

Early Detection Of Cardiac Arrest

at

**Sathyabama Institute of Science and Technology
(DEEMED TO BE UNIVERSITY)**

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering
Degree in Computer Science and Engineering

By

**CHERUKURIASWANTH
(REG NO37110137)
CHIKATLA VENKATA SIVAKOUSHIK
(REG NO37110142)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING SCHOOL OF
COMPITING**

**SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY JEPPIAR
NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU**

March – 2021



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC



(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI- 600119

www.sathyabamauniversity.ac.in

SCHOOL OF COMPUTING

BONAFIDE CERTIFICATE

This is to certify that the project Report is the Bonafide work of **CHERUKURI ASWANTH (Reg. No. 37110137), CHIKATLA VENKATA SIVA KOUSHIK (Reg. No. 37110142)** who carried out the project entitled "**Early Detection Of Cardiac Arrest**" under our supervision from November 2020 to March 2021.

Internal Guide

Dr. M. SARAVANAN, M.E., Ph.D.,

Head of the Department

Dr. S. VIGNESHWARI, M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

We **CHERUKURI ASWANTH (Reg.No. 37110137), CHIKATLAVENKATASIVA KOUSHIK (Reg.No.37110142)** , here by declare that the Professional Training Report on" **Early Detection Of Cardiac Arrest**" done under the guidance of **Dr. M. SARAVANAN, M.E., Ph.D.**, at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE:CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. SASIKALA, M.E., Ph.D., Dean, School of Computing** and **Dr. S. VIGNESHWARI, M.E., Ph.D., Head of the Department, Department of Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. M. SARAVANAN, M.E., Ph.D.,** for his valuable guidance, suggestions, and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of **Computer Science and Engineering** who were helpful in many ways For the Completion of the project.

ABSTRACT:

In India, about 25 percent deaths occur in the age group of 25-69 years because of a cardiac arrest. Many people among us lose their life because of heart attack. After the occurrence of heart attack only, the patient can be monitored. To overcome and help our society from heart attack, we are developing such a system which helps to decrease the death rate and early detection of a heart attack. This heart attack detection system helps to inform if a person is about to have a heart attack by analyzing the number of beats per minute (BPM) and informs as soon as the heart beat level does not fall within the permissible limit. Thus this system can be used to save life of many people. In advance we detect the prior occurrence of cardiac attack of a patient using their monitored lungs datasets using image processing along with IOT. Lung cancer is a very dangerous and rapidly spread disease in the world. In the investigating cases for suspected lung cancer, chest radiography is not only the key techniques of diagnosis based on the medical imaging but also the diagnostic radiology. In this project we proposed the automatic identification of tuberculosis affect in lung images. This detection helps the patient to take prior safety measures by the instructions given by their doctor who has monitored the prior occurrence of the attack.

TABLE OF CONTENTS

Chapter No	TITLE	Page No.
	ABSTRACT	v
	LIST OF FIGURES	vii
	LIST OF ABBREVIATIONS	viii
1	INTRODUCTION	1
	1.1 IMAGE	1
	1.2 IMAGE PROCESSING	5
	1.3 TUBERCULOSIS	31
2	LITERATURE SURVEY	38
3	THEME OF THE PROJECT	44
	3.1 FLOW DIAGRAM	44
	3.2 FLOW DIAGRAM DISCRIPTION	44
	3.3 BLOCK EXPLANATION	45
4	SOFTWARE AND HARDWARE	71
	SYSTEM REQRIMENTS FOR MATLAB	71
	WHAT IS MATLAB	72
5	CONCLUSION	77
6	REFERENCES	78

LIST OF FIGURES

FIGURE No.	FIGURE NAME	PAGE No.
1.1.1	PIXEL REPRESENTATION	3
1.2	STRUCTURE OF GREY SCALE IMAGE	7
1.2.6	THE GREY VALUE HISTOGRAM	16
1.2.7	IMAGE ON HANCEMENT	18
1.2.8	IMAGE RESTORATION	19
3.1	FLOW DIAGRAM	46

LIST OF ABBREVIATIONS

MTB	MYCOBACTERIUM TUBERCULOSIS
TST	TUBERCULIN SKIN TEST
MDR-TB	MULTIPLE DRUG-RESISTANT TUBERCULOSIS
NTM	NON TUBERCULOSIS MYCOBACTERIA
IGRA	INTERFERON GAMMA RELEASE ASSAYS
HWT	HAAR WAVELET TRANSFORMATION
CXR	CHEST-X-RAY
HOG	HISTOGRAM ORIENTED GRADIENTS
SVM	SUPPORT VECTOR MACHINE
SVC	SUPPORT VECTOR CLUSTERING

CHAPTER-1

INTRODUCTION

IMAGE

An image is an array or a matrix of square pixels (picture elements) arranged in columns and rows. An image (from Latin: imago) is an artifact, for example a two-dimensional picture, that has a similar appearance to some subject usually a physical object or a person.

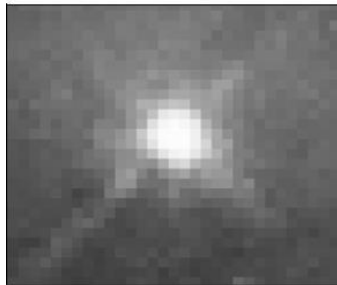


Figure : An image – an array or a matrix of pixels arranged in columns and rows

PIXEL

Image processing is a subset of the electronic domain where in the image is converted to an array of small integers, called *pixels*, representing a physical quantity such as scene radiance, stored in a digital memory and processed by computer or other digital hardware.

Suppose we take an image, a photo, say. For the moment, let's make things easy and suppose the photo is black and white (that is, lots

of shades of grey), so no colour. We may consider this image as being a two-dimensional function, where the function values give the brightness of the image at any given point, as shown in figure 1. We may assume that in such an image brightness values can be any real numbers in the range 0.0 (black) to 1.0 (white). The ranges of x and y will clearly depend on the image, but they can take all real values between their minima and maxima.

A digital image differs from a photo in that the x , y , and $f(x, y)$ values are all discrete. Usually they take on only integer values, so the image shown in figure 1.2 will have x and y ranging from 1 to 256 each, and the brightness values also ranging from 0 (black) to 255 (white). A digital image can be considered as a large array of discrete dots, each of which has a brightness associated with it. These dots are called picture elements, or more simply pixels. The pixels surrounding a given pixel constitute its neighbourhood. A neighbourhood can be characterized by its shape in the same way as a matrix: we can speak of a 3×3 neighbourhood, or of a 5×7 neighbourhood. Except in very special circumstances, neighbourhoods have odd numbers of rows and columns; this ensures that the current pixel is in the center of the neighbourhood. An example of a neighbourhood is given in figure 1.3. If a neighbourhood has an even number of rows or columns (or both), it may be necessary to specify which pixel in the neighbourhood is the "current pixel".

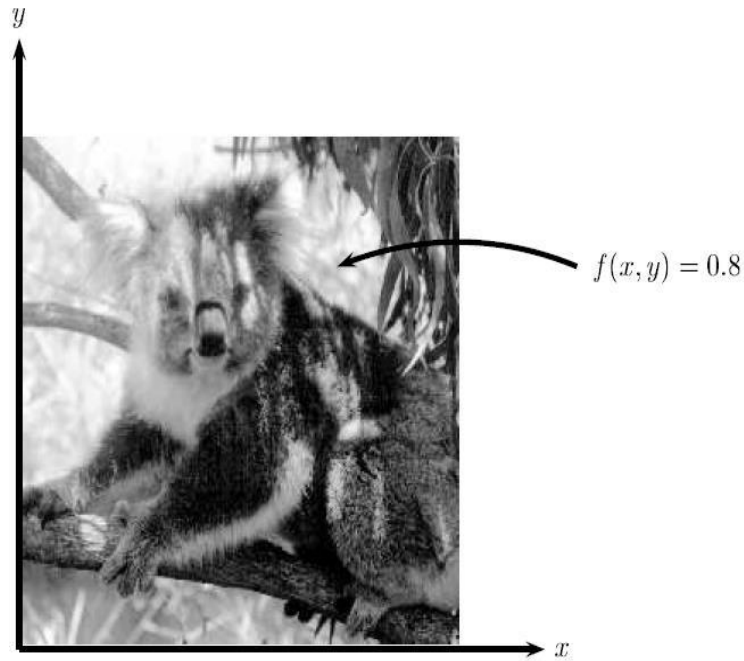


Figure :A gray scaleimage

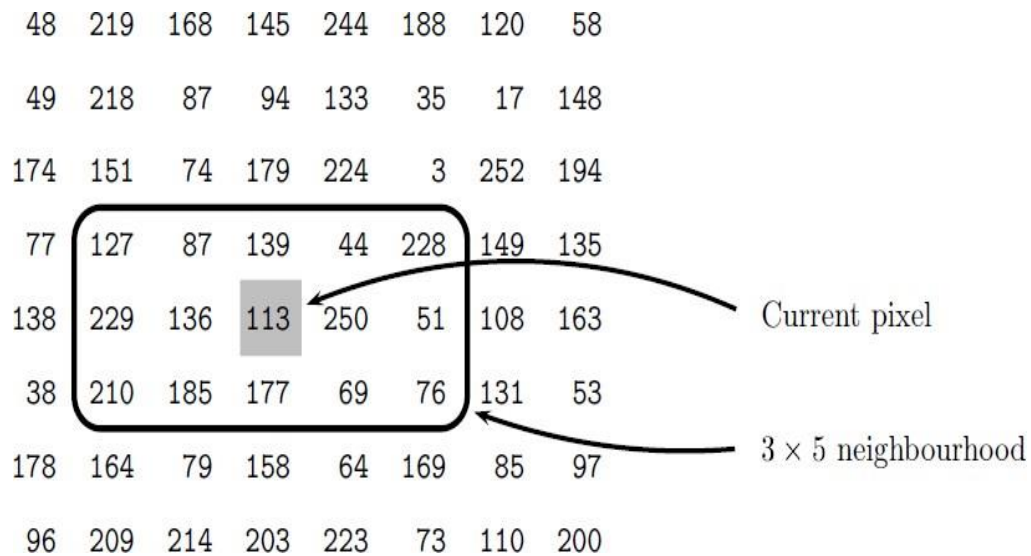


Figure: Pixelrepresentation

IMAGE RESOLUTION:

Pixels transform into inches through what is called “resolution,” -- the number of pixels per square inch on a computer. Resolution allows you to transform pixels into inches and back again.

Two resolution definitions are often used in place of one another. Pixel resolution is the size (in bytes) of your image or its appearance on a computer screen. This number is tied directly to how big your image is on your hard drive. The byte-size of the image file is directly proportional to the pixel count and its size on your computer screen, which simply displays all the pixels in a fixed one-to-one grid.

Embedded resolution is different. Embedded resolution tells your printer how far apart to spread the pixels in a printed image. It determines how “fine grained” the printed image will look. It is completely independent of the pixel count (file size) of the image. A high-pixel-count image can have a low embedded resolution or vice versa. Given the same pixel count, a high embedded resolution will result in a smaller printed image (the pixels are packed together more tightly), and a low embedded resolution will result in a larger image (the pixels are more spread out).

IMAGE BRIGHTNESS

‘Brightness’ is a problem word, used in most descriptions of image quality. Often, if an image looks bad we will say that it’s not ‘bright’, when it merely lacks contrast. So what does ‘brightness’ actually mean? Brightness actually describes how we experience light and not ‘how it is’. If we are going to describe light and ‘brightness’ properly, there are two essential terms:

Luminance and illuminance

Luminance is the light we see: reflected or radiating from objects. It is measured in candela per square metre (cd/m² and the same as ‘nit’).

Illuminance is light we can't see directly – it is so-called ambient light, light passing through air, actually invisible until it reflects off an object (which we then see as luminance). It is Measured in lux (lx or lm/m²).

Our eyes are exposed to such dramatic changes in light levels (far greater than a million to one) that our optical systems have to protect them. We can read newspaper headlines by the light of a full moon, which might be 0.2 lux of illuminance. Yet we can also read outside in direct sunshine, in more than 100,000 lux. So our eyes constantly adjust – our irises contract and expand – to maintain vision consistently, protecting against damage to the retina at the back of the eye. (That reflex to look away from bright lights is important!) Right away we can understand why a screen that looks so bright in a showroom can be barely visible in a shopwindow!

CONTRAST

Contrast refers to the difference between black and white levels in images, whether on a flat panel display or a projection screen. Without good contrast, images appear to lack 'brightness', colour and definition.

The image contrast ratio refers to the difference between the luminance of the white part of an image, divided by the black part. So if the white part is one hundred times brighter than the black part, it will be 100:1, and soon.

IMAGE PROCESSING

Image processing is any form of signal processing for which the input is an image, such as a photograph or video frame; the output of image processing may be either an image or a set of characteristics or parameters related to the image. Most image- processing techniques involve treating the image as a dimensional signal and applying standard signal-processing techniques to it.

Image processing usually refers to digital image processing, but optical and analog image processing also are possible. This article is about general techniques that apply to all of them. The acquisition of images (producing the input image in the first place) is referred to as imaging

Image processing allows one to enhance image features of interest while attenuating detail irrelevant to a given application, and then extract useful information about the scene from the enhanced image. This introduction is a practical guide to the challenges, and the hardware and algorithms used to meet them.

An image is digitized to convert it to a form which can be stored in a computer's memory or on some form of storage media such as a hard disk or CD-ROM. This digitization procedure can be done by a scanner, or by a video camera connected to a frame grabber board in a computer. Once the image has been digitized, it can be operated upon by various image processing operations.

Image processing operations can be roughly divided into three major categories, Image Compression, Image Enhancement and Restoration, and Measurement Extraction. Image compression is familiar to most people. It involves reducing the amount of memory needed to store a digital image.

Image defects which could be caused by the digitization process or by faults in the imaging set-up (for example, bad lighting) can be corrected using Image Enhancement techniques.

Once the image is in good condition, the Measurement Extraction operations can be used to obtain useful information from the image.

The example in figure operate on 256 gray-scale images. This means that each pixel in the image is stored as a number between 0 to 255, where 0 represents a black pixel, 255 represents a white pixel and values in-between represent shades of gray.

