domain, it is extremely difficult to detect, since it is not in the visual domain.

Digital images (those that appear on your computer) are broken up into pixels – tiny dots with a specific colour that together make up the image you can see. For images, steganographers encode the message into the pixel LSB. This means that, to the human eye, the colour of the pixel (represented by binary code to the computer) does not change. The hidden message can be withdrawn from the picture provided you know: a) that there is a message in the image b) that you use the same steganographic program for decoding as the one used to hide the message.

The resulting steganographic image Steganographic images are detectable. They do not appear any different to the human eye, but computers, programmed to look for them, can notice slight colour variations when modifying the LSB. It is for this reason that many security experts doubt the practicality of using steganography. If this proves to be the case, other methods, like encryption, can also be used. Some programs will not only code your message into an image, but will encrypt it, too. The steganalysts (those responsible for decoding steganographic messages) would still have to break the encryption in the message extracted from the image.

Image steganography is about exploiting the limited powers of the human visual system (HVS). Within reason, any plain text, ciphertext, other images, or anything that can be embedded in a bit stream can be hidden in an image. Image steganography has come quite far in recent years with the development of fast, powerful graphical computers, and steganographic software is now readily available over the Internet for everyday users.

Concealment in digital images

Information can be hidden many different ways in images. To hide information, straight message insertion may encode every bit of information in the message or selectively embed the message in “noisy” areas that draw less attention- those areas where there is a great deal of natural color

variation. The message may also be scattered randomly throughout the image. Redundant pattern encoding “wallpapers” the cover image with the message.

A number of ways exist to hide information in digital images. Common approaches include:

Least significant bit (LSB) insertion.

Masking and filtering.

Algorithms and transformations.

Least significant bit insertion

The least significant bit insertion method is probably the most well known image steganography technique. It is a common, simple approach to embedding information in a graphical image file. Unfortunately, it is extremely vulnerable to attacks, such as image manipulation. A simple conversion from a GIF or BMP format to a lossy compression format such as JPEG can destroy the hidden information in the image.

When applying LSB techniques to each byte of a 24-bit image, three bits can be encoded into each pixel. (As each pixel is represented by three bytes.) Any changes in the pixel bits will be indiscernible to the human eye. For example, the letter A can be hidden in three pixels. Assume the original three pixels are represented by the three 24-bit words below:

(00100111 11101001 11001000)
(00100111 11001000 11101001)
(11001000 00100111 11101001)
The binary value for the letter A is (10000011). Inserting the binary value of A into the three pixels, starting from the top left byte, would result in:
(0010011111101000 11001000)
(00100110 11001000 11101000)
When applying LSB techniques to each byte of a 24-bit image, three bits can be encoded into each pixel. (As each pixel is represented by three bytes.) Any changes in the pixel bits will be indiscernible to the human eye. For example, the letter A can be hidden in three pixels. Assume the original three pixels are represented by the three 24-bit words below:

(00100111 11101001 11001000)

(00100111 11001000 11101001)

(11001000 00100111 11101001)

The binary value for the letter A is (10000011). Inserting the binary value of A into the three pixels, starting from the top left byte, would result in:

(0010011111101000 11001000)

(00100110 11001000 11101000)

Masking and Filtering

Masking and filtering techniques usually restricted to 24-bit and tray-scale images hide information by marking an image, in a manner similar to paper watermarks. Watermarking techniques may be applied without fear of image destruction due to lossy compression because they are more integrated into the image.

Visible watermarks are not steganography by definition. The difference is primarily one of intent. Traditional steganography conceals information; watermarks extend information and become an attribute of the cover image. Digital watermarks may include such information as copyright, ownership, or license. In steganography, the object of communication is the hidden message. In digital watermarking, the object of communication is the cover.

To create a watermarked image, we increase the luminance of the masked area by 15 percent. If we were to change the luminance by a smaller percentage, the mask would be undetected by the human eye. Now we can use the watermarked image to hide plaintext or encoded information.
Masking is more robust than LSB insertion with respect to compression, cropping and some image processing. Masking techniques embed information in more significant areas so that the hidden message is more integral to the cover image than just hiding it in the “noise” level. This makes it more suitable than LSB with lossy JPEG images.

LSB manipulation is a quick and easy way to hide information but is vulnerable to small changes resulting from image processing or lossy compression. Such compression is a key advantage that JPEG images have over other formats. High quality images can be stored in relatively small files using JPEG compression method.

One steganographic method that integrates the compression algorithm for hiding the information is Jpeg-Jsteg.

Jpeg-Jsteg creates a JPEG stego image from the input of a message to be hidden and a lossless cover mage.

Another method used in Patchwork and similar techniques is the redundant pattern encoding. Here the hidden information is scattered throughout the cover mage. These approaches may help protect against image processing such as cropping and rotations and they hide information more thoroughly than by simply masking. They also support image manipulation more readily than tools that rely on LSB. In using redundant pattern encoding, you must trade off message size against robustness. A large message may be embedded only once because it would occupy a much greater portion of the image area.

Other techniques encrypt and scatter the hidden data throughout an image. Scattering the message makes it appear more like noise. Proponents of this approach assume that even if the message bits are extracted, they will be useless without the algorithm and stego-key to decode them.