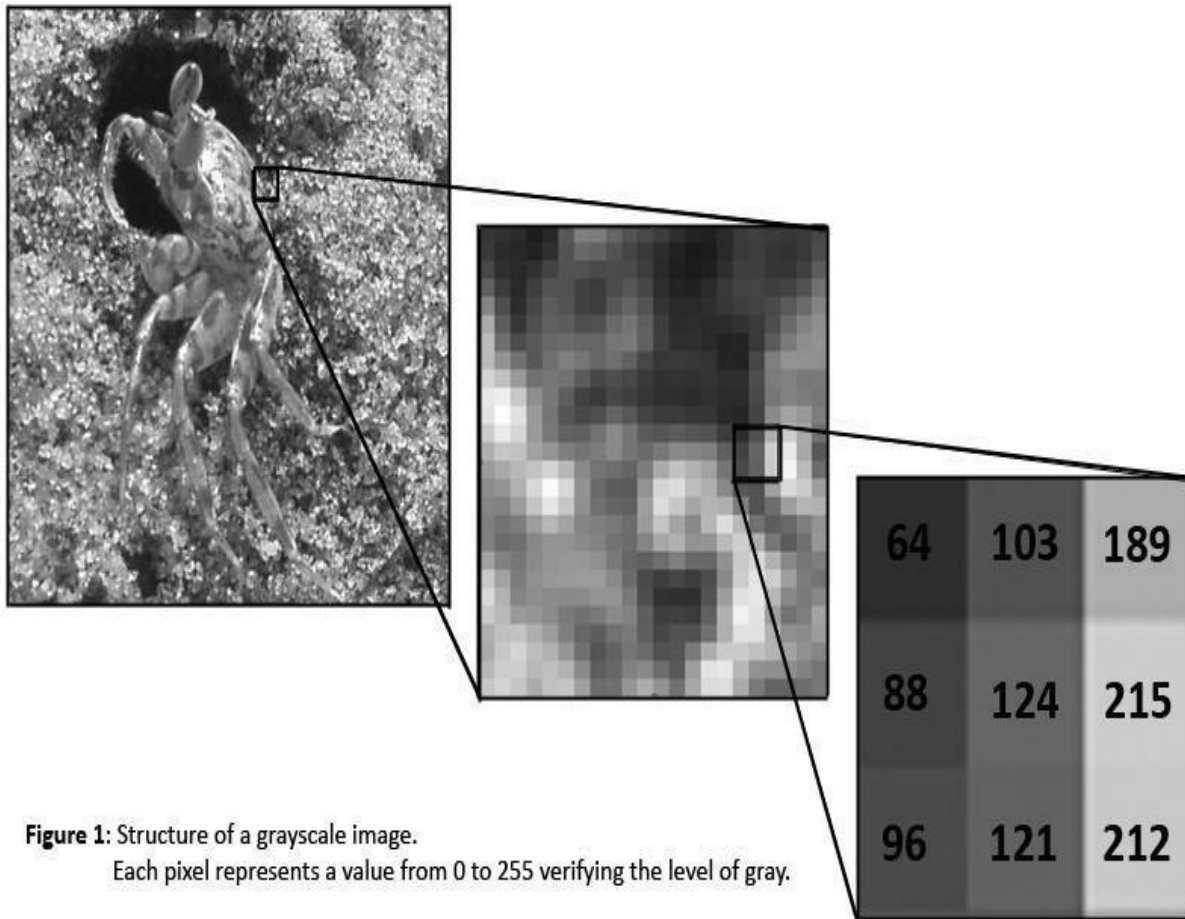


Figure :Structure of grey-scale image

In figure each pixel represents a value from 0 to 255 verifying the level of gray. These operations can be extended to colour images too.

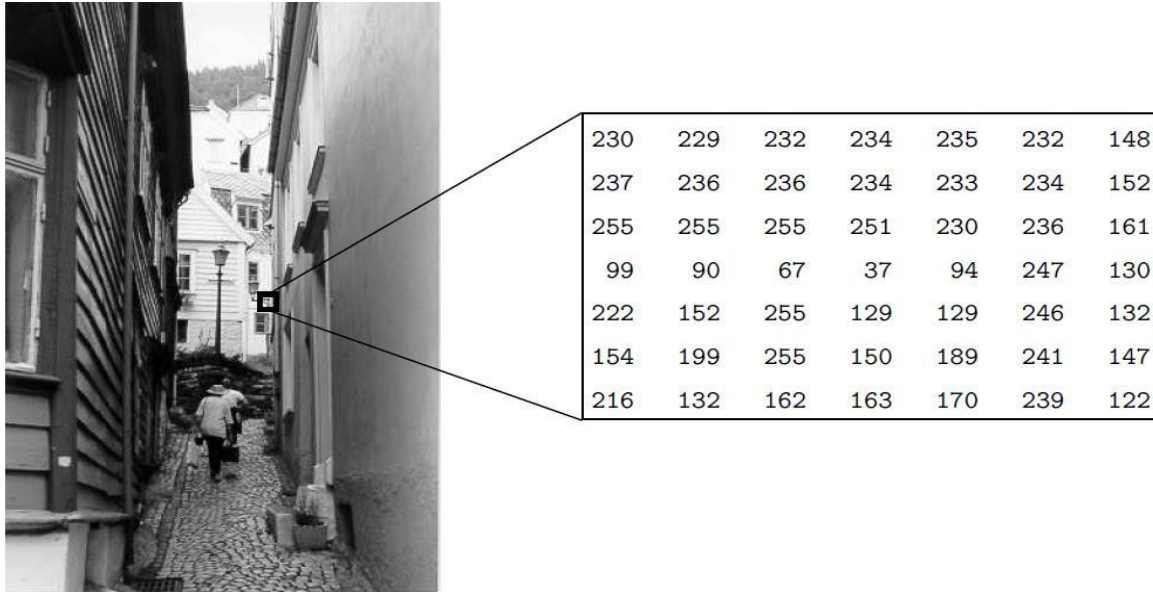


Figure: A greyscale image

In a (8-bit) greyscale image each picture element has an assigned intensity that ranges from 0 to 255. A grey scale image is what people normally call a black and white image, but the name emphasizes that such an image will also include many shades of grey.

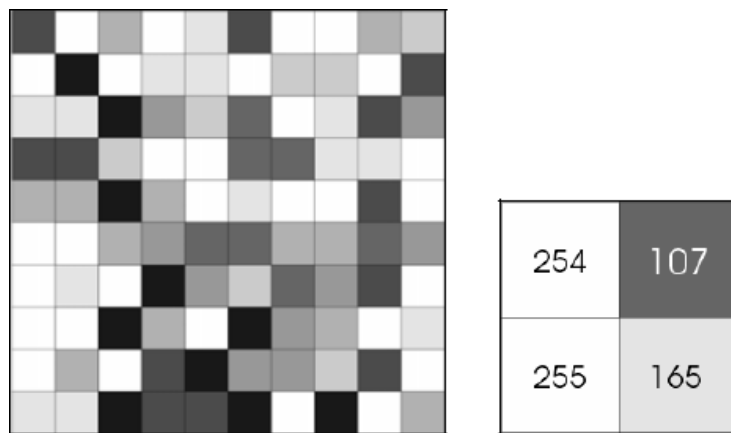


Figure: Different shades of grey

A normal greyscale image has 8 bitcolour depth = 256 greyscales

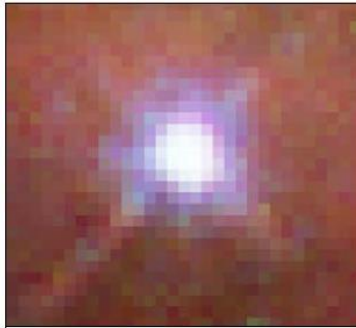


Figure: Colour image

A “true colour” image has 24 bitcolour depth = $8 \times 8 \times 8$ bits = $256 \times 256 \times 256$ colours
= ~16 million colours.



49	55	56	57	52	53	64	76	82	79	78	78	66	80	77	80	87	77
58	60	60	58	55	57	93	93	91	91	86	86	81	93	96	99	86	85
58	58	54	53	55	56	88	82	88	90	88	89	83	83	91	94	92	88
83	78	72	69	68	69	125	119	113	108	111	110	135	128	126	112	107	106
88	91	91	84	83	82	137	136	132	128	126	120	141	129	129	117	115	101
69	76	83	78	76	75	105	108	114	114	118	113	95	99	109	108	112	109
61	69	73	78	76	76	96	103	112	108	111	107	84	93	107	101	105	102

Red

Green

Blue

Figure: A true colour image

: IMAGE ENHANCEMENT

Image enhancement is the improvement of digital image quality (wanted e.g. for visual inspection or for machine analysis), without knowledge about the source of degradation. If the source of degradation is known, one calls the process image restoration. Both are iconic processes, viz. input and output are images.

Many different, often elementary and heuristic methods are used to improve images in some sense. The problem is, of course, not well defined, as there is no objective measure for image quality. Here, we discuss a few recipes that have shown to be useful both for the human observer and/or for machine recognition. These methods are very problem-oriented: a method that works fine in one case may be completely inadequate for another problem.

Apart from geometrical transformations some preliminary grey level adjustments may be indicated, to take into account imperfections in the acquisition system. This can be done pixel by pixel, calibrating with the output of an image with constant brightness. Frequently space-invariant grey value transformations are also done for contrast stretching, range compression, etc. The critical distribution is the relative frequency of each grey value, the grey value histogram.

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

Basics of image editing

Raster images are stored in a computer in the form of a grid of picture elements, or pixels. These pixels contain the image's color and brightness information. Image editors can change the pixels to enhance the image in many ways. The pixels can be changed as a group, or individually, by the sophisticated algorithms within the image editors. This article mostly refers to bitmap graphics editors, which are often used to alter photographs and other raster graphics. However, vector graphics software, such as Adobe Illustrator, CorelDRAW, Xara Designer Pro, PixelStyle Photo Editor, Inkscape or Vectr, are used to create and modify vector images, which are stored as descriptions of lines, Bézier curves, and text instead of pixels. It is easier to rasterize a vector image than to vectorize a raster image; how to go about vectorizing a raster image is the focus of much research in the field of computer vision. Vector images can be modified more easily, because they contain descriptions of the shapes for easy rearrangement. They are also scalable, being rasterizable at any resolution.

Automatic image enhancement

Camera or computer image editing programs often offer basic automatic image enhancement features that correct color hue and brightness imbalances as well as other image editing features, such as red eye removal, sharpness adjustments, zoom features and automatic cropping. These are called automatic because generally they happen without user interaction or are offered with one click of a button or mouse button or by selecting an option from a menu. Additionally, some automatic editing features offer a combination of editing actions with little or no user interaction.

Digital data compression

Many image file formats use data compression to reduce file size and save storage space. Digital compression of images may take place in the camera, or can be done in the computer with the image editor. When images are stored in JPEG format, compression

has already taken place. Both cameras and computer programs allow the user to set the level of compression.

Some compression algorithms, such as those used in PNG file format, are lossless, which means no information is lost when the file is saved. By contrast, the JPEG file format uses a lossy compression algorithm by which the greater the compression, the more information is lost, ultimately reducing image quality or detail that can not be restored. JPEG uses knowledge of the way the human brain and eyes perceive color to make this loss of detail less noticeable.

Image editor features

Listed below are some of the most used capabilities of the better graphic manipulation programs. The list is by no means all inclusive. There are a myriad of choices associated with the application of most of these features.

Selection

One of the prerequisites for many of the applications mentioned below is a method of selecting part(s) of an image, thus applying a change selectively without affecting the entire picture. Most graphics programs have several means of accomplishing this, such as:

- a marquee tool for selecting rectangular or other regular polygon-shaped regions,
- a lasso tool for freehand selection of a region,
- a magic wand tool that selects objects or regions in the image defined by proximity of color or luminance,
- vector-based pen tools,

as well as more advanced facilities such as edge detection, masking, alpha compositing, and color and channel-based extraction. The border of a selected area in an image is often animated with the marching ants effect to help the user to distinguish the selection border from the image background.

Layers

Another feature common to many graphics applications is that of Layers, which are analogous to sheets of transparent acetate (each containing separate elements that make up a combined picture), stacked on top of each other, each capable of being individually positioned, altered and blended with the layers below, without affecting any of the elements on the other layers. This is a fundamental workflow which has become the norm for the majority of programs on the market today, and enables maximum flexibility for the user while maintaining non-destructive editing principles and ease of use.

Image size alteration

Image editors can resize images in a process often called image scaling, making them larger, or smaller. High image resolution cameras can produce large images which are often reduced in size for Internet use. Image editor programs use a mathematical process called resampling to calculate new pixel values whose spacing is larger or smaller than the original pixel values. Images for Internet use are kept small, say 640 x 480 pixels which would equal 0.3 megapixels.

Cropping an image

Digital editors are used to crop images. Cropping creates a new image by selecting a desired rectangular portion from the image being cropped. The unwanted part of the image is discarded. Image cropping does not reduce the resolution of the areacropped.

Best results are obtained when the original image has a high resolution. A primary reason for cropping is to improve the image composition in the new image.

Histogram

Image editors have provisions to create an image histogram of the image being edited. The histogram plots the number of pixels in the image (vertical axis) with a particular brightness value (horizontal axis). Algorithms in the digital editor allow the user to visually adjust the brightness value of each pixel and to dynamically display the results as adjustments are made. Improvements in picture brightness and contrast can thus be obtained.

Noise reduction

Image editors may feature a number of algorithms which can add or remove noise in an image. Some JPEG artifacts can be removed; dust and scratches can be removed and an image can be de-speckled. Noise reduction merely estimates the state of the scene without the noise and is not a substitute for obtaining a "cleaner" image. Excessive noise reduction leads to a loss of detail, and its application is hence subject to a trade-off between the undesirability of the noise itself and that of the reduction artifacts.

Noise tends to invade images when pictures are taken in low light settings. A new picture can be given an 'antiqued' effect by adding uniform monochrome noise.

Removal of unwanted elements

Most image editors can be used to remove unwanted branches, etc., using a "clone" tool. Removing these distracting elements draws focus to the subject, improving overall composition.

Image orientation

Image editors are capable of altering an image to be rotated in any direction and to any degree. Mirror images can be created and images can be horizontally flipped or vertically flopped. A small rotation of several degrees is often enough to level the horizon, correct verticality (of a building, for example), or both. Rotated images usually require cropping afterwards, in order to remove the resulting gaps at the image edges.

Perspective control and distortion

Some image editors allow the user to distort (or "transform") the shape of an image. While this might also be useful for special effects, it is the preferred method of correcting the typical perspective distortion which results from photographs being taken at an oblique angle to a rectilinear subject. Care is needed while performing this task, as the image is reprocessed using interpolation of adjacent pixels, which may reduce overall image definition. The effect mimics the use of a perspective control lens, which achieves a similar correction in-camera without loss of definition.

Enhancing images

In computer graphics, the process of improving the quality of a digitally stored image by manipulating the image with software. It is quite easy, for example, to make an image lighter or darker, or to increase or decrease contrast. Advanced photo enhancement software also supports many filters for altering images in various ways. Programs specialized for image enhancement are sometimes called image editors.

Sharpening and softening images

Graphics programs can be used to both sharpen and blur images in a number of ways, such as unsharp masking or deconvolution. Portraits often appear more pleasing when selectively softened (particularly the skin and the background) to better make the

subject stand out.[citation needed] This can be achieved with a camera by using a large aperture, or in the image editor by making a selection and then blurring it. Edge enhancement is an extremely common technique used to make images appear sharper, although purists frown on the result as appearing unnatural.

Another form of image sharpening involves a form of contrast. This is done by finding the average color of the pixels around each pixel in a specified radius, and then contrasting that pixel from that average color. This effect makes the image seem clearer, seemingly adding details. An example of this effect can be seen to the right. It is widely used in the printing and photographic industries for increasing the local contrasts and sharpening the images.

Examples of simple grey level transformations in this domain are:

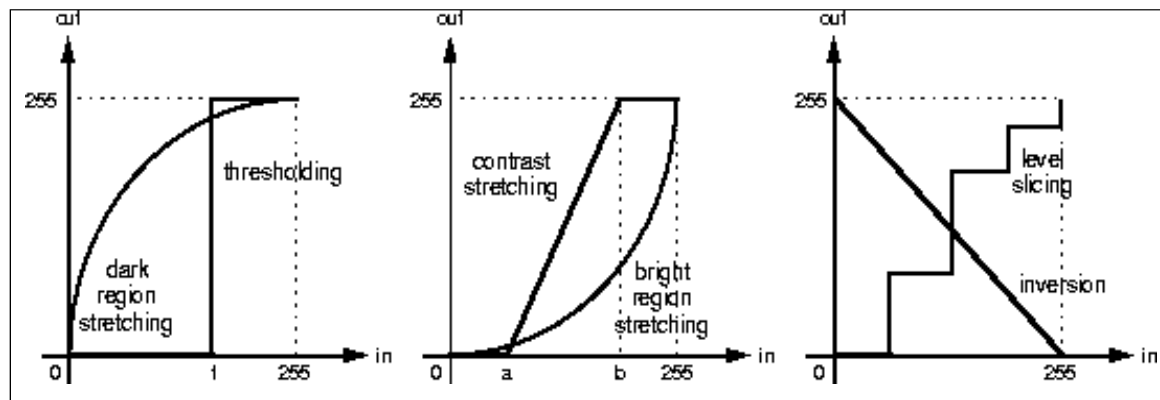


Figure 1.2.6: The grey value histogram

Grey values can also be modified such that their histogram has any desired shape, e.g.flat (every grey value has the same probability). All examples assume *point processing*, viz. each output pixel is the function of one input pixel; usually, the transformation is implemented with a look-uptable:

Physiological experiments have shown that very small changes in luminance are recognized by the human visual system in regions of continuous grey value, and not at all seen in regions of some discontinuities. Therefore, a design goal for image enhancement often is to smooth images in more uniform regions, but to preserve edges. On the other hand, it has also been shown that somehow degraded images with enhancement of certain features, e.g. edges, can simplify image interpretation both for a human observer and for machine recognition. A second design goal, therefore, is image sharpening. All these operations need neighbourhood processing, viz. the output pixel is a function of some neighbourhood of the input pixels:

These operations could be performed using linear operations in either the frequency or the spatial domain. We could, e.g. design, in the frequency domain, one-dimensional low or high pass filters (Filtering), and transform them according to the two-dimensional case.

Unfortunately, linear filter operations do not really satisfy the above two design goals; in this book, we limit ourselves to discussing separately only (and superficially) Smoothing and Sharpening.

Here is a trick that can speed up operations substantially, and serves as an example for both point and neighbourhood processing in a binary image: we number the pixels in a 3x3 neighbourhoodlike:

5	6	7
4	8	0
3	2	1

and denote the binary values (0,1) by b_i ($i = 0,8$); we then concatenate the bits into a 9-bit word, like $b_8b_7b_6b_5b_4b_3b_2b_1b_0$. This leaves us with a 9-bit grey value for each pixel, hence a new image (an 8-bit image with b_8 taken from the original binary image will also do). The new image corresponds to the result of a convolution of the binary image, with a 3×3 matrix containing as coefficients the powers of two. This neighbour image can then be passed through a look-up table to perform erosions, dilations, noise cleaning, skeletonization, etc.

Apart from point and neighbourhood processing, there are also global processing techniques, i.e. methods where every pixel depends on all pixels of the whole image. Histogram methods are usually global, but they can also be used in a neighbourhood.



(a)

(b)

Figure 1.2.7: Image enhancement (a) original image (b) enhanced image

IMAGE RESTORATION

The purpose of image restoration is to "compensate for" or "undo" defects which degrade an image. Degradation comes in many forms such as motion blur, noise, and camera misfocus. In cases like motion blur, it is possible to come up with a very good estimate of the actual blurring function and "undo" the blur to restore the original image. In cases where the image is corrupted by noise, the best we may hope to do is to

compensate for the degradation it caused. In this project, we will introduce and implement several of the methods used in the image processing world to restore images.

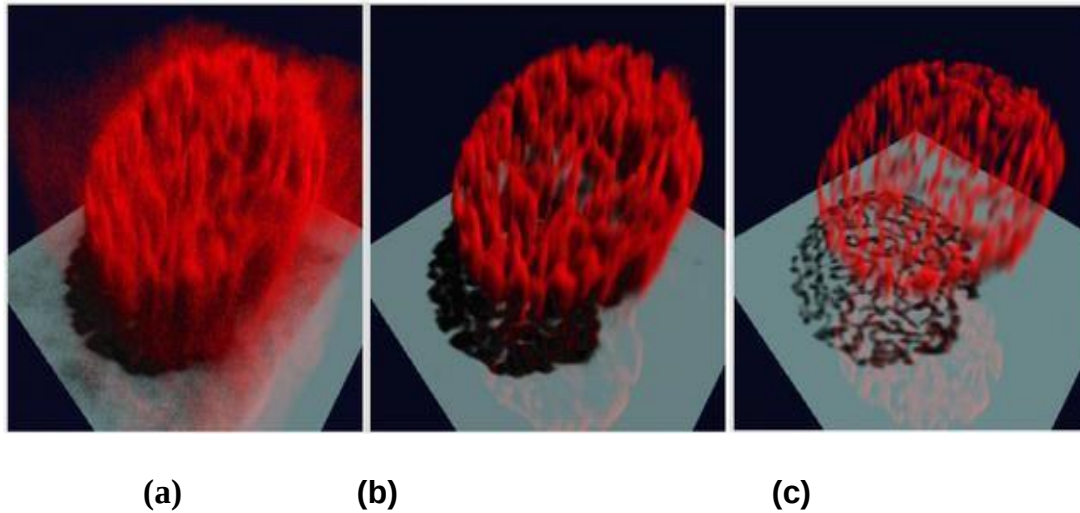


Figure 1.2.8: Image restoration (a) original image (b) partially restored image
(c) fully restored image

OBJECT RECOGNITION

Object recognition includes the process of determining the object's identity or location in space. The problem of object or target recognition starts with the sensing of data with the help of sensors, such as video cameras and thermal sensors, and then interpreting these data in order to recognize an object or objects. We can divide the object-recognition problem into two categories: the modeling problem and the recognition problem.

IMAGE SEGMENTATION

Image segmentation is the process of partitioning a digital image into multiple segments (sets of pixels, also known as super-pixels). The goal of segmentation is to simplify and/or change the representation of an image into something that is more

meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics.

The result of image segmentation is a set of segments that collectively cover the entire image, or a set of contours extracted from the image (see edge detection). Each of the pixels in a region are similar with respect to some characteristic or computed property, such as color, intensity, or texture. Adjacent regions are significantly different with respect to the same characteristic(s). When applied to a stack of images, typical in medical imaging, the resulting contours after image segmentation can be used to create 3D reconstructions with the help of interpolation algorithms like Marching cubes.

Image segmentation is the process of partitioning a digital image into disjointed, meaningful regions. The meaningful regions may represent objects in an image of three-dimensional scene, regions corresponding to industrial, residential, agricultural, or natural terrain in an aerial reconnaissance application, and so on. A region is a connected set of pixels and the objects are considered either four-connected, if only laterally adjacent pixels are considered, or they can be eight-connected, if diagonally adjacent pixels are also considered to be connected. Image segmentation is an efficient and natural process for humans. A human eye (or rather, mind) sees not a complex scene, but rather a collection of objects. In contrast, image segmentation is not an easy task in digital image processing, and it may become a serious problem if the number of objects is large or unknown or if the boundaries between objects are not clear. Three of the most commonly used techniques for digital image segmentation are the gray-level thresholding technique, gradient-based segmentation technique, and the region growing technique. A brief description of the three techniques is given below.

Gray-Level Thresholding

This technique is an example of the regional approach, which implies grouping pixels into distinct regions or objects. When using this technique for image segmentation, all pixels that have gray levels equal to or above a certain threshold are assigned to the object. The rest of the pixels are assigned to the object's background. If the image contains more than one object, then more threshold levels can be used to do the segmentation.

Gradient-Based Segmentation

This technique concentrates on the boundaries between different regions. Reference (5) describes three techniques that are based on the gradient technique. They are the boundary tracking technique, the gradient image thresholding, and Laplacian edge detection. We describe each of these techniques briefly.

Boundary Tracking

This technique starts with scanning the image for the pixel with the highest gradient. This pixel is for sure on the object's boundary. Then a 3-by-3 pixel segment (with the original pixel in its center) is used as tracking probe for the next pixel with the highest gradient, in the neighbor of the original one. This pixel is considered part of the object's boundary and the probe is moved to the new pixel to search for another boundary pixel. The process is repeated until a closed contour is formed. If, at any time, three adjacent pixels in the probe have the same highest gray level (i.e., gradient), the one in the middle is selected. If two have the same highest gradient, the choice is arbitrary.

Gradient Image Thresholding

This technique is based on the following phenomenon. If we threshold a gradient image at moderate gray level, the object and the background stand below the threshold and most edge points stand above it (5). The technique works like this: We first threshold the image at a low level to identify the object and background, which are separated by

bands of edge points. We then gradually increase the threshold level, causing both the object and background to grow. When they finally touch, without merging, the points of contact define the boundary (5).

Laplacian Edge Detection

The Laplacian is a scalar, second-order derivative operator that, for a two-dimensional function $f(x, y)$, can be given by

$$\nabla^2 f(x, y) = \frac{\partial^2}{\partial x^2} f(x, y) + \frac{\partial^2}{\partial y^2} f(x, y)$$

The Laplacian will produce an abrupt zero-crossing at an edge. If a noise-free image has sharp edges, the Laplacian can find them. In the presence of noise, however, low-pass filtering must be used prior to using the Laplacian.

Region Growing

Region growing is one of the artificial intelligence techniques for digital image segmentation. Unlike other techniques, this one is capable of utilizing many properties of the pixels inside the different objects, including average gray level, texture, and color information. The technique works as follows: First, the image is partitioned into many small regions. Then the properties of the pixels in each region are computed. In the next step, average properties of adjacent regions are examined. If the difference is significant, the boundary is considered strong and it is allowed to stay. Otherwise, the boundary is considered weak, and the two regions are merged. The process is repeated until there are no more boundaries weak enough to allow regions merging. This technique is especially useful in cases where prior knowledge of the scene features is not available.

Texture

We can define texture as an attribute representing variations in brightness among the image pixels. This variation may be in the level of brightness, spatial frequencies, or

orientation. Computer-generated images look very realistic for objects of metallic or plastic materials but tend to look smoother than necessary in images of organic materials, such as human skin. Surface roughness is a feature closely related to texture. Texture is sometimes seen as a measure of surface roughness. Random texture is a type of texture caused by noise from cameras, sensors, or films. This type has no recognizable pattern. Satellite images are a good example of images where patterned texture is most recognizable. Different uses of land, such as agriculture or construction, produce different textures. Some of the operators commonly used to detect texture in images are the rank operators, range operators, and gray-level variation. The rank operator measures the range between the maximum and minimum brightness values in the image. Larger range values correspond to surfaces with a larger roughness. The range operator converts the original image into one in which brightness represents texture (4). Gray-level variation can be measured by the sum of the squares of the differences between brightness of the center pixel and its neighbors. The root of resultant value is then calculated to give the root-mean square (rms) measurement of the surface roughness. Research indicates that the human eye is insensitive to texture differences of an order higher than a second(5).

Pattern Recognition

Patterning is a process to extract features and recognize objects and patterns from a given image. Usually there are two approaches to solve the problem of pattern recognition: the RF processing approach and the image-processing approach. In the RF processing approach, electromagnetic waves are transmitted and then reflected from the target. The reflected waves are analyzed to recognize the target. The RF processing is usually used when the target is very far or out of range of sight. In the image processing approach, a picture or an image of the field is taken. The picture is then processed and analyzed to recognize the objects in the picture.

Applications

Some of the practical applications of image segmentation are:

- Content-based image retrieval
- Machine vision
- Medical imaging
 1. Locate tumors and other pathologies
 2. Measure tissue volumes
 3. Diagnosis, study of anatomical structure
 4. Surgery planning
 5. Virtual surgery simulation
 6. Intra-surgery navigation
- Object detection
 1. Pedestrian detection
 2. Face detection
 3. Brake light detection
 4. Locate objects in satellite images (roads, forests, crops, etc.)
- Recognition Tasks
 1. Face recognition
 2. Fingerprint recognition
 3. Iris recognition
- Traffic control systems
- Video surveillance

Several general-purpose algorithms and techniques have been developed for image segmentation. To be useful, these techniques must typically be combined with a domain's specific knowledge in order to effectively solve the domain's segmentation problems.

FEATURE EXTRACTION

In image processing, feature extraction starts from an initial set of measured data and builds derived values (features) intended to be informative and non-redundant,

facilitating the subsequent learning and generalization steps, and in some cases leading to better human interpretations. Feature extraction is related to dimensionality reduction.

When the input data to an algorithm is too large to be processed and it is suspected to be redundant (e.g. the same measurement in both feet and meters, or the repetitiveness of images presented as pixels), then it can be transformed into a reduced set of features (also named a features vector). This process is called feature selection. The selected features are expected to contain the relevant information from the input data, so that the desired task can be performed by using this reduced representation instead of the complete initial data.

General

Feature extraction involves reducing the amount of resources required to describe a large set of data. When performing analysis of complex data one of the major problems stems from the number of variables involved. Analysis with a large number of variables generally requires a large amount of memory and computation power, also it may cause a classification algorithm to overfit to training samples and generalize poorly to new samples. Feature extraction is a general term for methods of constructing combinations of the variables to get around these problems while still describing the data with sufficient accuracy.

The best results are achieved when an expert constructs a set of application-dependent features, a process called feature engineering. Nevertheless, if no such expert knowledge is available, general dimensionality reduction techniques may help. These include:

- Independent component analysis
- Isomap
- Kernel PCA
- Latent semantic analysis

- Partial leastsquares
- Principal componentanalysis
- Multifactor dimensionalityreduction
- Nonlinear dimensionalityreduction
- Multilinear Principal ComponentAnalysis
- Multilinear subspacelearning
- Semidefinite embedding
- Autoencoder
- Deep featuresynthesis

WAVELETS

Wavelets are a relatively new area of signal processing and offer a potentially very useful approach to image analysis. Because of the fact that one has local control over resolution, many cue features of different size can be extracted from an image. Wavelets also show promise as an aid in understanding how early vision can be emulated and further understood.

Wavelet Transforms

It is well known that in signal processing, the more compact your method of representing information the better. In 1822, Jean Joseph Fourier devised a very efficient way to represent the information content of a signal. His idea was to represent a signal as the sum of its frequencies. Transmitted power spectra, carrier frequencies, brain activity, NMR signals - all these global descriptions provide a lot of information in a compact manner. However, most of the power of this kind of representation vanishes when one tries to represent information that changes its nature during the course of signal recording. A good example of this kind of a signal is a musical score. A global analysis of a recording of a musical selection with a Fourier transform (FT) will indicate the specificnotesplayedwithinthepieceofmusic,butthereisnowaytorecoverthetiming

of the notes. The musical score, on the other hand, indicates each note that was played and the time it was played. Wavelets represent a signal in a way such that local frequency information is available for each position within the signal. Wavelets are able to analyze a signal based on the position-varying spectra. The multiresolution pyramidal decomposition that results is also well matched to fractals and shows great potential for the removal of background noise, such as static in recordings, and for pattern recognition and texture segmentation. In the following section we discuss the conceptual understanding of how wavelets can be used in signal analysis.

1.2.4. IMAGE CLASSIFICATION

Image classification refers to the task of extracting information classes from a multiband raster image. The resulting raster from image classification can be used to create thematic maps. Depending on the interaction between the analyst and the computer during classification, there are two types of classification: supervised and unsupervised.

With the ArcGIS Spatial Analyst extension, there is a full suite of tools in the Multivariate toolset to perform supervised and unsupervised classification (see An overview of the Multivariate toolset). The classification process is a multi-step workflow, therefore, the Image Classification toolbar has been developed to provide an integrated environment to perform classifications with the tools. Not only does the toolbar help with the workflow for performing unsupervised and supervised classification, it also contains additional functionality for analyzing input data, creating training samples and signature files, and determining the quality of the training samples and signature files. The recommended way to perform classification and multivariate analysis is through the Image Classification toolbar.

Supervised Classification

With supervised classification, we identify examples of the Information classes (i.e., land cover type) of interest in the image. These are called "training sites". The image processing software system is then used to develop a statistical characterization of the reflectance for each information class. This stage is often called "signature analysis" and may involve developing a characterization as simple as the mean or the range of reflectance on each bands, or as complex as detailed analyses of the mean, variances and covariance over all bands. Once a statistical characterization has been achieved for each information class, the image is then classified by examining the reflectance for each pixel and making a decision about which of the signatures it resembles most. (Eastman,1995)

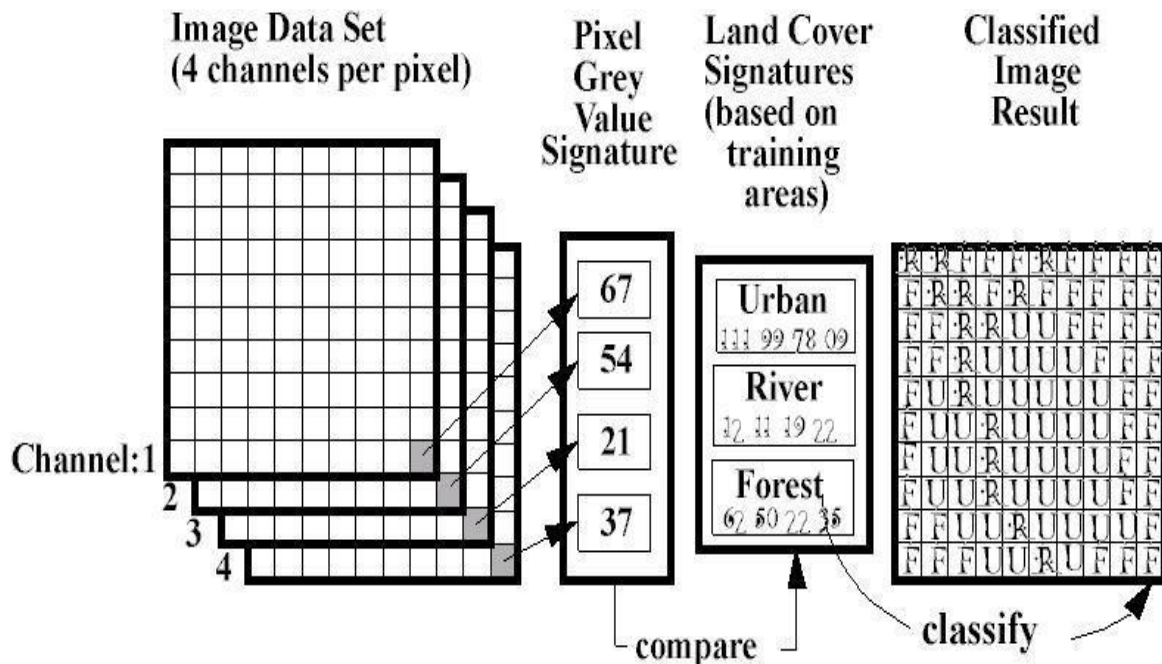


Figure Steps in Supervised classification

Maximum likelihood Classification

Maximum likelihood Classification is a statistical decision criterion to assist in the classification of overlapping signatures; pixels are assigned to the class of highest probability.

The maximum likelihood classifier is considered to give more accurate results than parallelepiped classification however it is much slower due to extra computations. We put the word 'accurate' in quotes because this assumes that classes in the input data have a Gaussian distribution and that signatures were well selected; this is not always a safe assumption.

Minimum distance Classification

Minimum distance classifies image data on a database file using a set of 256 possible class signature segments as specified by signature parameter. Each segment specified in signature, for example, stores signature data pertaining to a particular class. Only the mean vector in each class signature segment is used. Other data, such as standard deviations and covariance matrices, are ignored (though the maximum likelihood classifier uses this).

The result of the classification is a theme map directed to a specified database image channel. A theme map encodes each class with a unique gray level. The gray-level value used to encode a class is specified when the class signature is created. If the theme map is later transferred to the display, then a pseudo-color table should be loaded so that each class is represented by a different color.

Parallelepiped Classification

The parallelepiped classifier uses the class limits and stored in each class signature to determine if a given pixel falls within the class or not. The class limits specify the dimensions (in standard deviation units) of each side of a parallelepiped surrounding the mean of the class in feature space.

If the pixel falls inside the parallelepiped, it is assigned to the class. However, if the pixel falls within more than one class, it is put in the overlap class (code 255). If the pixel does not fall inside any class, it is assigned to the null class (code 0).

The parallelepiped classifier is typically used when speed is required. The drawback is (in many cases) poor accuracy and a large number of pixels classified as ties (or overlap, class 255).

Unsupervised Classification

Unsupervised classification is a method which examines a large number of unknown pixels and divides into a number of classes based on natural groupings present in the image values. Unlike supervised classification, unsupervised classification does not require analyst-specified training data. The basic premise is that values within a given cover type should be close together in the measurement space (i.e. have similar gray levels), whereas data in different classes should be comparatively well separated (i.e. have very different gray levels) (PCI, 1997; Lillesand and Kiefer, 1994; Eastman, 1995)

The classes that result from unsupervised classification are spectral classes which based on natural groupings of the image values, the identity of the spectral class will not be initially known, must compare classified data to some form of reference data (such as larger scale imagery, maps, or site visits) to determine the identity and informational values of the spectral classes. Thus, in the supervised approach, to define useful information categories and then examine their spectral separability; in the unsupervised approach the computer determines spectrally separable class, and then define their information value. (PCI, 1997; Lillesand and Kiefer, 1994)

Unsupervised classification is becoming increasingly popular in agencies involved in long term GIS database maintenance. The reason is that there are now systems that use clustering procedures that are extremely fast and require little in the nature of operational parameters. Thus it is becoming possible to train GIS analysis with only a general familiarity with remote sensing to undertake classifications that meet typical map

accuracy standards. With suitable ground truth accuracy assessment procedures, this tool can provide a remarkably rapid means of producing quality land cover data on a continuing basis.

TUBERCULOSIS

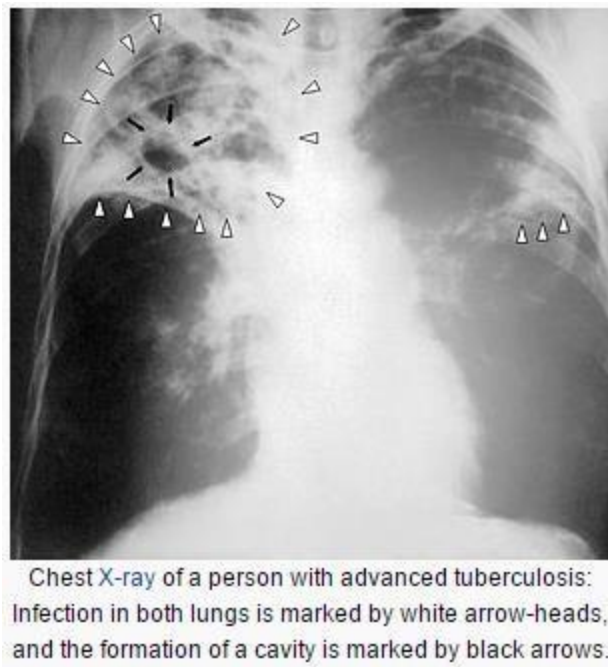
Tuberculosis (TB) is an infectious disease caused by the bacterium *Mycobacterium tuberculosis* (MTB). Tuberculosis generally affects the lungs, but can also affect other parts of the body. Most infections do not have symptoms, known as latent tuberculosis. About 10% of latent infections progress to active disease which, if left untreated, kills about half of those infected. The classic symptoms of active TB are a chronic cough with blood-containing sputum, fever, night sweats, and weight loss. The historical term "consumption" came about due to the weight loss. Infection of other organs can cause a wide range of symptoms.

Tuberculosis is spread through the air when people who have active TB in their lungs cough, spit, speak, or sneeze. People with latent TB do not spread the disease. Active infection occurs more often in people with HIV/AIDS and in those who smoke. Diagnosis of active TB is based on chest X-rays, as well as microscopic examination and culture of body fluids. Diagnosis of latent TB relies on the tuberculin skin test (TST) or blood tests.

Prevention of TB involves screening those at high risk, early detection and treatment of cases, and vaccination with the bacillus Calmette-Guérin vaccine. Those at high risk include household, workplace, and social contacts of people with active TB. Treatment requires the use of multiple antibiotics over a long period of time. Antibiotic resistance is a growing problem with increasing rates of multiple drug-resistant tuberculosis (MDR-TB).

One-third of the world's population is thought to be infected with TB. New infections occur in about 1% of the population each year. In 2014, there were 9.6 million cases of active TB which resulted in 1.5 million deaths. More than 95% of deaths

occurred in developing countries. The number of new cases each year has decreased since 2000. About 80% of people in many Asian and African countries test positive while 5– 10% of people in the United States population tests positive by the tuberculin test. Tuberculosis has been present in humans since ancient times.



Signs and symptoms

Tuberculosis may infect any part of the body, but most commonly occurs in the lungs (known as pulmonary tuberculosis). Extrapulmonary TB occurs when tuberculosis develops outside of the lungs, although extrapulmonary TB may coexist with pulmonary TB.

General signs and symptoms include fever, chills, night sweats, loss of appetite, weight loss, and fatigue. Significant nail clubbing may also occur.

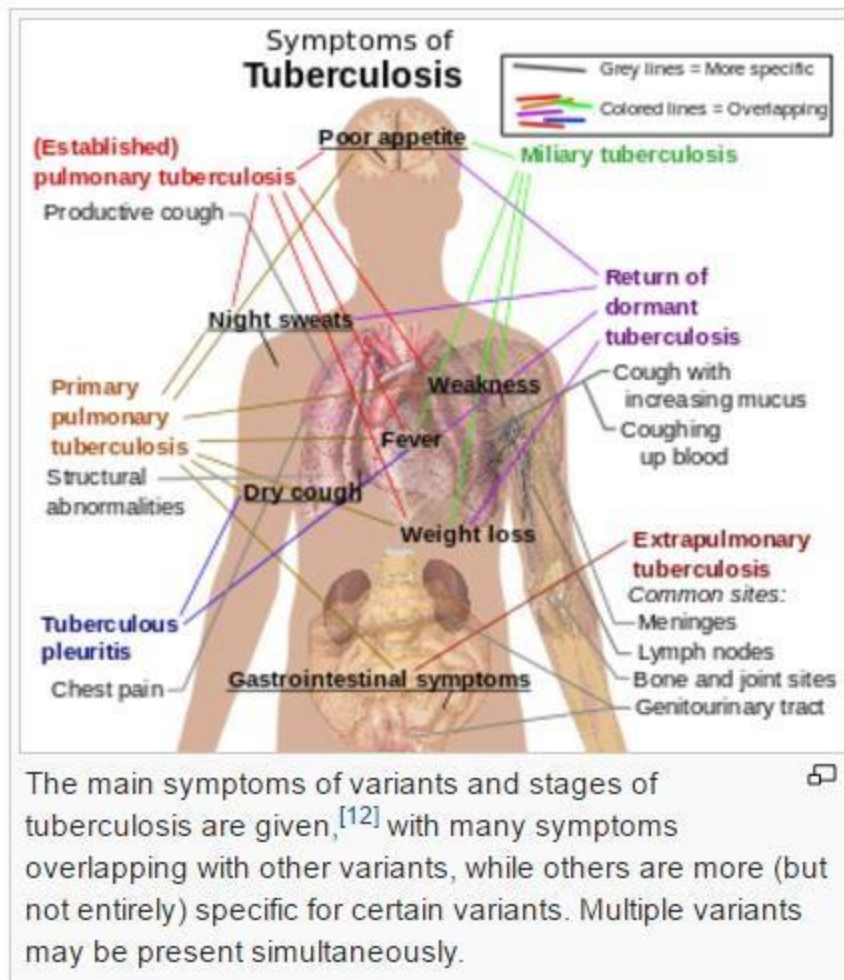
Pulmonary

If a tuberculosis infection does become active, it most commonly involves the lungs (in about 90% of cases). Symptoms may include chest pain and a prolonged cough producing sputum. About 25% of people may not have any symptoms (i.e. they remain

"asymptomatic"). Occasionally, people may cough up blood in small amounts, and in very rare cases, the infection may erode into the pulmonary artery or a Rasmussen's aneurysm, resulting in massive bleeding. Tuberculosis may become a chronic illness and cause extensive scarring in the upper lobes of the lungs. The upper lung lobes are more frequently affected by tuberculosis than the lower ones. The reason for this difference is not clear. It may be due to either better air flow, or poor lymph drainage within the upper lungs.

Extrapulmonary

In 15–20% of active cases, the infection spreads outside the lungs, causing other kinds of TB. These are collectively denoted as "extrapulmonary tuberculosis". Extrapulmonary TB occurs more commonly in immunosuppressed persons and young children. In those with HIV, this occurs in more than 50% of cases. Notable extrapulmonary infection sites include the pleura (in tuberculous pleurisy), the central nervous system (in tuberculous meningitis), the lymphatic system (in scrofula of the neck), the genitourinary system (in urogenital tuberculosis), and the bones and joints (in Pott disease of the spine), among others. When it spreads to the bones, it is also known as "osseous tuberculosis", a form of osteomyelitis. Sometimes, bursting of a tubercular abscess through skin results in tuberculous ulcer. An ulcer originating from nearby infected lymph nodes is painless, slowly enlarging and has an appearance of "wash leather". A potentially more serious, widespread form of TB is called "disseminated tuberculosis", also known as miliary tuberculosis. Miliary TB makes up about 10% of extrapulmonary cases.



CAUSES

Mycobacteria

The main cause of TB is *Mycobacterium tuberculosis*, a small, aerobic, nonmotile bacillus. The high lipid content of this pathogen accounts for many of its unique clinical characteristics. It divides every 16 to 20 hours, which is an extremely slow rate compared with other bacteria, which usually divide in less than an hour. *Mycobacteria* have an outer membrane lipid bilayer. If a Gram stain is performed, MTB either stains very weakly "Gram-positive" or does not retain dye as a result of the high lipid and mycolic acid content of its cell wall. MTB can withstand weak disinfectants and survive in a dry state for weeks. In nature, the bacterium can grow only within the cells of a host organism, but

M. tuberculosis can be cultured in the laboratory.

Using histological stains on expectorated samples from phlegm (also called "sputum"), scientists can identify MTB under a microscope. Since MTB retains certain stains even after being treated with acidic solution, it is classified as an acid-fast bacillus. The most common acid-fast staining techniques are the Ziehl–Neelsen stain and the Kinyoun stain, which dye acid-fast bacilli a bright red that stands out against a blue background. Auramine-rhodamine staining and fluorescence microscopy are also used.

The *M. tuberculosis* complex (MTBC) includes four other TB-causing mycobacteria: *M. bovis*, *M. africanum*, *M. canetti*, and *M. microti*. *M. africanum* is not widespread, but it is a significant cause of tuberculosis in parts of Africa. *M. bovis* was once a common cause of tuberculosis, but the introduction of pasteurized milk has almost completely eliminated this as a public health problem in developed countries. *M. canetti* is rare and seems to be limited to the Horn of Africa, although a few cases have been seen in African emigrants. *M. microti* is also rare and is seen almost only in immunodeficient people, although its prevalence may be significantly underestimated.

Other known pathogenic mycobacteria include *M. leprae*, *M. avium*, and *M. kansasii*. The latter two species are classified as "nontuberculous mycobacteria" (NTM). NTM cause neither TB nor leprosy, but they do cause pulmonary diseases that resemble TB.

Risk factors

A number of factors make people more susceptible to TB infections. The most important risk factor globally is HIV; 13% of all people with TB are infected by the virus. This is a particular problem in sub-Saharan Africa, where rates of HIV are high. Of people without HIV who are infected with tuberculosis, about 5–10% develop active disease during their lifetimes; in contrast, 30% of those coinfecting with HIV develop the active disease.

Tuberculosis is closely linked to both overcrowding and malnutrition, making it one of the principal diseases of poverty. Those at high risk thus include: people who

inject illicit drugs, inhabitants and employees of locales where vulnerable people gather (e.g. prisons and homeless shelters), medically underprivileged and resource-poor communities, high-risk ethnic minorities, children in close contact with high-risk category patients, and health-care providers serving these patients.

Chronic lung disease is another significant risk factor. Silicosis increases the risk about 30-fold. Those who smoke cigarettes have nearly twice the risk of TB compared to nonsmokers.

Other disease states can also increase the risk of developing tuberculosis. These include alcoholism and diabetes mellitus (three-fold increase).

Certain medications, such as corticosteroids and infliximab (an anti- α TNF monoclonal antibody), are becoming increasingly important risk factors, especially in the developed world.

Genetic susceptibility also exists, for which the overall importance remains undefined.

DIAGNOSIS

Active tuberculosis

Diagnosing active tuberculosis based only on signs and symptoms is difficult, as is diagnosing the disease in those who are immunosuppressed. A diagnosis of TB should, however, be considered in those with signs of lung disease or constitutional symptoms lasting longer than two weeks. A chest X-ray and multiple sputum cultures for acid-fast bacilli are typically part of the initial evaluation. Interferon- γ release assays and tuberculin skin tests are of little use in the developing world. IGRA have similar limitations in those with HIV.

A definitive diagnosis of TB is made by identifying *M. tuberculosis* in a clinical sample (e.g., sputum, pus, or a tissue biopsy). However, the difficult culture process for

this slow-growing organism can take two to six weeks for blood or sputum culture. Thus, treatment is often begun before cultures are confirmed.

Nucleic acid amplification tests and adenosine deaminase testing may allow rapid diagnosis of TB. These tests, however, are not routinely recommended, as they rarely alter how a person is treated. Blood tests to detect antibodies are not specific or sensitive, so they are not recommended.

Latent tuberculosis

The Mantoux tuberculin skin test is often used to screen people at high risk for TB. Those who have been previously immunized may have a false-positive test result. The test may be falsely negative in those with sarcoidosis, Hodgkin's lymphoma, malnutrition, and most notably, active tuberculosis. Interferon gamma release assays (IGRAs), on a blood sample, are recommended in those who are positive to the Mantoux test. These are not affected by immunization or most environmental mycobacteria, so they generate fewer false-positive results. However, they are affected by *M. szulgai*, *M. marinum*, and *M. kansasii*. IGRAs may increase sensitivity when used in addition to the skin test, but may be less sensitive than the skin test when used alone.

CHAPTER -2

LITERATURE SURVEY

In 2011, **Tao Xu, Irene Cheng, and Mrinal Mandal**, presented **“Automated Cavity Detection of Infectious Pulmonary Tuberculosis in Chest Radiographs”**. The presence of cavities in the upper lung zones is an important indicator of highly infectious Tuberculosis (TB). Diagnoses performed by the radiologists are labor intensive and of high inter-reader variation. After analyzing the existing computer-aided detection techniques, they propose an fully automated TB cavity detection system which combines a 2D Gaussian-model-based template matching (GTM) for candidates detection with Hessian-matrix-based image enhancement (HIE) for the following cavity segmentation and feature extraction. Experimental results demonstrate that our approach outperforms the existing TB cavity detection technique with higher accuracy and lower false positive rate.

In this paper, they propose a computer-aided system for TB cavity detection from chest radiographs. Without changing any information of the original image, a GTM is applied to find the cavity candidates. The HIE technique then improves the image quality for the following segmentation and feature extraction. Final classification results indicate that the proposed system has better detection performance than the previous technique. Furthermore, the detection will be much more accurate if it is combined with automatic lung field segmentation techniques.

In 2014, **Sema Candemir, Stefan Jaeger, Kannappan Palaniappan, Jonathan P. Musco, Rahul K. Singh, Zhiyun Xue, Alexandros Karargyris, Sameer Antani, George Thoma, and Clement J. McDonald** presented **“Lung Segmentation in Chest Radiographs Using Anatomical Atlases With Nonrigid Registration”**. The National Library of Medicine (NLM) is developing a digital chest X-ray (CXR) screening system for deployment in resource constrained communities and developing countries worldwide with a focus on early detection of tuberculosis. A critical component

in the computer-aided diagnosis of digital CXRs is the automatic detection of the lung regions. In this paper, they present a nonrigid registration-driven robust lung segmentation method using image retrieval-based patient specific adaptive lung models that detects lung boundaries, surpassing state-of-the-art performance. The method consists of three main stages: 1) a content-based image retrieval approach for identifying training images (with masks) most similar to the patient CXR using a partial Radon transform and Bhattacharyya shape similarity measure, 2) creating the initial patient-specific anatomical model of lung shape using SIFT-flow for deformable registration of training masks to the patient CXR, and 3) extracting refined lung boundaries using a graph cuts optimization approach with a customized energy function. Their average accuracy of 95.4% on the public JSRT database is the highest among published results. A similar degree of accuracy of 94.1% and 91.7% on two new CXR datasets from Montgomery County, MD, USA, and India, respectively, demonstrates the robustness of our lung segmentation approach.

They have presented a robust lung boundary detection method that is based on a patient-specific lung atlas using fast partial Radon profile similarity selection and SIFT-flow nonrigid registration with refinement using a graph cuts segmentation algorithm. We evaluated the algorithm using three different datasets containing 585 chest radiographs from patients with normal lungs and various pulmonary diseases. They are making our CXR datasets from Montgomery County, MD, USA and India publicly available to the research community. This includes the X-ray images as well as associated metadata and ground-truth lung segmentations. These two new CXR sets provide for the first time a publicly available research collection of CXRs with manifestations of tuberculosis. The X-rays cover different regions of the world, including high incidence regions. The ground truth segmentations are provided by a board certified radiologist, and represents the first publicly available collection to include abnormal lung shapes. These sets will enable new research opportunities, and they will improve the existing benchmark evaluations.

In 2014, **Wai Yan NyeinNaing, ZawZ. Hti** presented “**Advances in automatic tuberculosis Detection In chest x-Ray images**”. Tuberculosis (TB) is very dangerous and rapidly spread disease in the world. In the investigating cases for suspected tuberculosis (TB), chest radiography is not only the key techniques of diagnosis based on the medical imaging but also the diagnostic radiology. So, Computer aided diagnosis (CAD) has been popular and many researchers are interested in this research areas and different approaches have been proposed for the TB detection and lung disease classification. In this paper, the medical background history of TB disease in chest X-rays and a survey of the various approaches in TB detection and classification are presented. The literature in the related methods is surveyed papers in this research area until now 2014.

This paper is surveyed with the analysis of a CAD system for automated analysis of chest x-ray for identification of pulmonary TB. This paper emphasizes to give the steps of TB classification from X-ray images such as preprocessing, feature extraction and classification. Most of researches are going on in this area from several years. Table 1 shows the summary of the review. This paper is based on all these researches and experiments for detecting the possibility of TB in a chest radiograph. This will substantially reduce the effort of Medical officer and radiologist.

In 2015, **Hrudya Das** and **Ajay Nath** presented, “**An Efficient Detection of Tuberculosis from Chest X-rays**”. Tuberculosis (TB) is a major health problem in all over the world. Chest radiography is becoming an important tool for fighting against TB. Existing Methods are less reliable in high population. So a computer aided system for detecting TB is becoming more needful for the mass screening of TB. Detecting cavities from chest x-ray is an efficient method for diagnosing the TB. So here, an automatic method is explained for detecting the TB from CXR with less effort. Region based active contour segmentation is used for segmenting lung field and the extracted features are classified using supported vector machine as normal and abnormal. The Montgomery County (MC) Data set contains 138 posterior anterior cxrs, among which 80

cxrs are normal and 58 cxrs are abnormal with manifestations of TB are used. All images of the MC set are in 12-bit grayscale, captured with an Eureka stationary x-ray machine (CR). The abnormal cxrs cover a wide range of TB-related abnormalities, including effusions and miliary patterns. the full system explanation is discussed.

TB can be detected from Chest x-ray images by using image processing methods like segmentation, Feature Extraction and classification. Existing diagnostics method such as sputum staining has become less reliable in high population so this method will be helpful in rural areas. When increasing the features selected and when using another segmentation method it may get more accurate result.

In 2013, **Stefan Jaeger, Alexandros Karargyris, Sema Candemir, Les Folio, Jenifer Siegelman, Fiona Callaghan, Zhiyun Xue, Kannappan Palaniappan, Rahul K. Singh, Sameer Antani, George Thoma, Yi-Xiang Wang, Pu-Xuan Lu, and Clement J. McDonald** presented, **“Automatic Tuberculosis Screening Using Chest Radiographs”**. Tuberculosis is a major health threat in many regions of the world. Opportunistic infections in immunocompromised HIV/AIDS patients and multi-drug-resistant bacterial strains have exacerbated the problem, while diagnosing tuberculosis still remains a challenge. When left undiagnosed and thus untreated, mortality rates of patients with tuberculosis are high. Standard diagnostics still rely on methods developed in the last century. They are slow and often unreliable. In an effort to reduce the burden of the disease, this paper presents our automated approach for detecting tuberculosis in conventional posterior anterior chest radiographs. They first extract the lung region using a graph cut segmentation method. For this lung region, they compute a set of texture and shape features, which enable the x-rays to be classified as normal or abnormal using a binary classifier. They measure the performance of our system on two datasets: a set collected by the tuberculosis control program of our local county's health department in the United States, and a set collected by Shenzhen Hospital, China. The proposed computer-aided diagnostic system for TB screening, which is ready for field deployment, achieves a performance that approaches the performance of human experts. they achieve

an area under the ROC curve (AUC) of 87% (78.3% accuracy) for the first set, and an AUC of 90% (84% accuracy) for the second set. For the first set, they compare their system performance with the performance of radiologists. When trying not to miss any positive cases, radiologists achieve an accuracy of about 82% on this set, and their false positive rate is about half of our system's rate.

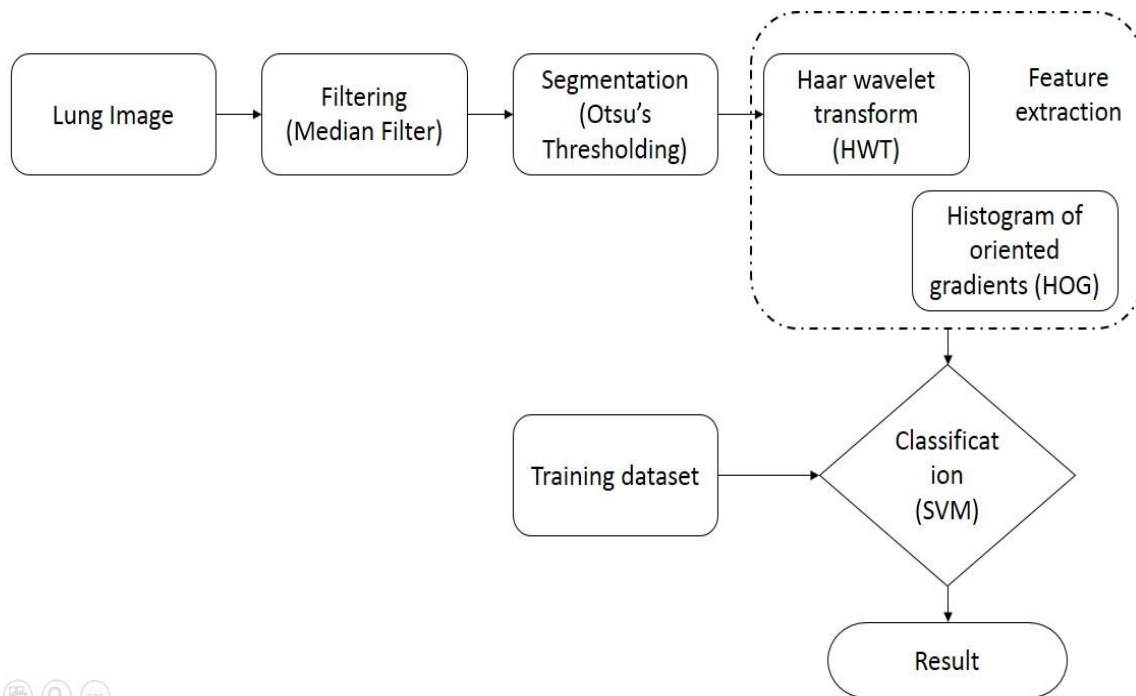
They have developed an automated system that screens CXRs for manifestations of TB. The system is currently set up for practical use in Kenya, where it will be part of a mobile system for TB screening in remote areas. When given a CXR as input, our system first segments the lung region using an optimization method based on graph cut. This method combines intensity information with personalized lung atlas models derived from the training set. We compute a set of shape, edge, and texture features as input to a binary classifier, which then classifies a given input image into either normal or abnormal. In this paper, we compare two different established feature sets: one set typically used for object recognition and the other used in image retrieval applications. They also experiment with different classifier architectures. Both feature sets and most of the classifier architectures we tested, provide a similar performance. To improve the performance further, we could try to improve the lung segmentation, which provides average performance compared to other systems in the literature. One approach would be to find optimal weights for the terms in the graph cut energy function. they could also try to partition the lung into different regions, as some of the existing CAD systems do. It is surprising that we achieve a relatively high performance compared to other approaches by using only global features. A final verdict on this issue can only be made once public benchmark data becomes available. Furthermore, our performance, while still lower than human performance, is reasonably close to the performance of radiologists. In an independent observer study with two radiologists, who were trying not to miss any positive case, our false positive rate is twice as high according to the ROC curve. The likelihood that both the radiologists and the machine reach a wrong conclusion is very low. This shows that it should be possible to reach human performance in the future, or at

least have a system that can assist radiologists and public health providers in the screening and decision process. These comparison results have encouraged us to test our system in the field under realistic conditions. In future experiments, we will evaluate our system on larger datasets that we will collect using our portable scanners in Kenya.

CHAPTER-3

THEME OF THE PROJECT

3.1. FLOWDIAGRAM



. FLOW DIAGRAM DESCRIPTION

First the input image selected for identification of brain tuberculosis affect in lung images. The selected image may have some noise so the image filtered by using median filter which removes the salt and pepper noise in the image. The filtered image then segmented using Otsu's thresholding algorithm which segments the lung part of the X-ray

image. Then the features of the image (i.e., mean, variance, entropy, standard deviation, etc.) are extracted by using the Haar wavelet transform (HWT) and Histogram of oriented gradients (HOG). Finally the features of the input image are compared with the trained dataset using support vector machine. The trained dataset consists of features of normal and affected lung images. Finally the result will be shown whether tuberculosis was affected or not.

BLOCK EXPLANATION

MEDIANFILTER

The median filter is a nonlinear digital filtering technique, often used to remove noise. Such noise reduction is a typical pre-processing step to improve the results of later processing (for example, edge detection on an image). Median filtering is very widely used in digital image processing because, under certain conditions, it preserves edges while removing noise (but see discussion below).

Algorithm description

The main idea of the median filter is to run through the signal entry by entry, replacing each entry with the median of neighboring entries. The pattern of neighbors is called the "window", which slides, entry by entry, over the entire signal. For 1D signals, the most obvious window is just the first few preceding and following entries, whereas for 2D (or higher-dimensional) signals such as images, more complex window patterns are possible (such as "box" or "cross" patterns). Note that if the window has an odd number of entries, then the median is simple to define: it is just the middle value after all the entries in the window are sorted numerically. For an even number of entries, there is more than one possible median, see median for more details.

Worked 1D example

To demonstrate, using a window size of three with one entry immediately preceding and following each entry, a median filter will be applied to the following simple 1D signal:

$x = [2 \ 80 \ 6 \ 3]$

So, the median filtered output signal y will be:

$y[1] = \text{Median}[2 \ 2 \ 80] = 2$

$y[2] = \text{Median}[2 \ 80 \ 6] = \text{Median}[2 \ 6 \ 80] = 6$

$y[3] = \text{Median}[80 \ 6 \ 3] = \text{Median}[3 \ 6 \ 80] = 6$

$y[4] = \text{Median}[6 \ 3 \ 3] = \text{Median}[3 \ 3 \ 6] = 3$

i.e. $y = [2 \ 6 \ 6 \ 3]$.

Boundary issues

Note that, in the example above, because there is no entry preceding the first value, the first value is repeated, as with the last value, to obtain enough entries to fill the window. This is one way of handling missing window entries at the boundaries of the signal, but there are other schemes that have different properties that might be preferred in particular circumstances:

- Avoid processing the boundaries, with or without cropping the signal or image boundary afterwards,
- Fetching entries from other places in the signal. With images for example, entries from the far horizontal or vertical boundary might be selected,
- Shrinking the window near the boundaries, so that every window is full.

2D median filter pseudo code

Code for a simple 2D median filter algorithm might look like this:

```

allocate outputPixelValue[image width][image height]
allocate window>window width * window height edgex
:= (window width / 2) rounded down
edgey := (window height / 2) rounded down
for x from edgex to image width - edgex
  for y from edgey to image height -
    edgeyi = 0
    for fx from 0 to window width for
      fy from 0 to windowheight
window[i] := inputPixelValue[x + fx - edgex][y + fy - edgey] i
:= i + 1
sort entries in window[]
outputPixelValue[x][y] := window>window width * window height / 2]

```

Note that this algorithm:

- Processes one color channel only,
- Takes the "not processing boundaries" approach (see above discussion about boundary issues).

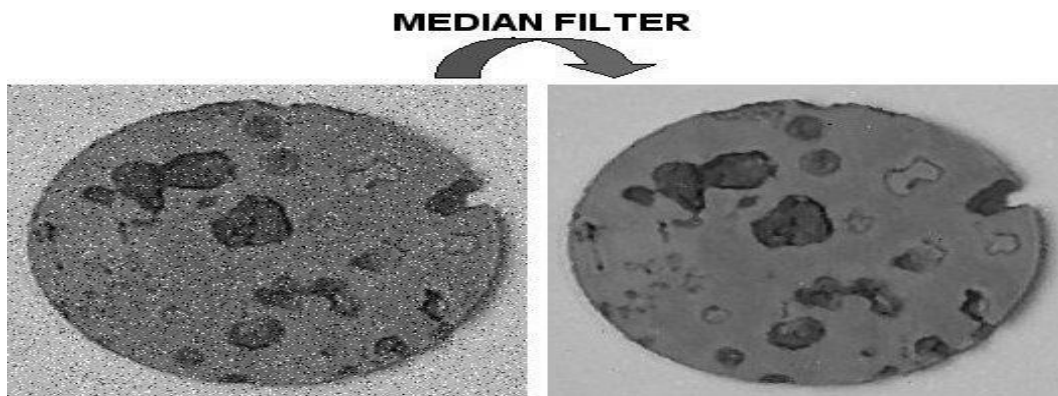
Algorithm implementation issues

Typically, by far the majority of the computational effort and time is spent on calculating the median of each window. Because the filter must process every entry in the signal, for large signals such as images, the efficiency of this median calculation is a critical factor in determining how fast the algorithm can run. The "vanilla"

implementation described above sorts every entry in the window to find the median; however, since only the middle value in a list of numbers is required, selection algorithms can be much more efficient. Furthermore, some types of signals (very often the case for images) use whole number representations: in these cases, histogram medians can be far more efficient because it is simple to update the histogram from window to window, and finding the median of a histogram is not particularly onerous.

Edge preservation properties

Median filtering is one kind of smoothing technique, as is linear Gaussian filtering. All smoothing techniques are effective at removing noise in smooth patches or smooth regions of a signal, but adversely affect edges. Often though, at the same time as reducing the noise in a signal, it is important to preserve the edges. Edges are of critical importance to the visual appearance of images, for example. For small to moderate levels of (Gaussian) noise, the median filter is demonstrably better than Gaussian blur at removing noise whilst preserving edges for a given, fixed window size. However, its performance is not that much better than Gaussian blur for high levels of noise, whereas, for speckle noise and salt and pepper noise (impulsive noise), it is particularly effective. Because of this, median filtering is very widely used in digital imageprocessing.



OTSU'S THRESHOLDING

In computer vision and image processing, Otsu's method, named after Nobuyuki Otsu (大津展之Ôtsu Nobuyuki?), is used to automatically perform clustering-based

image thresholding or the reduction of a graylevel image to a binary image. The algorithm assumes that the image contains two classes of pixels following bi-modal histogram (foreground pixels and background pixels), it then calculates the optimum threshold separating the two classes so that their combined spread (intra-class variance) is minimal, or equivalently (because the sum of pairwise squared distances is constant), so that their inter-class variance is maximal. Consequently, Otsu's method is roughly a one- dimensional, discrete analog of Fisher's DiscriminantAnalysis.

The extension of the original method to multi-level thresholding is referred to as the multi Otsu method.

Method

In Otsu's method we exhaustively search for the threshold that minimizes the intra- class variance (the variance within the class), defined as a weighted sum of variances of the two classes:

$$\sigma_w^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t)$$

Weights $\omega_{\{0,1\}}$ are the probabilities of the two classes separated by a threshold t and $\sigma_{\{0,1\}}^2$ are variances of these two classes.

The class probability $\omega_{\{0,1\}}(t)$ is computed from the L histograms:

$$\omega_0(t) = \sum_{i=0}^{t-1} p(i)$$

$$\omega_1(t) = \sum_{i=t}^{L-1} p(i)$$

Otsu shows that minimizing the intra-class variance is the same as maximizing inter-class variance:

$$\begin{aligned}\sigma_b^2(t) &= \sigma^2 - \sigma_w^2(t) = \omega_0(\mu_0 - \mu_T)^2 + \omega_1(\mu_1 - \mu_T)^2 \\ &= \omega_0(t)\omega_1(t)[\mu_0(t) - \mu_1(t)]^2\end{aligned}$$

which is expressed in terms of class probabilities ω and class means μ . while the class mean $\mu_{0,1,T}(t)$ is:

$$\begin{aligned}\mu_0(t) &= \sum_{i=0}^{t-1} i \frac{p(i)}{\omega_0} \\ \mu_1(t) &= \sum_{i=t}^{L-1} i \frac{p(i)}{\omega_1} \\ \mu_T &= \sum_{i=0}^{L-1} i p(i)\end{aligned}$$

The following relations can be easily verified:

$$\begin{aligned}\omega_0 \mu_0 + \omega_1 \mu_1 &= \mu_T \\ \omega_0 + \omega_1 &= 1\end{aligned}$$

The class probabilities and class means can be computed iteratively. This idea yields an effective algorithm.

Algorithm

1. Compute histogram and probabilities of each intensity level
2. Set up initial $\omega_i(0)$ and $\mu_i(0)$
3. Step through all possible thresholds $t=1, \dots$ maximum intensity
 - Update ω_i and μ_i
 - Compute $\sigma_b^2(t)$
4. Desired threshold corresponds to the maximum $\sigma_b^2(t)$

Limitations

Otsu's method exhibits the relatively good performance if the histogram can be assumed to have bimodal distribution and assumed to possess a deep and sharp valley between two peaks. But if the object area is small compared with the background area, the histogram no longer exhibits bimodality. And if the variances of the object and the background intensities are large compared to the mean difference, or the image is severely corrupted by additive noise, the sharp valley of the gray level histogram is degraded. Then the possibly incorrect threshold determined by Otsu's method results in the segmentation error. (Here we define the object size to be the ratio of the object area to the entire image area and the mean difference to be the difference of the average intensities of the object and the background)

From the experimental results, the performance of global thresholding techniques including Otsu's method is shown to be limited by the small object size, the small mean difference, the large variances of the object and the background intensities, the large amount of noise added, and so on.

HAAR WAVELET TRANSFORMATION

Introduction

The easiest of all discrete wavelet transformations is the Discrete Haar Wavelet Transformation (HWT). Let's motivate its construction with the following example:

Suppose you had the eight numbers 100, 200, 44, 50, 20, 20, 4, 2 (these could be grayscale intensities) and you wanted to send an approximation of the list to a friend. Due to bandwidth constraints (this is a really old system!), you are only allowed to send your friend four values. What four values would you send your friend that might represent an approximation of the eight given values?

There are obviously many possible answers to this question, but one of the most common solutions is to take the eight numbers, two at a time, and average them. This computation would produce the four values 150, 47, 20, and 3. This list would represent an approximation to the original eight values.

Unfortunately, if your friend receives the four values 150, 47, 20, and 3, she has no chance of producing the original eight values from them - more information is needed. Suppose you are allowed to send an additional four values to your friend. With these values and the first four values, she should be able to reconstruct your original list of eight values. What values would you send her?

Suppose we sent our friend the values 50, 3, 0, and -1. How did we arrive at these values? They are simply the directed distances from the pairwise average to the second number in each pair: $150 + 50 = 200$, $47 + 3 = 50$, $20 + 0 = 20$, and $3 + (-1) = 2$. Note that if we subtract the values in this list from the pairwise averages, we arrive at the first number in each pair: $150 - 50 = 100$, $47 - 3 = 44$, $20 - 0 = 20$, and $3 - (-1) = 4$. So with the lists (150,47,20,3) and (50,3,0,-1), we can completely reconstruct the original list (100,200,44,50,20,20,4,2).

Given two numbers a and b , we have the following transformation:

$$(a, b) \rightarrow ((b + a)/2, (b - a)/2)$$

We will call the first output the average and the second output the difference.

So why would we consider sending (150,47,20,3 | 50, 3, 0, -1) instead of (100,200,44,50,20,20,4,2)? Two reasons quickly come to mind. The differences in the transformed list tell us about the trends in the data - big differences indicate large jumps between values while small values tell us that there is relatively little change in that portion of the input. Also, if we are interested in lossy compression, then small differences can be converted to zero and in this way we can improve the efficiency of the coder. Suppose we converted the last three values of the transformation to zero. Then we would transmit (150, 47, 20, 3 | 50, 0, 0, 0). The recipient could invert the process and obtain the list

$$(150-50, 150+50, 47-0, 47+0, 20-0, 20+0, 3-0, 3+0) = (100,200,47,47,20,20,3,3)$$

The "compressed" list is very similar to the original list!

Matrix Formulation

For an even-length list (vector) of numbers, we can also form a matrix product that computes this transformation. For the sake of illustration, let's assume our list (vector) is length 8. If we put the averages as the first half of the output and differences as the second half of the output, then we have the following matrix product:

$$\tilde{W}_8 \mathbf{v} = \begin{bmatrix} 1/2 & 1/2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 1/2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ -1/2 & 1/2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1/2 & 1/2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1/2 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1/2 & 1/2 \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ v_8 \end{bmatrix} = \begin{bmatrix} (v_1 + v_2)/2 \\ (v_3 + v_4)/2 \\ (v_5 + v_6)/2 \\ (v_7 + v_8)/2 \\ (v_2 - v_1)/2 \\ (v_4 - v_3)/2 \\ (v_6 - v_5)/2 \\ (v_8 - v_7)/2 \end{bmatrix} = \mathbf{y}$$

Inverting the Process

Inverting is easy - if we subtract y_5 from y_1 , we obtain v_1 . If we add y_5 and y_1 , we obtain v_2 . We can continue in a similar manner adding and subtracting pairs to completely recover $\mathbf{v}$. We can also write the inverse process as a matrix product. We have:

$$\tilde{W}_8^{-1} \mathbf{y} = \begin{bmatrix} 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} (v_1 + v_2)/2 \\ (v_3 + v_4)/2 \\ (v_5 + v_6)/2 \\ (v_7 + v_8)/2 \\ (v_2 - v_1)/2 \\ (v_4 - v_3)/2 \\ (v_6 - v_5)/2 \\ (v_8 - v_7)/2 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ v_8 \end{bmatrix} = \mathbf{v}$$

The matrix W_8 satisfies another interesting property - we can compute the inverse by doubling the transpose! That is,

$$\tilde{W}_8^{-1} = 2\tilde{W}_8^T$$

For those of you who have taken a linear algebra course, you may remember that orthogonal matrices U satisfy $U^{-1} = U^T$. We almost have that with our transformation. Indeed if we construct $W_8 = 2\tilde{W}_8$, we have

$$\begin{aligned} W_8^{-1} &= (\sqrt{2}\tilde{W}_8)^{-1} \\ &= (1/\sqrt{2})\tilde{W}_8^{-1} \\ &= (1/\sqrt{2})2\tilde{W}_8^T \\ &= \sqrt{2}\tilde{W}_8^T \\ &= (\sqrt{2}\tilde{W}_8^T)^T \\ &= W_8^T \end{aligned}$$

HWT and Digital Images

How do we apply the HWT to a digital grayscale image? If the image is stored in matrix A with even dimensions $M \times N$, then the natural thing to try is to compute WMA . We can view this matrix multiplication as WM applied to each column of A so the output should be an $M \times N$ matrix where each column is $M/2$ weighted averages followed by $M/2$ weighted differences. The plots below illustrate the process:



A digital image. [Fullsize version](#)



$W_{160}A$. [Fullsize version](#)

We have used the Haar matrix to process the columns of image matrix A . It is desirable to process the rows of the image as well. We proceed by multiplying WMA on the right by WNT. Transposing the wavelet matrix puts the filter coefficients in the columns and multiplication on the right by WNT means that we will be dotting the rows of WMA with the columns of WNT (columns of W_N). So the two dimensional HWT is defined as:

$$B = W_M A W_N^T$$

The process is illustrated below.



The two-dimensional HWT. [Fullsize version](#)

Analysis of the Two-Dimensional HWT

You can see why the wavelet transformation is well-suited for image compression. The two-dimensional HWT of the image has most of the energy conserved in the upper left-hand corner of the transform - the remaining three-quarters of the HWT consists primarily of values that are zero or near zero. The transformation is local as well - it turns out any element of the HWT is constructed from only four elements of the original input image. If we look at the HWT as a block matrix product, we can gain further insight about the transformation.

Suppose that the input image is square so we will drop the subscripts that indicate the dimension of the HWT matrix. If we use H to denote the top block of the HWT matrix and G to denote the bottom block of the HWT, we can express the transformation as:

$$B = WAW^T = \begin{bmatrix} H \\ G \end{bmatrix} A \begin{bmatrix} H \\ G \end{bmatrix}^T = \begin{bmatrix} H \\ G \end{bmatrix} A \begin{bmatrix} H^T \\ G^T \end{bmatrix} = \begin{bmatrix} HA \\ GA \end{bmatrix} \begin{bmatrix} H^T \\ G^T \end{bmatrix} = \begin{bmatrix} HAH^T & HAG^T \\ GAH^T & GAG^T \end{bmatrix}$$

We now see why there are four blocks in the wavelet transform. Let's look at each block individually. Note that the matrix H is constructed from the lowpass Haar filter and computes weighted averages while G computes weighted differences.

The upper left-hand block is HAH^T - HA averages columns of A and the rows of this product are averaged by multiplication with H^T . Thus the upper left-hand corner is an approximation of the entire image. In fact, it can be shown that elements in the upper left-hand corner of the HWT can be constructed by computing weighted averages of each 2 x 2 block of the input matrix. Mathematically, the mapping is

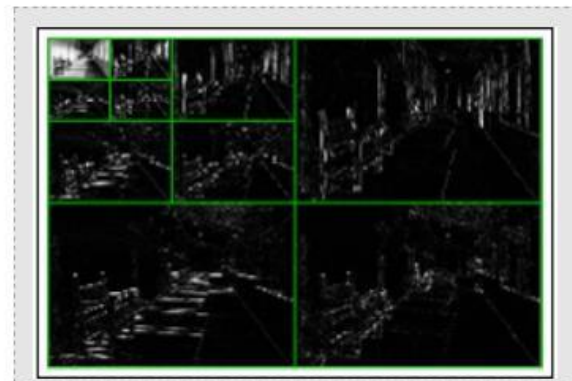
$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \rightarrow 2 \cdot (a + b + c + d) / 4$$

Iterating the Process

If there is not much change in the image, the difference blocks are comprised of (near) zero values. If we apply quantization and convert near-zero values to zero, then the HWT of the image can be effectively coded and the storage space for the image can be drastically reduced. We can iterate the HWT and produce an even better result to pass to the coder. Suppose we compute the HWT of a digital image. Most of the high intensities are contained in the blur portion of the transformation. We can iterate and apply the HWT to the blur portion of the transform. So in the composite transformation, we replace the blur by its transformation! The process is completely invertible - we apply the inverse HWT to the transform of the blur to obtain the blur. Then we apply the inverse HWT to obtain the original image. We can continue this process as often as we desire (and provided the dimensions of the data are divisible by suitable powers of two). The illustrations below show two iterations and three iterations of the HWT.



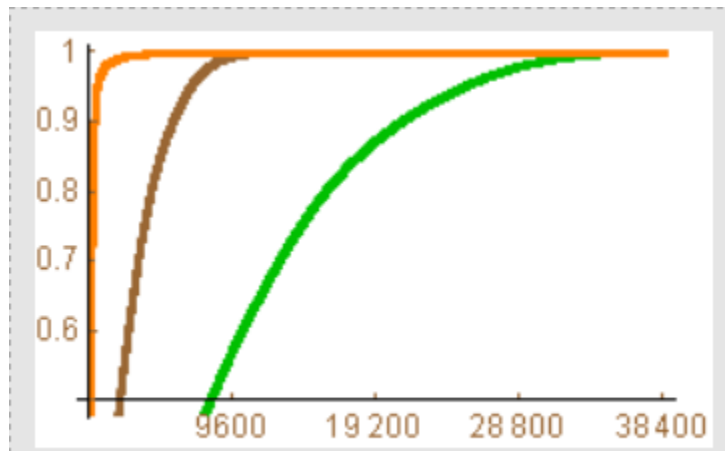
Two iterations of the HWT. [Fullsize version](#)



Three iterations of the HWT. [Fullsize version](#)

The iterated HWT is an effective tool for conserving the energy of a digital image. The plot below shows the energy distribution for the original image (green), one iteration of the HWT (brown), and three iterations of the HWT (orange). The horizontal scale is pixels (there are 38,400 pixels in the thumbnail of the image). For a given pixel value p , the height represents the percentage of energy stored in the largest p pixels of the image.

Note that the HWT gets to 1 (100% of the energy) much faster than the original image and the iterated HWT is much better than either the HWT or the original image.



Energy distribution for the image and HWTs.

Summary

The HWT is a wonderful tool for understanding how a discrete wavelet transformation works. It is not desirable in practice because the filters are too short - since each filter is length two, the HWT decouples the data to create values of the transform. In particular, each value of the transform is created from a 2 x 2 block from the original input. If there is a large change between say row 6 and row 7, the HWT will not detect it. The HWT also send integers to irrational numbers and for lossless image compression, it is crucial that the transform send integers to integers. For these reasons, researchers developed more sophisticated filters. Be sure to check out the other subsections to learn more other types of wavelet filters.

HISTOGRAM OF ORIENTED GRADIENTS

The histogram of oriented gradients (HOG) is a feature descriptor used in computer vision and image processing for the purpose of object detection. The technique counts occurrences of gradient orientation in localized portions of an image. This method is similar to that of edge orientation histograms, scale-invariant feature transform

descriptors, and shape contexts, but differs in that it is computed on a dense grid of uniformly spaced cells and uses overlapping local contrast normalization for improved accuracy.

Navneet Dalal and Bill Triggs, researchers for the French National Institute for Research in Computer Science and Automation (INRIA), first described HOG descriptors at the 2005 Conference on Computer Vision and Pattern Recognition (CVPR). In this work they focused on pedestrian detection in static images, although since then they expanded their tests to include human detection in videos, as well as to a variety of common animals and vehicles in static imagery.

Theory

The essential thought behind the histogram of oriented gradients descriptor is that local object appearance and shape within an image can be described by the distribution of intensity gradients or edge directions. The image is divided into small connected regions called cells, and for the pixels within each cell, a histogram of gradient directions is compiled. The descriptor is the concatenation of these histograms. For improved accuracy, the local histograms can be contrast-normalized by calculating a measure of the intensity across a larger region of the image, called a block, and then using this value to normalize all cells within the block. This normalization results in better invariance to changes in illumination and shadowing.

The HOG descriptor has a few key advantages over other descriptors. Since it operates on local cells, it is invariant to geometric and photometric transformations, except for object orientation. Such changes would only appear in larger spatial regions. Moreover, as Dalal and Triggs discovered, coarse spatial sampling, fine orientation sampling, and strong local photometric normalization permits the individual body movement of pedestrians to be ignored so long as they maintain a roughly upright position. The HOG descriptor is thus particularly suited for human detection in images.

Algorithm implementation

Gradient computation

The first step of calculation in many feature detectors in image pre-processing is to ensure normalized color and gamma values. As Dalal and Triggs point out, however, this step can be omitted in HOG descriptor computation, as the ensuing descriptor normalization essentially achieves the same result. Image pre-processing thus provides little impact on performance. Instead, the first step of calculation is the computation of the gradient values. The most common method is to apply the 1-D centered, point discrete derivative mask in one or both of the horizontal and vertical directions. Specifically, this method requires filtering the color or intensity data of the image with the following filterkernels:

$$[-1, 0, 1] \text{ and } [-1, 0, 1]^T.$$

Dalal and Triggs tested other, more complex masks, such as the 3x3 Sobel mask or diagonal masks, but these masks generally performed more poorly in detecting humans in images. They also experimented with Gaussian smoothing before applying the derivative mask, but similarly found that omission of any smoothing performed better in practice.

Orientation binning

The second step of calculation is creating the cell histograms. Each pixel within the cell casts a weighted vote for an orientation-based histogram channel based on the values found in the gradient computation. The cells themselves can either be rectangular or radial in shape, and the histogram channels are evenly spread over 0 to 180 degrees or 0 to 360 degrees, depending on whether the gradient is “unsigned” or “signed”. Dalal and Triggs found that unsigned gradients used in conjunction with 9 histogram channels performed best in their human detection experiments. As for the vote weight, pixel contribution can either be the gradient magnitude itself, or some function of the magnitude. In tests, the gradient magnitude itself generally produces the best results.

Other options for the vote weight could include the square root or square of the gradient magnitude, or some clipped version of the magnitude.

Descriptor blocks

To account for changes in illumination and contrast, the gradient strengths must be locally normalized, which requires grouping the cells together into larger, spatially connected blocks. The HOG descriptor is then the concatenated vector of the components of the normalized cell histograms from all of the block regions. These blocks typically overlap, meaning that each cell contributes more than once to the final descriptor. Two main block geometries exist: rectangular R-HOG blocks and circular C-HOG blocks. R-HOG blocks are generally square grids, represented by three parameters: the number of cells per block, the number of pixels per cell, and the number of channels per cell histogram. In the Dalal and Triggs human detection experiment, the optimal parameters were found to be four 8x8 pixels cells per block (16x16 pixels per block) with 9 histogram channels. Moreover, they found that some minor improvement in performance could be gained by applying a Gaussian spatial window within each block before tabulating histogram votes in order to weight pixels around the edge of the blocks less. The R-HOG blocks appear quite similar to the scale-invariant feature transform (SIFT) descriptors; however, despite their similar formation, R-HOG blocks are computed in dense grids at some single scale without orientation alignment, whereas SIFT descriptors are usually computed at sparse, scale-invariant key image points and are rotated to align orientation. In addition, the R-HOG blocks are used in conjunction to encode spatial form information, while SIFT descriptors are used singly.

Circular HOG blocks (C-HOG) can be found in two variants: those with a single, central cell and those with an angularly divided central cell. In addition, these C-HOG blocks can be described with four parameters: the number of angular and radial bins, the radius of the center bin, and the expansion factor for the radius of additional radial bins. Dalal and Triggs found that the two main variants provided equal performance, and that two radial bins with four angular bins, a center radius of 4 pixels, and an expansion factor

of 2 provided the best performance in their experimentation. Also, Gaussian weighting provided no benefit when used in conjunction with the C-HOG blocks. C-HOG blocks appear similar to shape context descriptors, but differ strongly in that C-HOG blocks contain cells with several orientation channels, while shape contexts only make use of a single edge presence count in their formulation.

Block normalization

Dalal and Triggs explored four different methods for block normalization. Let v be the non-normalized vector containing all histograms in a given block, $\|v\|_k$ be its k -norm for $k = 1, 2$ and e be some small constant (the exact value, hopefully, is unimportant). Then the normalization factor can be one of the following:

$$\text{L2-norm: } f = \frac{v}{\sqrt{\|v\|_2^2 + e^2}}$$

L2-hys: L2-norm followed by clipping (limiting the maximum values of v to 0.2) and renormalizing, as in [5]

$$\text{L1-norm: } f = \frac{v}{(\|v\|_1 + e)}$$

$$\text{L1-sqrt: } f = \sqrt{\frac{v}{(\|v\|_1 + e)}}$$

In addition, the scheme L2-hys can be computed by first taking the L2-norm, clipping the result, and then renormalizing. In their experiments, Dalal and Triggs found the L2-hys, L2-norm, and L1-sqrt schemes provide similar performance, while the L1-norm provides slightly less reliable performance; however, all four methods showed very significant improvement over the non-normalized data.

SVM classifier

The final step in object recognition using histogram of oriented Gradient descriptors is to feed the descriptors into some recognition system based on supervised learning. The support vector machine (SVM) classifier is a binary classifier which looks for an optimal hyperplane as a decision function. Once trained on images containing

some particular object, the SVM classifier can make decisions regarding the presence of an object, such as a human, in additional test images.

Neural Network Classifier

The feature of the gradient descriptors are also fed into the neural network classifiers which provides more accuracy in the classification comparing other classifiers (SVM). The neural classifiers can accept the descriptor feature as the binary function or the optimal function.

Performance

In their original human detection experiment, Dalal and Triggs compared their R-HOG and C-HOG descriptor blocks against generalized Haar wavelets, PCA-SIFT descriptors, and shape context descriptors. Generalized Haar wavelets are oriented Haar wavelets, and were used in 2001 by Mohan, Papageorgiou, and Poggio in their own object detection experiments. PCA-SIFT descriptors are similar to SIFT descriptors, but differ in that principal component analysis is applied to the normalized gradient patches. PCA-SIFT descriptors were first used in 2004 by Ke and Sukthankar and were claimed to outperform regular SIFT descriptors. Finally, shape contexts use circular bins, similar to those used in C-HOG blocks, but only tabulate votes on the basis of edge presence, making no distinction with regards to orientation. Shape contexts were originally used in 2001 by Belongie, Malik, and Puzicha.

The testing commenced on two different data sets. The Massachusetts Institute of Technology (MIT) pedestrian database contains 509 training images and 200 test images of pedestrians on city streets. The set only contains images featuring the front or back of human figures and contains little variety in human pose. The set is well-known and has been used in a variety of human detection experiments, such as those conducted by Papageorgiou and Poggio in 2000. The MIT database is currently available for research at <http://cbcl.mit.edu/cbcl/software-datasets/PedestrianData.html>. The second set was developed by Dalal and Triggs exclusively for their human detection experiment due to

the fact that the HOG descriptors performed near-perfectly on the MIT set. Their set, known as INRIA, contains 1805 images of humans taken from personal photographs. The set contains images of humans in a wide variety of poses and includes difficult backgrounds, such as crowd scenes, thus rendering it more complex than the MIT set. The INRIA database is currently available for research at <http://lear.inrialpes.fr/data>.

The above site has an image showing examples from the INRIA human detection database.

As for the results, the C-HOG and R-HOG block descriptors perform comparably, with the C-HOG descriptors maintaining a slight advantage in the detection miss rate at fixed false positive rates across both data sets. On the MIT set, the C-HOG and R-HOG descriptors produced a detection miss rate of essentially zero at a 10^{-4} false positive rate. On the INRIA set, the C-HOG and R-HOG descriptors produced a detection miss rate of roughly 0.1 at a 10^{-4} false positive rate. The generalized Haar wavelets represent the next highest performing approach: they produced roughly a 0.01 miss rate at a 10^{-4} false positive rate on the MIT set, and roughly a 0.3 miss rate on the INRIA set. The PCA- SIFT descriptors and shape context descriptors both performed fairly poorly on both data sets. Both methods produced a miss rate of 0.1 at a 10^{-4} false positive rate on the MIT set and nearly a miss rate of 0.5 at a 10^{-4} false positive rate on the INRIA set.

SUPPORT VECTOR MACHINE

support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide

as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

When data are not labeled, supervised learning is not possible, and an unsupervised learning approach is required, which attempts to find natural clustering of the data to groups, and then map new data to these formed groups. The clustering algorithm which provides an improvement to the support vector machines is called support vector clustering and is often used in industrial applications either when data are not labeled or when only some data are labeled as a preprocessing for a classification pass.

Motivation

Classifying data is a common task in machine learning. Suppose some given data points each belong to one of two classes, and the goal is to decide which class a new data point will be in. In the case of support vector machines, a data point is viewed as a p - dimensional vector (a list of p numbers), and we want to know whether we can separate such points with a $(p - 1)$ dimensional hyperplane. This is called a linear classifier. There are many hyperplanes that might classify the data. One reasonable choice as the best hyperplane is the one that represents the largest separation, or margin, between the two classes. So we choose the hyperplane so that the distance from it to the nearest data point on each side is maximized. If such a hyperplane exists, it is known as the maximum-margin hyperplane and the linear classifier it defines is known as a maximum margin classifier; or equivalently, the perceptron of optimal stability.

Definition

More formally, a support vector machine constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification,

regression, or other tasks. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier.

Whereas the original problem may be stated in a finite dimensional space, it often happens that the sets to discriminate are not linearly separable in that space. For this reason, it was proposed that the original finite-dimensional space be mapped into a much higher-dimensional space, presumably making the separation easier in that space. To keep the computational load reasonable, the mappings used by SVM schemes are designed to ensure that dot products may be computed easily in terms of the variables in the original space, by defining them in terms of a kernel function $k(x, y)$ selected to suit the problem. The hyperplanes in the higher-dimensional space are defined as the set of points whose dot product with a vector in that space is constant. The vectors defining the hyperplanes can be chosen to be linear combinations with parameters α_i of images of feature vectors x_i that occur in the data base. With this choice of a hyperplane, the points x in the feature space that are mapped into the hyperplane are defined by the relation: $\sum_i \alpha_i k(x_i, x) = \text{constant}$. Note that if $k(x, y)$ becomes small as y grows further away from x , each term in the sum measures the degree of closeness of the test point x to the corresponding data base point x_i . In this way, the sum of kernels above can be used to measure the relative nearness of each test point to the data points originating in one or the other of the sets to be discriminated. Note the fact that the set of points x mapped into any hyperplane can be quite convoluted as a result, allowing much more complex discrimination between sets which are not convex at all in the original space.

Applications

SVMs can be used to solve various real world problems:

- SVMs are helpful in text and hypertext categorization as their application can significantly reduce the need for labeled training instances in both the standard inductive and transductive settings.
- Classification of images can also be performed using SVMs. Experimental results show that SVMs achieve significantly higher search accuracy than traditional query refinement schemes after just three to four rounds of relevance feedback. This is also true of image segmentation systems, including those using a modified version SVM that uses the privileged approach as suggested by Vapnik.
- Hand-written characters can be recognized using SVM.
- The SVM algorithm has been widely applied in the biological and other sciences. They have been used to classify proteins with up to 90% of the compounds classified correctly. Permutation tests based on SVM weights have been suggested as a mechanism for interpretation of SVM models. Support vector machine weights have also been used to interpret SVM models in the past. Posthoc interpretation of support vector machine models in order to identify features used by the model to make predictions is a relatively new area of research with special significance in the biological sciences.

Properties

SVMs belong to a family of generalized linear classifiers and can be interpreted as an extension of the perceptron. They can also be considered a special case of Tikhonov regularization. A special property is that they simultaneously minimize the empirical classification error and maximize the geometric margin; hence they are also known as maximum margin classifiers.

Parameter selection

The effectiveness of SVM depends on the selection of kernel, the kernel's parameters, and soft margin parameter C . A common choice is a Gaussian kernel, which

has a single parameter γ . The best combination of C and γ is often selected by a grid search with exponentially growing sequences of C and γ , for example, $C \in \{2^{-5}, 2^{-3}, \dots, 2^{13}, 2^{15}\}$. Typically, each combination of parameter choices is checked using cross validation, and the parameters with best cross-validation accuracy are picked. Alternatively, recent work in Bayesian optimization can be used to select C and γ , often requiring the evaluation of far fewer parameter combinations than grid search. The final model, which is used for testing and for classifying new data, is then trained on the whole training set using the selected parameters.

Issues

Potential drawbacks of the SVM include the following aspects:

- Requires full labeling of input data
- Uncalibrated class membership probabilities
- The SVM is only directly applicable for two-class tasks. Therefore, algorithms that reduce the multi-class task to several binary problems have to be applied; see the multi-class SVM section.
- Parameters of a solved model are difficult to interpret.

Extensions

Support vector clustering (SVC)

SVC is a similar method that also builds on kernel functions but is appropriate for unsupervised learning and data-mining. It is considered a fundamental method in data science.

Multiclass SVM

Multiclass SVM aims to assign labels to instances by using support vector machines, where the labels are drawn from a finite set of several elements.

The dominant approach for doing so is to reduce the single multiclass problem into multiple binary classification problems. Common methods for such reduction include:

- Building binary classifiers which distinguish (i) between one of the labels and the rest (one-versus-all) or (ii) between every pair of classes (one-versus-one). Classification of new instances for the one-versus-all case is done by a winner-takes-all strategy, in which the classifier with the highest output function assigns the class (it is important that the output functions be calibrated to produce comparable scores). For the one-versus-one approach, classification is done by a max-wins voting strategy, in which every classifier assigns the instance to one of the two classes, then the vote for the assigned class is increased by one vote, and finally the class with the most votes determines the instance classification.
- Directed acyclic graph SVM(DAGSVM)
- Error-correcting output codes

Crammer and Singer proposed a multiclass SVM method which casts the multiclass classification problem into a single optimization problem, rather than decomposing it into multiple binary classification problems. See also Lee, Lin and Wahba.

Structured SVM

SVMs have been generalized to structured SVMs, where the label space is structured and of possibly infinite size.

Implementation

The parameters of the maximum-margin hyperplane are derived by solving the optimization. There exist several specialized algorithms for quickly solving the QP problem that arises from SVMs, mostly relying on heuristics for breaking the problem down into smaller, more-manageable chunks.

Another common method is Platt's sequential minimal optimization (SMO) algorithm, which breaks the problem down into 2-dimensional sub-problems that are solved analytically, eliminating the need for a numerical optimization algorithm and matrix storage. This algorithm is conceptually simple, easy to implement, generally faster, and has better scaling properties for difficult SVM problems.

The special case of linear support vector machines can be solved more efficiently by the same kind of algorithms used to optimize its close cousin, logistic regression; this class of algorithms includes sub-gradient descent (e.g., PEGASOS) and coordinate descent (e.g., LIBLINEAR). LIBLINEAR has some attractive training time properties. Each convergence iteration takes time linear in the time taken to read the train data and the iterations also have a Q-Linear Convergence property, making the algorithm extremely fast.

The general kernel SVMs can also be solved more efficiently using sub-gradient descent (e.g. P-packSVM), especially when parallelization is allowed.

Kernel SVMs are available in many machine learning toolkits, including LIBSVM, MATLAB, SAS, SVMlight, kernlab, scikit-learn, Shogun, Weka, Shark, JKernelMachines, OpenCV and others.

CHAPTER -4

SOFTWARE AND HARDWARE

SYSTEM REQUIREMENTS FOR MATLAB

Operating Systems

Windows 10

Windows 8.1

Windows 8

Windows 7 Service Pack 1

Windows Server2016

Windows Server2012

Windows Server 2008 R2 Service Pack 1

Processors

Any Intel or AMD x86-64 processor

AVX2 instruction set support is recommended

With Polyspace, 4 cores is recommended

DiskSpace

2 GB for MATLAB only,

4–6 GB for a typical installation.

RAM

2GB

With Simulink, 4 GB is required

With Polyspace, 4 GB per core is recommended

Graphics

No specific graphics card is required.

Hardware accelerated graphics card supporting OpenGL 3.3 with 1GB GPU memory

MATLAB

What Is MATLAB?

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

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

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar noninteractive language such as C or Fortran.

The name MATLAB stands for *matrix laboratory*. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects. Today, MATLAB uses software developed by the LAPACK and ARPACK projects, which together represent the state-of-the-art in software for matrix computation. MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis.

Toolboxes

MATLAB features a family of application-specific solutions called *toolboxes*. Very important to most users of MATLAB, toolboxes allow you to learn and apply

specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others.

The MATLAB System

The MATLAB system consists of five main parts:

Development Environment.

This is the set of tools and facilities that help you use MATLAB functions and files. Many of these tools are graphical user interfaces.

It includes the MATLAB desktop and Command Window, a command history, and browsers for viewing help, the workspace, files, and the search path.

The MATLAB Mathematical Function Library.

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

The MATLAB language.

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

Handle Graphics®.

This is the MATLAB graphics system. It includes high-level commands for two-dimensional and three-dimensional data visualization, image processing, animation, and

presentation graphics. It also includes low-level commands that allow you to fully customize the appearance of graphics as well as to build complete graphical user interfaces on your MATLAB applications.

The MATLAB Application Program Interface (API).

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

Image processing in matlab

Images and pictures. As we mentioned in the preface, human beings are predominantly visual creatures: we rely heavily on our vision to make sense of the world around us. We not only look at things to identify and classify them, but we can scan for differences, and obtain an overall rough *_feeling_* for a scene with a quick glance. Humans have evolved very precise visual skills: we can identify a face in an instant; we can differentiate colours; we can process a large amount of visual information very quickly.

However, the world is in constant motion: stare at something for long enough and it will change in some way. Even a large solid structure, like a building or a mountain, will change its appearance depending on the time of day (day or night); amount of sunlight (clear or cloudy), or various shadows falling upon it. We are concerned with single images: snapshots, if you like, of a visual scene. Although image processing can deal with changing scenes, we shall not discuss it in any detail in this text.

For our purposes, an image is a single picture which represents something. It may be a picture of a person, of people or animals, or of an outdoor scene, or a microphotograph of an electronic component, or the result of medical imaging. Even if the picture is not immediately recognizable, it will not be just a random blur.

What is image processing?

Image processing involves changing the nature of an image in order to either

1. improve its pictorial information for human interpretation,
2. render it more suitable for autonomous machine perception.

We shall be concerned with digital image processing, which involves using a computer to change the nature of a digital image (see below). It is necessary to realize that these two aspects represent two separate but equally important aspects of image processing. A procedure which satisfies condition,

(1) A procedure which makes an image look better may be the very worst procedure for satisfying condition

(2). Humans like their images to be sharp, clear and detailed; machines prefer their images to be simple and uncluttered.

Some applications

Image processing has an enormous range of applications; almost every area of science and technology can make use of image processing methods. Here is a short list just to give some indication of the range of image processing applications.

1. Medicine

Inspection and interpretation of images obtained from X-rays, MRI or CAT scans, analysis of cell images, of chromosome karyotypes.

2. Agriculture

Satellite/aerial views of land, for example to determine how much land is being used for different purposes, or to investigate the suitability of different regions for different crops, inspection of fruit and vegetables_distinguishing good and fresh produce from old.

3. Industry

Automatic inspection of items on a production line,
inspection of paper samples.

4. Lawenforcement

Fingerprint analysis,
sharpening or de-blurring of speed-camera images.

CHAPTER-5

CONCLUSION

Pulmonary tuberculosis (TB) persists as a great public health problem in Korea. Increases in the overall age of the population and the rise of drug-resistant TB have reinforced the need for rapid diagnostic improvements and new modalities to detect TB and drug-resistant TB, as well as to improve TB control. An automatic method is presented to detect abnormalities in frontal chest radiographs which are aggregated into an overall abnormality score. The method is aimed at finding abnormal signs of a diffuse textural nature, such as they are encountered in mass chest screening against tuberculosis (TB). The scheme starts with filtering the chest radiographs using median filter then the features of the filtered image will be extracted using Haar wavelet transform (HWT) and histogram of oriented gradients then the automatic segmentation of the lung fields will be done by using Otsu's thresholding algorithm. Finally the extracted features are compared with the trained datasets using support vector machine and the result will be shown whether then tuberculosis is affected or not. All the above process completed and lung images are successfully classified whether the lung image was affected or not by using support vectormachine

CHAPTER-6

REFERENCES

1. Stefan Jaeger, AlexandrosKarargyris, Serna Candemir, Les Folio, Jenifer Siegelman, Fiona Caliaghan,ZhiyunXue, KannappanPalaniappan, Rahul K. Singh,SameerAntani,GeorgeThoma,Yi•XiangWang,Pu-XuanLu,and Clement J. McDonald, "Automatic Tuberculosis Screening Using Chest Radiographs", IEEE transactions on medical imaging, vol. 33, no. 2, February 2014
2. Wai YanNyeinNaing, Zaw Z. Htike, "Advances in automatic tuberculosis detection in chest x-ray images", signal & image 802 processing an international journal (sipij) vol.5, no.6, December2014.
3. Rui Shen, , Irene Cheng, , and Anup 8asu, Senior Member, "A Hybrid Knowledge-Guided Detection Technique for Screening of Infectious Pulmonary Tuberculosis From Chest Radiographs". IEEE transactions on biomedical engineering, vol. 57, no. 11, November2010
4. Tao Xu, Irene Cheng, Senior Member IEEE, andMrinalMandai, Senior Member, IEEE, "Automated Cavity Detection of Infectious Pulmonary Tuberculosis in Chest Radiographs", 33rd Annual international Conference of the IEEE EMBS Boston, Massachusells USA, August 30 - September 3,2011.
5. Serna Candemir, Stefan Jaeger, KannappanPalaniappan, Sameer Antani, George Thoma, and Clement J. McDonald," Lung Segmentation in Chest Radiographs Using Anatomical Atlases With Nonrigid Registration", IEEE transactions on medical imaging, vol. 33, no. 2, february2014.
6. HrudyaDasl ,Ajay Nath2 , "An Efficient Detection of Tuberculosis from Chest X-rays ", international Journal of Advance Research in Computer Science and Management Studies", Volume 3, Issue 5, May 2015 pg.149-154.
7. P. Maduskar, H. Laurens, R. Philipsen, and B. Ginneken, "Automated localization of costophrenic recesses and costophrenic angle measurement on frontal chest radiographs," inProc. SPIE, 2013, vol.8670.
8. A.Gururajan, H. Sari-Sarraf, and E. Hequet, "Interactive texture segmentation via IT-SNAPS," inIEEE Southwest Symp. Image Anal. Interpret.,2010..

9. L. Hogeweg, C. Snchez, P. A. Jong, P. Maduskar, and B. Ginneken, "Clavicle segmentation in chest radiographs," *Med. Image Anal.*, vol.16, no. 8, pp. 1490–1502, 2012.
10. M. Rahman, S. Antani, D. Fushman, and G. Thoma, "Biomedical image retrieval using multimodal context and concept feature spaces," *Med. Content-Based Retrieval Clin. Decision Support*, vol. 7075, pp.24–35, 2012.
11. S. Candemir and Y. S. Akgul, "Statistical significance based graph cut regularization for medical image segmentation," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 19, no. 6, pp. 957–972, 2011.
12. C. Liu, J. Yuen, and A. Torralba, "SIFTflow: Dense correspondence across different scenes and its applications.," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, 2011.
13. A. Dawoud, "Lung segmentation in chest radiographs by fusing shape information in iterative thresholding," *IET Comput. Vis.*, vol. 5, no. 3, pp. 185–190, 2011.
14. S. Candemir, S. Jaeger, K. Palaniappan, S. Antani, and G. Thoma, "Graph-cut based automatic lung boundary detection in chest radiographs," in *IEEE Healthcare Technol. Conf.: Translat. Eng. Health Med.*, 2012, pp. 31–34.